Multi-Party Threshold Private Set Intersection with Sublinear Communication

Saikrishna Badrinarayanan[1], Peihan Miao[2], Srinivasan Raghuraman[1], and Peter Rindal[1]

[1] Visa Research, {sabadrin, srraghur, perindal}@visa.com

[2] University of Illinois at Chicago, peihan@uic.edu

Abstract

In multi-party threshold private set intersection (PSI), n parties each with a private set wish to compute the intersection of their sets if the intersection is sufficiently large. Previously, Ghosh and Simkin (CRYPTO 2019) studied this problem for the two-party case and demonstrated interesting lower and upper bounds on the communication complexity. In this work, we investigate the communication complexity of the multi-party setting ($n \geq 2$). We consider two functionalities for multi-party threshold PSI. In the first, parties learn the intersection if each of their sets and the intersection differ by at most T. In the second functionality, parties learn the intersection if the union of all their sets and the intersection differ by at most T.

For both functionalities, we show that any protocol must have communication complexity $\Omega(nT)$. We build protocols with a matching upper bound of $O(nT)$ communication complexity for both functionalities assuming threshold FHE. We also construct a computationally more efficient protocol for the second functionality with communication complexity $\widetilde{O}(nT)$ under a weaker assumption of threshold additive homomorphic encryption. As a direct implication, we solve one of the open problems in the work of Ghosh and Simkin (CRYPTO 2019) by designing a two-party protocol with communication cost $\widetilde{O}(T)$ from assumptions weaker than FHE.

As a consequence of our results, we achieve the first "regular" multi-party PSI protocol where the communication complexity only grows with the size of the set difference and does not depend on the size of the input sets.

1 Introduction

Private set intersection (PSI) protocols allow several mutually distrustful parties $P_1, P_2, \ldots, P_n$ each holding a private set $S_1, S_2, \ldots, S_n$ respectively to jointly compute the intersection $I = \bigcap_{i=1}^{n} S_i$ without revealing any other information. PSI has numerous privacy-preserving applications, e.g., DNA testing and pattern matching [TPKC07], remote diagnostics [BPSW07], botnet detection [NMH+10], online advertising [IKN+17, MPR+20]. Over the last years enormous progress has been made towards realizing this functionality efficiently [HFH99, FNP04, KS05, DCT10, DCW13, PSZ14, PSSZ15, KKRT16, OOS16, RR17, KMP+17, HV17, PSWW18, PRTY19, GN19, PRTY20, CM20] in the two-party, multi-party, and server-aided settings with both semi-honest and malicious security.

Threshold PSI. In certain scenarios, the standard PSI functionality is not sufficient. In particular, the parties may only be willing to reveal the intersection if they have a *large* intersection. For

example, in privacy-preserving data mining and machine learning [MZ17] where the data is vertically partitioned among multiple parties (that is, each party holds different features of the same object), the parties may want to learn the intersection of their datasets and start their collaboration only if their common dataset is sufficiently large. If their common dataset is too small, in which case they are not interested in collaboration, it is undesirable to let them learn the intersection. In privacy-preserving ride sharing [HOS17], multiple users only want to share a ride if large parts of their trajectories on a map intersect. In this case, the users may be interested in the intersection of their routes, but only when the intersection is large. This problem can be formalized as *threshold private set intersection*, where, roughly speaking, the parties only learn the intersection if their sets differ by at most T elements.

Many works [FNP04, HOS17, PSWW18, ZC18, GN19] achieve this functionality by first computing the cardinality of the intersection and then checking if this is sufficiently large. The communication complexity of these protocols scales at least linearly in the size of the smallest input set. Notice that Freedman et al. [FNP04] proved a lower bound of $\Omega(m)$ on the communication complexity of any private set intersection protocol, where m is the size of the smallest input set. This lower bound directly extends to protocols that only compute the cardinality of the intersection, which constitutes a fundamental barrier to the efficiency of the above protocols.

Recently, the beautiful work of Ghosh and Simkin [GS19a] revisited the communication complexity of *two-party* threshold PSI and demonstrated that the $\Omega(m)$ lower bound can be circumvented by performing a private intersection cardinality testing (i.e., testing whether the intersection is sufficiently large) instead of computing the actual cardinality. After passing the cardinality testing, their protocol allows each party to learn the set difference, where the communication complexity only grows with T, which could be sublinear in m. Specifically, [GS19a] proved a communication lower bound of $\Omega(T)$ for two-party threshold PSI and presented a protocol achieving a matching upper bound $O(T)$ based on fully homomorphic encryption (FHE). They also showed a computationally more efficient protocol with communication complexity of $\widetilde{O}(T^2)$ based on weaker assumptions, namely additively homomorphic encryption (AHE).

In this work, we investigate the communication complexity of *multi-party* threshold PSI. In particular, we ask the question of whether sublinear lower and upper bounds can also be achieved in the multi-party setting.

1.1 Our Contributions

We first identify and formalize the definition of multi-party threshold private set intersection. We put forth and study *two* functionalities that are in fact equivalent in the two-party case but are vastly different in the multi-party scenario. Assume there are n parties $P_1, P_2, \ldots, P_n$, and each party P_i holds a private set S_i of size m. The first functionality allows the parties to learn the intersection $I = \bigcap_{i=1}^{n} S_i$ only if $\forall i, |S_i \setminus I| \leq T$, or equivalently, $|I| \geq m - T$. In the second functionality, the parties can learn the intersection I only if $|\left(\bigcup_{i=1}^{n} S_i\right) \setminus I| \leq T$.

We briefly discuss the difference between the two functionalities. The first functionality focuses on whether the intersection is sufficiently large, hence we call it $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$. The second functionality focuses on whether the set difference is sufficiently small, thus we call it $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$. In the two-party case, we have the guarantee that $|\left(\bigcup_{i=1}^{n} S_i\right) \setminus I| = 2 \cdot |S_i \setminus I|$, so we do *not* have to differentiate between these two functionalities. However, in the multi-party case, we only know that $2 \cdot |S_i \setminus I| \leq |\left(\bigcup_{i=1}^{n} S_i\right) \setminus I| \leq n \cdot |S_i \setminus I|$, hence the two functionalities could lead to very different outcomes. Which functionality to choose and what threshold to set in practice highly depend on the actual

application.

Sublinear Communication. The core contribution of this work is demonstrating sublinear (in the set sizes) communication lower and upper bounds for both functionalities. We summarize our results in Table 1. For lower bound, we prove that both functionalities require at least $\Omega(nT)$ bits of communication. For upper bound, we present protocols for both functionalities achieving a matching upper bound of $O(nT)$ based on n-out-of-n threshold fully homomorphic encryption (TFHE) [BGG+18]. We also give a computationally more efficient protocol based on weaker assumptions, namely n-out-of-n threshold additively homomorphic encryption (TAHE) [Ben94, Pai99], with communication complexity of $\widetilde{O}(nT)$ that almost matches the lower bound.[1] All these protocols achieve semi-honest security where up to $(n-1)$ parties could be corrupted.

Functionality	Communication Lower Bound	TFHE-based Upper Bound	TAHE-based Upper Bound
$\mathcal{F}_{\mathsf{TPSI\text{-}int}}$	$\Omega(nT)$	$O(nT)$	unknown
$\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$	$\Omega(nT)$	$O(nT)$	$\widetilde{O}(nT)$

Table 1: Communication lower and upper bounds for multi-party threshold PSI.

Our Protocols. As summarized in Table 1, we present three protocols for upper bounds, one for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and two for $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$. At a high level, all three protocols compute their functionality in two phases. In the first phase, they perform a multi-party private intersection cardinality testing where the parties jointly decide whether their intersection is sufficiently large. In particular, for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$, the cardinality testing, which we call $\mathcal{F}_{\mathsf{CTest\text{-}int}}$, allows all the parties to learn whether $|I| \geq (m-T)$. For $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$, the cardinality testing, which we call $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$, allows all the parties to learn whether $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. The communication complexity of our protocols for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ and $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ is summarized in Table 2. In particular, for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$, we present a protocol with communication complexity $O(nT)$ based on TFHE. For $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$, we show a TFHE-based construction with communication complexity $O(nT)$ and a TAHE-based construction with communication complexity $\widetilde{O}(nT)$.

Functionality	TFHE-based Protocol	TAHE-based Protocol
$\mathcal{F}_{\mathsf{CTest\text{-}int}}$	$O(nT)$	unknown
$\mathcal{F}_{\mathsf{CTest\text{-}diff}}$	$O(nT)$	$\widetilde{O}(nT)$

Table 2: Communication complexity of our protocols for multi-party private cardinality testing.

If the intersection is sufficiently large, namely it passes the cardinality testing, then the parties start the second phase of our protocols, which allows each party P_i to learn their set difference $S_i \setminus I$. We present a singe protocol for the second phase, which works for both $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and

[1] $\widetilde{O}(\cdot)$ hides $\mathsf{polylog}$ factors. All the upper bounds omit a $\mathsf{poly}(\lambda)$ factor where λ is the security parameter.

$\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$. The second-phase protocol is based on TAHE and has communication complexity of $O(nT)$. Thus, to construct a protocol for multi-party threshold PSI, we combine the first-phase protocols summarized in Table 2 with the second-phase one described above. Doing so, we achieve the communication upper bounds in Table 1.

This modular design enables our constructions to minimize the use of TFHE as it is not needed in the second phase. Moreover, it allows future work to focus on improving Table 2. In particular, to design a protocol for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ from assumptions weaker than TFHE, future work could focus on building protocols for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ and directly plug in our second phase protocol after that.

Communication Topology. All our protocols are designed in the so-called *star network* topology, where a designated party communicates with every other party. An added benefit of this topology is that not all parties must be online at the same time. Our communication lower bounds are proved in *point-to-point* fully connected networks, which are a generalization of the star network.

For networks with broadcast channels, we prove another communication lower bound of $\Omega(T \log n + n)$ for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ in Appendix C and leave further exploration in the broadcast model for future work.

1.2 Other Implications

Two-Party Threshold PSI. Recall that in the two-party case, both functionalities $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ are identical. Ghosh and Simkin [GS19a] built a two-party threshold PSI protocol from AHE with communication complexity $\widetilde{O}(T^2)$. They left it as an open problem to build a two-party threshold PSI protocol with communication complexity $\widetilde{O}(T)$ from assumptions weaker than FHE. Observe that for the special case of $n = 2$, we can achieve a two-party threshold PSI protocol with communication complexity $\widetilde{O}(T)$ from AHE thereby solving this open problem (refer to Section 6 and Section 7 for more details).

Sublinear Communication PSI. Our multi-party threshold PSI protocols for both $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ can also be used to achieve multi-party "regular" PSI[2] where the communication complexity only grows with the size of the set difference and independent of the input set sizes. In particular, if we run a sequence of multi-party threshold PSI protocols on $T = 2^0, 2^1, 2^2, \ldots$ until hitting the smallest $T = 2^k$ where the protocol outputs the intersection, then we can achieve multi-party PSI. The communication complexity of the resulting protocol is a factor $\log T$ times that of a single instance but still independent of the input set sizes. Therefore, when the intersection is very large, namely the set difference is significantly smaller than the set sizes, this new approach achieves the *first* multi-party PSI with sublinear (in the set sizes) communication complexity.

Compact MPC. It is an open problem to construct a *compact* MPC protocol in the plain model where the communication complexity does not grow with the output length of the function. Prior works [HW15, BFK+19] construct compact MPC for general functions in the presence of a trusted setup (CRS, random oracle) from strong computational assumptions such as obfuscation. Our multi-party threshold PSI protocols have communication complexity independent of the output size (the set intersection). To the best of our knowledge, ours are the *first* compact MPC protocols for any non-trivial function in the plain model. The only prior compact protocol in the plain model we are aware of is the two-party threshold PSI protocol [GS19a].

[2] By "regular" PSI, we refer to the standard notion of PSI without threshold.

1.3 Concurrent and Independent Work

Concurrent to our work, a recent update to the full version of the paper by Ghosh and Simkin [GS19b] extends the two-party threshold PSI protocol to the multi-party setting and consider the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$. They do not consider the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ that we additionally consider in our work. For $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$, [GS19b] also first constructs a TFHE-based protocol for the intersection cardinality testing $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ with communication complexity $O(nT)$. Then in the second phase for computing the intersection, they use an MPC protocol to compute the evaluations of a random polynomial, where the communication complexity depends on how the MPC is instantiated, which is not discussed

Another concurrent work by Branco, Döttling, and Pu [BDP21] studies multi-party private intersection cardinality testing with the functionality $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ and presents a TAHE-based protocol with communication complexity $\widetilde{O}(nT^2)$, which complements our Table 2. They also do not consider the other functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$.

1.4 Roadmap

We describe some notations and definitions in Section 2, a technical overview in Section 3, and the lower bound in Section 4. We present the TFHE based protocols for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ and $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ in Section 5 and the TAHE based protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ in Section 6. We present the second phase protocol to compute the actual intersection in Section 7.

2 Preliminaries

2.1 Notations

We use λ to denote the security parameters. By $\mathsf{poly}(\lambda)$ we denote a polynomial function in λ. By $\mathsf{negl}(\lambda)$ we denote a negligible function, that is, a function f such that $f(\lambda) < 1/p(\lambda)$ holds for any polynomial $p(\cdot)$ and sufficiently large λ. We use $[\![x]\!]$ to denote an encryption of x. We use $\widetilde{O}(x)$ to ignore any $\mathsf{polylog}$ factor, namely $\widetilde{O}(x) = O(x \cdot \mathsf{polylog}(x))$.

2.2 Secure Multi-Party Computation

We follow the security definitions for secure multi-party computation (MPC) in the universal composability (UC) framework [Can01]. We provide a brief overview here and refer the reader to [Can01] for more details of the security model.

We consider a protocol Π amongst n parties that implements an ideal functionality $\mathcal{F}$. We define security in the real/ideal world paradigm. In the real execution, the parties execute the protocol Π, which is allowed to call any ideal functionality $\mathcal{G}$. An environment $\mathcal{Z}$ chooses the inputs of all parties, models everything that is external to the protocol, and represents the adversary. $\mathcal{Z}$ may corrupt any subset (not all) of the parties and get access to those parties' internal tapes. In the ideal execution, n dummy parties send their inputs to the ideal functionality $\mathcal{F}$ and get back the output of the computation. A simulator Sim, also known as the ideal world adversary, emulates $\mathcal{Z}$'s view of a real protocol execution. Sim has full control of the corrupted dummy parties and emulates $\mathcal{Z}$'s view of those parties. Let $\mathsf{Real}[\mathcal{Z}, \Pi, \mathcal{G}]$ and $\mathsf{Ideal}[\mathcal{Z}, \mathsf{Sim}, \mathcal{F}]$ be the output of $\mathcal{Z}$ in the real and ideal executions, respectively. We say the protocol Π securely implements $\mathcal{F}$ if for any

PPT $\mathcal{Z}$, there exists a PPT Sim such that

$$\mathsf{Real}[\mathcal{Z}, \Pi, \mathcal{G}] \overset{c}{\approx} \mathsf{Ideal}[\mathcal{Z}, \mathsf{Sim}, \mathcal{F}].$$

2.3 Multi-Party Threshold Private Set Intersection

Setting. Consider n parties $P_1, \ldots, P_n$ with input sets $S_1, \ldots, S_n$ respectively. Throughout the paper, we consider all the sets to be of equal size m. We assume that the set elements come from a field $\mathbb{F}_p$, where p is a $\Theta(\lambda)$-bit prime. Also, throughout the paper, we focus only on the point-to-point network channels. For the lower bounds, we consider a setting where every pair of parties has a point-to-point channel between them. For the upper bounds, we consider a more restrictive model – the star network, where only one central party has a point-to-point channel with every other party and the other parties cannot communicate with each other.

The goal of the parties is to run an MPC protocol Π at the end of which each party learns the intersection I of all the sets if certain conditions hold. In the definition of two-party threshold PSI, both parties P_1 and P_2 learn the intersection I if the size of their set difference is small, namely $|(S_1 \setminus S_2) \cup (S_2 \setminus S_1)| < 2T$. In the multi-party case, we consider two different functionalities, each of which might be better suited to different applications.

Functionalities. In the first definition, we consider functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$, in which each party P_i learns the intersection I if the size of its own set minus the intersection is small, namely $|S_i \setminus I| \leq T$ for some threshold T. Recall that we consider all the sets to be of equal size, hence either all the parties learn the output or all of them don't. In the second definition, we consider a functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$, where each party learns the intersection I if the size of the union of all the sets minus the intersection is small, namely $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. The formal definitions of the two ideal functionalities are shown in Figure 1 and Figure 2.

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$.
Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}_p$ for all $j \in [m]$.
Output: Each party P_i receives $I = \bigcap_{i=1}^{n} S_i$ if and only if $\lvert S_i \setminus I \rvert \leq T$.

Figure 1: Ideal functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ for multi-party threshold PSI.

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$.
Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}_p$ for all $j \in [m]$.
Output: Each party P_i receives $I = \bigcap_{i=1}^{n} S_i$ if and only if $\lvert (\bigcup_{i=1}^{n} S_i) \setminus I \rvert \leq T$.

Figure 2: Ideal functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ for multi-party threshold PSI.

2.4 Multi-Party Private Intersection Cardinality Testing

An important building block in our multi-party threshold PSI protocols is a multi-party protocol for private intersection cardinality testing which we define below. Consider n parties $P_1, \ldots, P_n$ with input sets $S_1, \ldots, S_n$ respectively of equal size m. Their goal is to run an MPC protocol Π at the end of which each party learns whether the size of the intersection I of all the sets is sufficiently

large. As before, we consider two functionalities. In the first functionality $\mathcal{F}_{\mathsf{CTest\text{-}int}}$, each party P_i learns whether $|S_i \setminus I| \leq T$. In the second functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$, each party learns whether $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. The formal definitions of the two ideal functionalities are presented in Figure 3 and Figure 4.

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}_p$ for all $j \in [m]$.

Output: Each party P_i receives $\mathsf{similar}$ if $|S_i \setminus I| \leq T$ and $\mathsf{different}$ otherwise where $I = \bigcap_{i=1}^{n} S_i$.

Figure 3: Ideal functionality $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ for multi-party intersection cardinality test.

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}_p$ for all $j \in [m]$.

Output: Each party P_i receives $\mathsf{similar}$ if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$ and $\mathsf{different}$ otherwise where $I = \bigcap_{i=1}^{n} S_i$.

Figure 4: Ideal functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ for multi-party intersection cardinality test.

2.5 Threshold Fully Homomorphic Encryption

We define the notion of a threshold fully homomorphic encryption (TFHE) scheme with distributed setup introduced in the work of Boneh et al. [BGG+18]. Also, throughout the paper, we are only interested in the n-out-of-n threshold setting.

Definition 2.1. ***(Threshold Fully Homomorphic Encryption (TFHE) with Distributed Setup)*** *Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be a set of parties. A TFHE scheme consists of a tuple of PPT algorithms* $\mathsf{TFHE} = (\mathsf{TFHE.DistSetup}, \mathsf{TFHE.Enc}, \mathsf{TFHE.Eval}, \mathsf{TFHE.PartialDec}, \mathsf{TFHE.Combine})$ *with the following syntax:*

- $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, i)$*: On input the security parameter λ and a party index i, the distributed setup algorithm outputs the parameters associated with the i-th party: a component of the public key pk_i, and a share of the secret key sk_i. We denote the public key of the scheme pk to be $(\mathsf{pk}_1 || \ldots || \mathsf{pk}_n)$.*
- $[\![m]\!] \leftarrow \mathsf{TFHE.Enc}(\mathsf{pk}, m)$*: On input a public key pk and a plaintext m in the message space $\mathcal{M}$, the encryption algorithm outputs a ciphertext $[\![m]\!]$.*
- $[\![y]\!] \leftarrow \mathsf{TFHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!])$*: On input a public key pk, a circuit C of polynomial size that takes k inputs each from the message space and outputs one value in the message space, and a set of ciphertexts $[\![m_1]\!], \ldots, [\![m_k]\!]$, the evaluation algorithm outputs a ciphertext $[\![y]\!]$.*
- $[\![m : \mathsf{sk}_i]\!] \leftarrow \mathsf{TFHE.PartialDec}(\mathsf{sk}_i, [\![m]\!])$*: On input a a secret key share sk_i and a ciphertext $[\![m]\!]$, the partial decryption algorithm outputs a partial decryption $[\![m : \mathsf{sk}_i]\!]$.*
- $m/\perp \leftarrow \mathsf{TFHE.Combine}(\mathsf{pk}, \{[\![m : \mathsf{sk}_i]\!]\}_{i \in [n]})$*: On input a public key pk and a set of partial decryptions $\{[\![m : \mathsf{sk}_i]\!]\}_{i \in [n]}$, the combination algorithm either outputs a plaintext m or the symbol $\perp$.*

As in a standard homomorphic encryption scheme, we require that a TFHE scheme satisfies compactness, correctness and security. We discuss these properties below.

Compactness. A TFHE scheme is said to be compact if there exists polynomials $\mathsf{poly}_1(\cdot)$ and $\mathsf{poly}_2(\cdot)$ such that for all λ, all message spaces $\mathcal{M}$ with size of each message being $\mathsf{poly}_3(\lambda)$, circuit C of size at most $\mathsf{poly}_4(\lambda)$ and $m_i \in \mathcal{M}$ for $i \in [k]$, the following condition holds. For $\{(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, j)\}_{j\in[n]}$, $[\![m_i]\!] \leftarrow \mathsf{TFHE.Enc}(\mathsf{pk}, m_i)$ for $i \in [k]$, $[\![y]\!] \leftarrow \mathsf{TFHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!])$, $[\![y : \mathsf{sk}_j]\!] = \mathsf{TFHE.PartialDec}([\![y]\!], \mathsf{sk}_j)$ for $j \in [n]$, $|\,[\![y]\!]\,| \leq \mathsf{poly}_1(\lambda)$ and $|\,[\![y : \mathsf{sk}_j]\!]\,| \leq \mathsf{poly}_2(\lambda)$.[3]

Evaluation Correctness. Informally, a TFHE scheme is said to be correct if recombining partial decryptions of a ciphertext output by the evaluation algorithm returns the correct evaluation of the corresponding circuit on the underlying plaintexts. Formally, We say that a TAHE scheme satisfies evaluation correctness if for all λ, all message spaces $\mathcal{M}$, circuit C and $m_i \in \mathcal{M}$ for $i \in [k]$, the following condition holds. For $\{(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, j)\}_{j\in[n]}$, $[\![m_i]\!] \leftarrow \mathsf{TFHE.Enc}(\mathsf{pk}, m_i)$ for $i \in [k]$, $[\![y]\!] \leftarrow \mathsf{TFHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!])$,

$$\Pr[\mathsf{TFHE.Combine}(\mathsf{pk}, \mathsf{TFHE.PartialDec}([\![y]\!], \mathsf{sk}_i)_{i\in[n]}) = \mathsf{C}(m_1, \ldots, m_k)] = 1 - \mathsf{negl}(\lambda).$$

Semantic Security. Informally, a TFHE scheme is said to provide semantic security if any PPT adversary cannot distinguish between encryptions of arbitrarily chosen plaintext messages m_0 and m_1, even given the secret key shares corresponding to a subset $\mathcal{S}$ of the parties for any set $\mathcal{S}$ of size at most $(n-1)$. Formally, a TFHE scheme satisfies semantic security if for all λ, message space $\mathcal{M}$, for any PPT adversary $\mathcal{A}$, $\Pr[\mathsf{Expt}_{\mathsf{TFHE},\mathsf{sem}}(1^\lambda) = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ where the experiment $\mathsf{Expt}_{\mathsf{TFHE},\mathsf{sem}}(1^\lambda)$ is defined as:

$\underline{\mathsf{Expt}_{\mathsf{TFHE},\mathsf{sem}}(1^\lambda)}$:

1. On input the security parameter 1^λ, the message space $\mathcal{M}$ and the number of parties n, the adversary $\mathcal{A}$ does the following: Pick a set $\mathcal{S}$ of size at most $(n-1)$. For each $i \in \mathcal{S}$, compute $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TAHE.DistSetup}(1^\lambda, i)$. Pick two messages (m_0, m_1). Send $(\mathcal{S}, \{\mathsf{pk}_i, \mathsf{sk}_i\}_{i\in\mathcal{S}}, m_0, m_1)$ to the challenger.

2. The challenger runs $\{(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, j)\}_{j\in[n]\setminus\mathcal{S}}$, and provides $\{\mathsf{pk}_i\}_{i\in[n]\setminus\mathcal{S}}$) along with $\mathsf{TFHE.Enc}(\mathsf{pk}, m_b)$ to $\mathcal{A}$ where b is picked uniformly at random and $\mathsf{pk} = (\mathsf{pk}_1 || \ldots || \mathsf{pk}_n)$.

3. $\mathcal{A}$ outputs a guess b'. The experiment outputs 1 if $b = b'$.

Simulation Security. Informally, a TFHE scheme is said to provide simulation security if there exists an efficient algorithm $\mathsf{TFHE.Sim}$ that takes as input a circuit C, a ciphertext $[\![y]\!]$ that is computed by running the $\mathsf{TFHE.Eval}$ algorithm on a set of ciphertexts $[\![m_1]\!], \ldots, [\![m_k]\!]$, the output of C on the corresponding plaintexts, and outputs a set of partial decryptions corresponding to

[3]Note that in the definition given in [BGG+18], the size of the partial decryption also grows with n for arbitrary t-out-of-n threshold schemes. Here, we focus only on the n-out-of-n threshold, in which case there is no dependence on n for the sizes. Also, the definition given in [BGG+18] specifies an upper bound d on the circuit depth and the size of partial decryption grows with the circuit depth. We do not include d in our definition because we will rely on circular-secure LWE to achieve the "strong" compactness.

some subset of parties, such that its output is computationally indistinguishable from the output of a real algorithm that homomorphically evaluates the circuit C on the ciphertexts $[\![m_1]\!], \ldots, [\![m_k]\!]$ and outputs partial decryptions using the corresponding secret key shares for the same subset of parties. In particular, the computational indistinguishability holds even when a PPT adversary is given the secret key shares corresponding to a subset $\mathcal{S}$ of the parties, so long as $\mathsf{TFHE.Sim}$ also gets the secret key shares corresponding to the set of parties in $\mathcal{S}$.

Formally, a TFHE scheme satisfies simulation security if for all λ, message space $\mathcal{M}$, for any PPT adversary $\mathcal{A}$, there exists a simulator $\mathsf{TFHE.Sim}$ such that the following two experiments $\mathsf{Expt}_{\mathsf{TFHE,Real}}(1^\lambda)$ and $\mathsf{Expt}_{\mathsf{TFHE,Ideal}}(1^\lambda)$ are computationally indistinguishable.

$\underline{\mathsf{Expt}_{\mathsf{TFHE,Real}}(1^\lambda)}$:

1. On input of the security parameter 1^λ, the message space $\mathcal{M}$ and the number of parties n, the adversary $\mathcal{A}$ does the following: Pick a set $\mathcal{S}$ of size at most $(n-1)$. For each $i \in \mathcal{S}$, compute $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TAHE.DistSetup}(1^\lambda, i)$. Send $(\mathcal{S}, \{\mathsf{pk}_i, \mathsf{sk}_i\}_{i\in\mathcal{S}})$ to the challenger.
2. The challenger runs $\{(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, 1^d, j)\}_{j\in[n]\setminus\mathcal{S}}$, and provides $(\{\mathsf{pk}_i\}_{i\in[n]\setminus\mathcal{S}})$ to $\mathcal{A}$. Set $\mathsf{pk} := (\mathsf{pk}_1 || \ldots || \mathsf{pk}_n)$.
3. The adversary picks a set of messages $m_1, \ldots, m_k$ for $k = \mathsf{poly}(\lambda)$ and a set $\mathcal{S}_1 \subseteq [k]$. It computes $[\![m_i]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, m_i; r_i)$ for each $i \in \mathcal{S}_1$ and sends $(m_1, \ldots, m_k, \{r_i\}_{i\in[\mathcal{S}_1]})$ to the challenger.
4. The challenger computes and sends $[\![m_i]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, m_i; r_i)$ to $\mathcal{A}$ for each $i \in [k] \setminus \mathcal{S}_1$.
5. $\mathcal{A}$ issues a query with a circuit C. The challenger first computes $[\![y]\!] := \mathsf{TFHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!])$ where $[\![m_i]\!] = \mathsf{TFHE.Enc}(\mathsf{pk}, m_i; r_i)$. Then, it outputs $\{\mathsf{TFHE.PartialDec}(\mathsf{sk}_i, [\![y]\!])\}_{i\notin\mathcal{S}}$ to $\mathcal{A}$.
6. The adversary may repeat step 5 $\mathsf{poly}(\lambda)$ many times.
7. At the end of the experiment, $\mathcal{A}$ outputs a distinguishing bit b.

$\underline{\mathsf{Expt}_{\mathsf{TFHE,Ideal}}(1^\lambda)}$:

1. Perform steps 1-4 of the real world experiment $\mathsf{Expt}_{\mathsf{TFHE,Real}}(1^\lambda)$.
2. $\mathcal{A}$ issues a query with a circuit C. The challenger first computes $[\![y]\!] := \mathsf{TFHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!])$. Then, the challenger outputs $\mathsf{TFHE.Sim}(\mathsf{C}, \mathsf{C}(m_1, \ldots, m_k), [\![y]\!], \{\mathsf{sk}_i\}_{i\in\mathcal{S}})$ to $\mathcal{A}$.
3. The adversary may repeat step 2 $\mathsf{poly}(\lambda)$ many times.
4. At the end of the experiment, $\mathcal{A}$ outputs a distinguishing bit b.

Imported Theorem 2.2 ([BGG+18, BJMS20]). *Assuming circular-secure LWE, there exists a TFHE scheme for the n-out-of-n threshold access structure.*[4]

[4]Note that we require circular-secure LWE and not standard LWE only because we require "strong" compactness where the size of the ciphertext and partial decryption doesn't grow with the circuit depth.

2.6 Threshold Additive Homomorphic Encryption

We define a threshold additive homomorphic encryption scheme (TAHE) with distributed setup by following the definition of threshold fully homomorphic encryption above but restricting it to only additive homomorphism. A TAHE scheme with distributed setup consists of the following PPT algorithms $\mathsf{TAHE} = (\mathsf{TAHE.DistSetup}, \mathsf{TAHE.Enc}, \mathsf{TAHE.Eval}, \mathsf{TAHE.PartialDec}, \mathsf{TAHE.Combine})$ with the only difference from TFHE being that in the algorithm $\mathsf{TAHE.Eval}$, the circuit C is only allowed to be linear. That is, by a linear circuit, we mean that $\mathsf{C}(x_1, \ldots, x_k) = (\Sigma_{i=1}^{k} a_i \cdot x_i + b)$ for some values $(a_1, \ldots, a_k, b)$ hardwired into the circuit.

Instantiations. We note that several popular additively homomorphic encryption schemes in literature such as ElGamal encryption [Gam84] (based on the Decisional Diffie Hellman assumption) and Paillier encryption [Pai99] (based on the Decisional Composite Residuosity assumption). can in fact be easily converted into a TAHE scheme with the security properties we require. We refer the reader to the work of Hazay and Venkitasubramaniam [HV17] for more details.

Re-randomization. We implicitly assume that each homomorphic evaluation on a set of ciphertexts is concluded with a refresh operation, where the party adds the resulting ciphertext with an independently generated ciphertext that encrypts zero. This is required in order to ensure that the randomness of the final ciphertext is independent of the randomness of the original set of ciphertexts. For the schemes we mentioned above, a homomorphically evaluated ciphertext is statistically identical to a fresh ciphertext. That is, for any $\{(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{TAHE.DistSetup}(1^\lambda, j)\}_{j \in [n]}$, any linear circuit C with input $m_1, \ldots, m_k$, any $[\![m_i]\!] \leftarrow \mathsf{TAHE.Enc}(\mathsf{pk}, m_i)$, it holds that

$$\mathsf{TAHE.Eval}(\mathsf{pk}, \mathsf{C}, [\![m_1]\!], \ldots, [\![m_k]\!]) \equiv \mathsf{TAHE.Enc}(\mathsf{pk}, \mathsf{C}(m_1, \ldots, m_k)).$$

2.7 Linear Algebra

In the security proofs of our protocols, we will make use of a few lemmas about polynomials stated below. The proofs are deferred to Appendix A.

Imported Lemma 2.3 (Lemma 2 from [GS19a])**.** *Let $\mathbb{F}$ be a finite field of order $q = \Omega(2^\lambda)$. Let polynomial $p(x) \in \mathbb{F}[x]$ be an arbitrary but fixed non-zero polynomial of degree at most d_p and let $R(x) \in \mathbb{F}[x]$ be a uniformly random polynomial of degree d_R. Then*

$$\mathsf{Pr}[\gcd(p(x), R(x)) \neq 1] \leq \mathsf{negl}(\lambda).$$

Lemma 2.4. *Let $\mathbb{F}$ be a finite field of order $q = \Omega(2^\lambda)$. Fix any $n = O(\mathsf{poly}(\lambda))$. For all polynomials $p_1(x), ..., p_n(x) \in \mathbb{F}[x]$ such that $\gcd(p_1, ..., p_n) = 1$, for all $1 \leq i < n$,*

$$\Pr_{r_j}[\gcd(p'_1 + ... + p'_i, p'_{i+1}, ..., p'_n) \neq 1] \leq \mathsf{negl}(\lambda)$$

where for all $j \in [n]$, $p'_j(x) := p(x) \cdot (x - r_j)$ and $r_j \xleftarrow{\$} \mathbb{F}$.

Lemma 2.5. *Let $\mathbb{F}$ be a finite field of prime order q. Fix any $n = O(\mathsf{poly}(\lambda))$. For all polynomials $p_1(x), \ldots, p_n(x) \in \mathbb{F}[x]$ each of degree α with $\gcd(p_1, \ldots, p_n) = 1$, let $R_1(x), \ldots, R_n(x) \in \mathbb{F}[x]$ be random polynomials with degree $\beta \geq \alpha$. Specifically, $R_1(x) = \sum_{j=0}^{\beta} r_{1,j} x^j, \ldots, R_n(x) = \sum_{j=0}^{\beta} r_{n,j} x^j$ where $r_{i,j} \xleftarrow{\$} \mathbb{F}$ are sampled independently and uniformly at random. Let $U(x) = \sum_{i=1}^{n} (p_i(x) \cdot R_i(x)) = \sum_{j=0}^{\alpha+\beta} u_j x^j$, then u_j's are distributed uniformly and independently over $\mathbb{F}$.*

3 Technical Overview

We now give an overview of the techniques used in our work. We denote P_1 as the designated party that can communicate with all the other parties.

3.1 TFHE-Based Protocol for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$

In Section 5.1 we construct a protocol for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ from TFHE. Our starting point is the two-party protocol of [GS19a]. Recall that there are two parties Alice and Bob with sets $S_A = \{a_1, \ldots, a_m\}$ and $S_B = \{b_1, \ldots, b_m\}$ respectively. These sets define two polynomials $\mathsf{p}_A(\mathrm{x}) := \prod_{i=1}^{m}(\mathrm{x} - a_i)$ and $\mathsf{p}_B(\mathrm{x}) := \prod_{i=1}^{m}(\mathrm{x} - b_i)$. Let $I := S_A \cap S_B$ be the intersection. A key observation in [MTZ03, GS19a] is that $\mathsf{p}(\mathrm{x}) := \frac{\mathsf{p}_B(\mathrm{x})}{\mathsf{p}_A(\mathrm{x})} = \frac{\mathsf{p}_{B\setminus I}(\mathrm{x})}{\mathsf{p}_{A\setminus I}(\mathrm{x})}$. Both the numerator and denominator of p have degree $m - |I|$. If $m - |I| = |S_A \setminus I| \leq T$, then $\mathsf{p}(\mathrm{x})$ has degree at most $2T$ and can be recovered from $2T + 1$ evaluations by rational function interpolation.[5] Given $\mathsf{p}(\mathrm{x})$, the elements in $S_A \setminus I$ are simply the roots of the polynomial in the denominator.

Two-party protocol. At a high level, the two-party protocol [GS19a] works as follows. First, Alice and Bob evaluate their own polynomials on $2T+1$ publicly known distinct points $\{\alpha_1, \ldots, \alpha_{2T+1}\}$ to obtain $\{\mathsf{p}_A(\alpha_1), \ldots, \mathsf{p}_A(\alpha_{2T+1})\}$ and $\{\mathsf{p}_B(\alpha_1), \ldots, \mathsf{p}_B(\alpha_{2T+1})\}$, respectively. Then, Alice generates a public-secret key pair for FHE and sends Bob the FHE public key, encrypted evaluations $\{[\![\mathsf{p}_A(\alpha_1)]\!], \ldots, [\![\mathsf{p}_A(\alpha_{2T+1})]\!]\}$, a uniformly random z and encrypted evaluation $[\![\mathsf{p}_A(z)]\!]$. Bob can homomorphically interpolate the rational function $[\![\mathsf{p}(\mathrm{x})]\!]$ from $\{[\![\mathsf{p}_A(\alpha_1)]\!], \ldots, [\![\mathsf{p}_A(\alpha_{2T+1})]\!]\}$ and $\{\mathsf{p}_B(\alpha_1), \ldots, \mathsf{p}_B(\alpha_{2T+1})\}$, and then homomorphically compute $[\![\mathsf{p}(z)]\!]$. Bob can also compute $\mathsf{p}_B(z)$ and homomorphically compute $\frac{\mathsf{p}_B(z)}{[\![\mathsf{p}_A(z)]\!]}$. We know that $\mathsf{p}(z) = \frac{\mathsf{p}_B(z)}{\mathsf{p}_A(z)}$ if and only if the degree of $\mathsf{p}(\mathrm{x})$ is $\leq 2T$. Therefore Bob homomorphically computes an encryption of the predicate $[\![b]\!] := \left([\![\mathsf{p}(z)]\!] \stackrel{?}{=} \frac{\mathsf{p}_B(z)}{[\![\mathsf{p}_A(z)]\!]}\right)$ and sends the encryption $[\![b]\!]$ back to Alice. Finally Alice decrypts and learns b.

Multi-party protocol. For n parties, a natural idea is to consider

$$\mathsf{p}(\mathrm{x}) := \frac{\mathsf{p}_2(\mathrm{x}) + \cdots + \mathsf{p}_n(\mathrm{x})}{\mathsf{p}_1(\mathrm{x})} = \frac{\mathsf{p}_{2\setminus I}(\mathrm{x}) + \cdots + \mathsf{p}_{n\setminus I}(\mathrm{x})}{\mathsf{p}_{1\setminus I}(\mathrm{x})}, \tag{1}$$

where $\mathsf{p}_i(\mathrm{x})$ encodes the set $S_i = \{a_1^i, \ldots, a_m^i\}$ as $\mathsf{p}_i(\mathrm{x}) := \prod_{j=1}^{m}(\mathrm{x} - a_j^i)$. The n parties first jointly generate the TFHE keys. Each party P_i sends encrypted evaluations $\{[\![\mathsf{p}_i(\alpha_1)]\!], \ldots, [\![\mathsf{p}_i(\alpha_{2T+1})]\!], [\![\mathsf{p}_i(z)]\!]\}$ to P_1. Now P_1 can interpolate $[\![\mathsf{p}(\mathrm{x})]\!]$ from $2T + 1$ evaluations and compute an encryption $[\![b]\!] := \left([\![\mathsf{p}(z)]\!] \stackrel{?}{=} \frac{[\![\mathsf{p}_2(z)]\!] + \cdots + [\![\mathsf{p}_n(z)]\!]}{\mathsf{p}_1(z)}\right)$. Finally the parties jointly decrypt $[\![b]\!]$.

Unexpected degree reduction. This seemingly correct protocol has a subtle issue.[6] Intuitively, we want to argue that $\mathsf{p}(\mathrm{x})$ in Equation 1 has degree $\leq 2T$ if and only if $|S_1 \setminus I| \leq T$. However, this is

[5] A rational function is a fraction of two polynomials. We refer to Minskey et al. [MTZ03] for details on rational function interpolation over a field. Also, we note that monic polynomials can be interpolated with $2T$ evaluation but we use $2T + 1$ for consistency with our other protocols.

[6] In fact, this subtle issue was initially overlooked by [GS19b] in their recent update of the multi-party protocol. It has subsequently been fixed after we notified them.

not true because elements not in the intersection might be accidentally canceled out, which results in a lower degree than the intersection carnality would imply. As a concrete example, consider three sets with distinct elements $S_1 = \{a\}$, $S_2 = \{b\}$, $S_3 = \{c\}$, where $b+c = 2\cdot a$. The intersection $I = \emptyset$. Ideally we hope the rational polynomial $\mathsf{p}(\mathrm{x})$ has degree 1 in both the numerator and denominator because $|S_1 \setminus I| = 1$. However,

$$\mathsf{p}(\mathrm{x}) = \frac{(\mathrm{x} - b) + (x - c)}{x - a} = \frac{2x - (b + c)}{x - a} = \frac{2x - 2a}{x - a} = 2.$$

Randomness to the rescue. On first thought, this approach seems fundamentally flawed as additional roots can always be created if we add polynomials in the numerator. To solve this problem, we add a random multiplicative term $(\mathrm{x} - r_i)$ to each polynomial p_i and set a new polynomial $\mathsf{p}'_i(\mathrm{x}) := \mathsf{p}_i(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ for a random r_i chosen by party P_i. Now, consider the rational polynomial

$$\mathsf{p}'(\mathrm{x}) := \frac{\mathsf{p}'_2(\mathrm{x}) + \cdots + \mathsf{p}'_n(\mathrm{x})}{\mathsf{p}'_1(\mathrm{x})} = \frac{\mathsf{p}'_{2\setminus I}(\mathrm{x}) + \cdots + \mathsf{p}'_{n\setminus I}(\mathrm{x})}{\mathsf{p}'_{1\setminus I}(\mathrm{x})}.$$

At a high level, the terms $(\mathrm{x} - r_i)$ will randomize the roots of the numerator sufficiently to ensure that these roots are unlikely to coincide with the roots of the denominator.

3.2 TFHE-Based Protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$

In Section 5.2 we present an TFHE-based protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$. In summary, party P_1 tries to homomorphically interpolate

$$\widetilde{\mathsf{p}}_i(\mathrm{x}) = \frac{\mathsf{p}_i(\mathrm{x})}{\mathsf{p}_1(\mathrm{x})} = \frac{\mathsf{p}_{i\setminus 1}(\mathrm{x})}{\mathsf{p}_{1\setminus i}(\mathrm{x})}$$

from $(2T + 1)$ evaluations and computes encrypted $D_{1,i} = S_1 \setminus S_i$ as well as $D_{i,1} = S_i \setminus S_1$ for every other party P_i. Note that if $|\left(\bigcup_{i=1}^m S_i\right) \setminus I| \le T$, then $|S_i \setminus I| \le T$ for all i and the degree of each $\widetilde{\mathsf{p}}_i(\mathrm{x})$ is at most $2T$, hence P_1 can interpolate it using $(2T + 1)$ evaluations. Observe that $\left(\bigcup_{i=1}^m S_i\right) \setminus I = \bigcup_{i=2}^m (D_{1,i} \cup D_{i,1})$, because each element $a \in \left(\bigcup_{i=1}^m S_i\right) \setminus I$ must be one of the two cases: (1) $a \in S_1$ and $a \notin S_i$ for some i (i.e., $a \in D_{1,i}$), or (2) $a \notin S_1$ and $a \in S_i$ for some i (i.e., $a \in D_{i,1}$). Therefore, party P_1 can homomorphically compute an encryption of $\left(\bigcup_{i=1}^m S_i\right) \setminus I$ and an encryption of the predicate $b = \left(|\left(\bigcup_{i=1}^m S_i\right) \setminus I| \stackrel{?}{\le} T\right)$. Finally, as before, the n parties jointly decrypt $[\![b]\!]$ to learn the output.

3.3 TAHE-Based Protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$

Section 6 presents our protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ based on TAHE. This protocol reduces the communication complexity for two-party from $\widetilde{O}(T^2)$ to $\widetilde{O}(T)$ as well as generalizes it to multi-party with communication $\widetilde{O}(Tn)$.

Two-party protocol. For two parties Alice and Bob with private sets S_A and S_B, if we encode their elements into two polynomials $\mathsf{p}_A(\mathrm{x}) = \sum_{i=1}^m \mathrm{x}^{a_i}$ and $\mathsf{p}_B(\mathrm{x}) = \sum_{i=1}^m \mathrm{x}^{b_i}$, then the number of monomials in the polynomial $\mathsf{p}(\mathrm{x}) := \mathsf{p}_A(\mathrm{x}) - \mathsf{p}_B(\mathrm{x})$ is exactly $|(S_A \setminus S_B) \cup (S_B \setminus S_A)|$. Now the

problem of cardinality testing (i.e., determining if $|(S_A \setminus S_B) \cup (S_B \setminus S_A)| \leq 2T$) has be reduced to determining whether the number of monomials in $\mathsf{p}(\mathrm{x})$ is $\leq 2T$. Using the polynomial sparsity test of Grigorescu et al. [GJR10], we can further reduce the problem to determining whether the Hankel matrix below is singular or not:

$$H = \begin{bmatrix} \mathsf{p}(u^0) & \mathsf{p}(u^1) & \dots & \mathsf{p}(u^{2T}) \\ \mathsf{p}(u^1) & \mathsf{p}(u^2) & \dots & \mathsf{p}(u^{2T+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{p}(u^{2T}) & \mathsf{p}(u^{2T+1}) & \dots & \mathsf{p}(u^{4T}) \end{bmatrix},$$

where u is chosen uniformly at random. In the two-party protocol, Alice generates a public-secret key pair for AHE and sends Bob the public key, a uniformly random u along with encrypted Hankel matrix for p_A. Then Bob can homomorphically compute encrypted Hankel matrix for p. Now Alice holds the secret key and Bob holds an encryption of matrix H. They need to jointly perform a secure matrix singularity testing to determine if the matrix is singular, which can be done using the protocol of Kiltz et al. [KMWF07] with communication $\widetilde{O}(T^2)$.

Our approach. Our key observation is that the protocol of Kiltz et al. [KMWF07] can be used to perform singularity testing for *arbitrary* matrices, while we are only interested in testing the singularity of Hankel matrices. Since a Hankel matrix only has linear (in its dimension) number of distinct entries, there is a more efficient way to test its singularity. In particular, the work of Brent et al. [BGY80] demonstrates an elegant connection between the problem of testing singularity of a Hankel matrix and the so-called "half-GCD" problem, which can be solved in quasi-linear time. Thus, testing singularity of the Hankel matrix H only takes $\widetilde{O}(T)$ computation. In our scenario, we can first let Alice and Bob learn an additive share of H, and then engage in a two-party computation (using AHE or Yao's garbled circuits) to jointly test if H is singular or not. The important point to note here is that both communication and computation are only quasi-linear in the dimension of H. This is already an improvement over the quadratic cost of protocol in [KMWF07] and solves the open problem posed by Ghosh and Simkin [GS19a].

Multi-party protocol. In designing a multi-party protocol, our strategy is to first find a polynomial where the number of monomials equals the size of the set difference $|(\bigcup_{i=1}^{m} S_i) \setminus I|$. Furthermore, the polynomial should only involve linear operations among the parties, which allows the parties to obtain additive secret shares of the Hankel matrix for the polynomial. Then, the parties perform an MPC protocol to test singularity of the Hankel matrix.

3.4 Computing Set Intersection

In Section 7 we present a single construction that computes the concrete set intersection for both $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ after the cardinality testing.

Two-party protocol. For two parties Alice and Bob, we use the first encoding method to encode the elements into two polynomials $\mathsf{p}_A(\mathrm{x}) = \prod_{i=1}^{m}(\mathrm{x} - a_i)$ and $\mathsf{p}_B(\mathrm{x}) = \prod_{i=1}^{m}(\mathrm{x} - b_i)$. After the cardinality testing, we already know that the rational polynomial $\mathsf{p}(\mathrm{x}) := \frac{\mathsf{p}_B(\mathrm{x})}{\mathsf{p}_A(\mathrm{x})} = \frac{\mathsf{p}_{B\setminus I}(\mathrm{x})}{\mathsf{p}_{A\setminus I}(\mathrm{x})}$ has degree at most $2T$. If Alice learns the evaluation of $\mathsf{p}_B(\cdot)$ on $2T+1$ distinct points $\{\alpha_1, \dots, \alpha_{2T+1}\}$,

then she can evaluate p_A on those points by herself and compute $\{\mathsf{p}(\alpha_1), \ldots, \mathsf{p}(\alpha_{2T+1})\}$. Using these evaluations of $\mathsf{p}(\cdot)$, Alice can recover $\mathsf{p}(\mathrm{x})$ by rational polynomial interpolation, and then learn the set difference $S_A \setminus I$ from the denominator of $\mathsf{p}(\mathrm{x})$. However, $\mathsf{p}(\mathrm{x})$ also allows Alice to learn $S_B \setminus I$, which breaks security. Instead of letting Alice learn the evaluations of $\mathsf{p}_B(\cdot)$, the two-party protocol of [GS19a] enables Alice to learn the evaluations of a "noisy" polynomial $\mathsf{V}(\mathrm{x}) := \mathsf{p}_A(\mathrm{x}) \cdot \mathsf{R}_1(\mathrm{x}) + \mathsf{p}_B(\mathrm{x}) \cdot \mathsf{R}_2(\mathrm{x})$, where R_1 and R_2 are uniformly random polynomials of degree T. Note that

$$\mathsf{p}'(\mathrm{x}) := \frac{\mathsf{V}(\mathrm{x})}{\mathsf{p}_A(\mathrm{x})} = \frac{\mathsf{p}_{A\setminus I}(\mathrm{x}) \cdot \mathsf{R}_1(\mathrm{x}) + \mathsf{p}_{B\setminus I}(\mathrm{x}) \cdot \mathsf{R}_2(\mathrm{x})}{\mathsf{p}_{A\setminus I}(\mathrm{x})}$$

has degree at most $3T$. Given $3T+1$ evaluations of $\mathsf{V}(\cdot)$, Alice can interpolate $\mathsf{p}'(\mathrm{x})$ and figure out the denominator, but now the numerator is sufficiently random and does not leak any other information about S_B.

Multi-party protocol. For n parties, we first encode each set $S_i = \{a_1^i, \ldots, a_m^i\}$ as a polynomial $\mathsf{p}_i(\mathrm{x}) := \prod_{j=1}^m (\mathrm{x} - a_j^i)$, and then define

$$\begin{aligned}\mathsf{V}(\mathrm{x}) :=& \mathsf{p}_1(\mathrm{x}) \cdot \mathsf{R}_1(\mathrm{x}) + \cdots + \mathsf{p}_n(\mathrm{x}) \cdot \mathsf{R}_n(\mathrm{x}) \\ :=& \mathsf{p}_1(\mathrm{x}) \cdot (\mathsf{R}_{1,1}(\mathrm{x}) + \cdots + \mathsf{R}_{n,1}(\mathrm{x})) + \cdots + \mathsf{p}_n(\mathrm{x}) \cdot (\mathsf{R}_{1,n}(\mathrm{x}) + \cdots + \mathsf{R}_{n,n}(\mathrm{x}))\,,\end{aligned}$$

where $(\mathsf{R}_{i,1}, \ldots, \mathsf{R}_{i,n})$ are random polynomials of degree T generated by party P_i. Different from the two-party protocol, it is crucial that each party P_i contributes a random term in every polynomial $\mathsf{R}_1, \ldots, \mathsf{R}_n$. For both functionalities $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$, if the protocol passes the cardinality testing, then

$$\mathsf{p}'(\mathrm{x}) := \frac{\mathsf{V}(\mathrm{x})}{\mathsf{p}_1(\mathrm{x})} = \frac{\mathsf{p}_{1\setminus I}(\mathrm{x}) \cdot \mathsf{R}_1(\mathrm{x}) + \cdots + \mathsf{p}_{n\setminus I}(\mathrm{x}) \cdot \mathsf{R}_n(\mathrm{x})}{\mathsf{p}_{1\setminus I}(\mathrm{x})}$$

has degree at most $3T$. If P_1 learns $3T+1$ evaluations of $\mathsf{V}(\cdot)$, then it can interpolate $\mathsf{p}'(\mathrm{x})$ and recover $S_1 \setminus I$ from the denominator while the numerator does not leak any other information. Since $\mathsf{V}(\cdot)$ can be broken down to linear operations among the parties, it can be securely evaluated by TAHE.

Communication blow-up. However, this protocol requires $O(n^2)$ communication complexity per evaluation, and the total communication complexity is $O(n^2T)$ for $(3T+1)$ evaluations. Observe that the bottleneck of the communication in this approach is that every party P_i needs to contribute n randomizing polynomials $(\mathsf{R}_{i,1}, \ldots, \mathsf{R}_{i,n})$. Through a careful analysis we demonstrate that it is sufficient for each party to only contribute two randomizing polynomials. The first is used to randomize their own polynomial while the second randomizes the polynomials from the other parties. Nevertheless, there is a subtle issue of unexpected degree reduction, similar to what we have seen in the TFHE-based protocol $\mathcal{F}_{\mathsf{CTest\text{-}int}}$. We follow the same approach as in the TFHE-based protocol by adding additional randomness in th polynomial, which reduces the communication complexity to $O(nT)$.

3.5 Lower Bounds

We briefly discuss the communication lower bound for multi-party threshold PSI. To prove lower bound in the point-to-point network, we perform a reduction from two-party threshold PSI (for

which [GS19a] showed a lower bound of $\Omega(T)$) to multi-party threshold PSI. We first prove that the total "communication complexity of any party" is $\Omega(T)$ which denotes the sum of all the bits exchanged by that party (both sent and received). As a corollary, the total communication complexity of any multi-party threshold PSI protocol is $\Omega(nT)$. We refer to Section 4 for more details about the reduction.

To prove a lower bound in the broadcast model, we rely on the communication lower bound of the multi-party set disjointness problem shown by Braverman and Oshman [BO15]. We reduce the problem of multi-party set disjointness to multi-party threshold PSI $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and prove a lower bound $\Omega(T \log n + n)$ for any multi-party threshold PSI protocol in the broadcast network. We refer to Appendix C for more details about the reduction.

4 Communication Lower Bound

In this section, we prove communication lower bounds for multi-party threshold PSI protocols in the point-to-point network model. Recall that we consider all parties to have sets of the same size m. We show that any secure protocol must have communication complexity at least $\Omega(n \cdot T)$ for both functionalities $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ and $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$.

4.1 Lower Bound for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$

Before proving the lower bound, we first prove another related theorem below.

Theorem 4.1. *For any multi-party threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ that is secure against a semi-honest adversary that can corrupt up to $(n-1)$ parties, for every party P_i, the communication complexity of P_i is $\Omega(T)$.*[7]

Proof. Suppose this is not true. That is, suppose there exists a secure multi-party threshold PSI protocol Π for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ in which for some party P_{i^*}, $\mathsf{CC}(P_{i^*}) = o(T)$ where $\mathsf{CC}(\cdot)$ denotes the communication complexity. We will now use this protocol Π as a subroutine to design a secure two-party threshold PSI protocol which has communication complexity $o(T)$.

Consider two parties Q_1 and Q_2 with input sets X_1 and X_2 (of same size m) who wish to run a secure two-party threshold PSI protocol for the following functionality: both parties learn the output if $|(X_1 \setminus X_2) \cup (X_2 \setminus X_1)| \leq 2 \cdot T$. We invoke the multi-party threshold PSI protocol Π with threshold T as follows: Q_1 emulates the role of party P_{i^*} with input set $S_{i^*} = X_1$ and Q_2 emulates the role of all the other $(n-1)$ parties with each of their input sets as X_2. From the definition of the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$, Q_1 learns the output at the end of the protocol if and only if $|X_1 \setminus I| \leq T$. Similarly, Q_2 learns the output at the end of the protocol if and only if $|X_2 \setminus I| \leq T$. Notice that since $|X_1| = |X_2|$ and $I = X_1 \cap X_2$, $|X_1 \setminus I| = |X_2 \setminus I|$. Thus, the parties learn the output if and only if $(|X_1 \setminus I|) + (|X_2 \setminus I|) \leq 2 \cdot T$, namely $|(X_1 \setminus X_2) \cup (X_2 \setminus X_1)| \leq 2 \cdot T$, which is the functionality of the two-party threshold PSI. Therefore, correctness is easy to observe. For security, notice that if Q_1 is corrupt, we can simulate it by considering only a corrupt P_{i^*} in the underlying protocol Π and if Q_2 is corrupt, we can simulate it by considering all parties except P_{i^*} to be corrupt in the underlying protocol Π.

[7] We define the communication complexity of a party P_i in any protocol execution as the complexity of all the communication that P_i is involved in. That is, the complexity of the messages both incoming to and outgoing from P_i.

Finally, notice that the communication complexity of the two-party protocol is exactly the same as $\mathsf{CC}(P_{i^*})$ in the multi-party protocol Π, which is $o(T)$. However, recall from the work of Ghosh and Simkin [GS19a] that any two-party threshold PSI for this functionality has communication complexity lower bound $\Omega(T)$ leading to a contradiction. Thus, the assumption that there exists a secure multi-party PSI protocol Π in which for some party P_{i^*}, $\mathsf{CC}(P_{i^*}) = o(T)$ is wrong and this completes the proof of the theorem. □

It is easy to observe that as a corollary of the above theorem, in a setting with only point-to-point channels (which also includes the star network), the overall communication complexity of the protocol must be at least n times the minimum communication complexity that each party is involved in, giving the lower bound of $\Omega(n \cdot T)$. Formally,

Corollary 4.2. *For any multi-party threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ that is secure against a semi-honest adversary that can corrupt up to $(n-1)$ parties, the communication complexity is $\Omega(n \cdot T)$.*

4.2 Lower Bound for $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$

The lower bound proof for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ is very similar to the one for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$. The only difference is that in the reduction, we invoke the two-party threshold PSI protocol where both parties learn the output if $|(X_1 \setminus X_2) \cup (X_2 \setminus X_1)| \leq T$ instead of $2 \cdot T$ as in the previous case. We elaborate on the proof for the sake of completeness.

Once again, before we prove the lower bound, we first prove another related theorem below.

Theorem 4.3. *For any multi-party threshold private intersection protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ that is secure against a semi-honest adversary that can corrupt up to $(n-1)$ parties, for every party P_i, the communication complexity of P_i is $\Omega(T)$.*

Proof. Suppose this is not true. That is, suppose there exists a secure multi-party threshold PSI protocol Π for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ in which for some party P_{i^*}, $\mathsf{CC}(P_{i^*}) = o(T)$. We will now use this protocol Π as a subroutine to design a secure two-party threshold PSI protocol which has communication complexity $o(T)$.

Consider two parties Q_1 and Q_2 with input sets X_1 and X_2 (of same size) who wish to run a secure two-party threshold PSI protocol for the following functionality: both parties learn the output if $|(X_1 \setminus X_2) \cup (X_2 \setminus X_1)| \leq T$. We invoke the multi-party threshold PSI protocol Π with threshold T as follows: Q_1 emulates the role of party P_{i^*} with input set $S_{i^*} = X_1$ and Q_2 emulates the role of all the other $(n-1)$ parties with each of their input sets as X_2. From the definition of the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$, the parties learn the output at the end of the protocol if and only if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$, namely $|(X_1 \setminus X_2) \cup (X_2 \setminus X_1)| \leq T$, which is the functionality of the two-party threshold PSI primitive. Thus, correctness is easy to observe. For security, notice that if Q_1 is corrupt, we can simulate it by considering only a corrupt P_{i^*} in the underlying protocol Π and if Q_2 is corrupt, we can simulate it by considering all parties except P_{i^*} to be corrupt in the underlying protocol Π.

Finally, notice that the communication complexity of the two-party protocol is exactly the same as $\mathsf{CC}(P_{i^*})$ in the multi-party protocol Π, which is $o(T)$. However, recall from the work of Ghosh and Simkin [GS19a] that any two-party threshold PSI for this functionality has communication complexity lower bound $\Omega(T)$ leading to a contradiction. Thus, the assumption that there exists a

secure multi-party threshold PSI protocol Π in which for some party P_{i^*}, $\mathsf{CC}(P_{i^*}) = o(T)$ is wrong and this completes the proof of the theorem. $\square$

Similarly as in the proof for $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$, we get the following corollary from the above theorem:

Corollary 4.4. *For any multi-party threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ that is secure against a semi-honest adversary that can corrupt up to $(n-1)$ parties, the communication complexity is $\Omega(n \cdot T)$ where n is the number of parties and T is the threshold parameter.*

5 TFHE-Based Private Intersection Cardinality Testing

In this section, we present two protocols for private intersection cardinality testing, one for functionalities $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ (described in Figure 3) and the other for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ (described in Figure 4). Both protocols are based on n-out-of-n threshold fully homomorphic encryption with distributed setup. The former functionality states that the intersection must be of size at least $(m - T)$ where m is the size of each set. The latter functionality requires the difference between the union of all the sets and the intersection be of size at most T. Due to the possibility of elements appearing in a strict subset of the sets, these two functionalities are not equivalent.

5.1 Protocol for Functionality $\mathcal{F}_{\mathsf{CTest\text{-}int}}$

In this protocol, we compute the cardinality predicate b where $b = 1$ if and only if $\forall i, |S_i \setminus I| \leq T$. The communication complexity of this protocol involves sending $O(nT)$ TFHE ciphertexts and performing a single decryption of the result. We briefly describe the approach below.

Each party P_i first encodes their set S_i as a polynomial $\mathsf{p}_i(\mathrm{x}) := \prod_{a \in S_i}(\mathrm{x} - a) \in \mathbb{F}[\mathrm{x}]$. Each of these polynomials are then randomized as $\mathsf{p}'_i(\mathrm{x}) := \mathsf{p}_i(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ where P_i uniformly samples $r_i \xleftarrow{\$} \mathbb{F}$. The central party also picks a random $z \xleftarrow{\$} \mathbb{F}$ which is sent to every other party. Each party P_i then computes $e_{i,j} := \mathsf{p}'_i(j)$ for $j \in [2T + 3]$ and $e'_i := \mathsf{p}'_i(z)$. P_i sends the ciphertexts $[\![e_{i,j}]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e_{i,j})$ and $[\![e'_i]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e'_i)$ to P_1. Party P_1 considers the rational polynomial

$$\mathsf{p}'(\mathrm{x}) = \frac{\mathsf{p}'_2(\mathrm{x}) + \cdots + \mathsf{p}'_n(\mathrm{x})}{\mathsf{p}'_1(\mathrm{x})}$$

and homomorphically computes $2T + 3$ encrypted evaluations

$$\left(j, \left[\!\!\left[\frac{e_{2,j} + \cdots + e_{n,j}}{e_{1,j}}\right]\!\!\right]\right)$$

for $j = [2T + 3]$. Using these encrypted evaluations, P_1 homomorphically computes an encrypted rational polynomial $[\![\mathsf{p}^*(\mathrm{x})]\!]$ using rational polynomial interpolation. Note that $\mathsf{p}^*(\mathrm{x}) = \mathsf{p}'(\mathrm{x})$ if $\mathsf{p}'(\mathrm{x})$ has degree at most $2T + 2$. Furthermore, P_1 can homomorphically compute an encryption of the predicate $b := \left(\mathsf{p}^*(z) \stackrel{?}{=} \frac{e'_2 + \cdots + e'_n}{e'_1}\right)$. Finally the parties jointly perform a threshold decryption of $[\![b]\!]$ and party P_1 learns the output which is sent to every other party. The full protocol is detailed in Figure 5.

Theorem 5.1. *Assuming threshold FHE with distributed setup, protocol $\Pi_{\mathsf{TFHE\text{-}CTest\text{-}int}}$ (Figure 5) securely realizes $\mathcal{F}_{\mathsf{CTest\text{-}int}}$ (Figure 3).*

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$. $\mathbb{F}$ is a finite field where $|\mathbb{F}| = \Omega(2^\lambda)$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}$ for all $j \in [m]$.

Output: Each party P_i receives $\mathsf{similar}$ if $|S_i \setminus I| \leq T$ and $\mathsf{different}$ otherwise where $I = \bigcap_{i=1}^n S_i$.

Protocol:

1. Each party P_i generates $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, i)$ and sends pk_i to P_1. Then P_1 sends $\mathsf{pk} = (\mathsf{pk}_1 \| \ldots \| \mathsf{pk}_n)$ to all the other parties.
2. P_1 picks a random value $z \in \mathbb{F}$ and sends it to all the other parties.
3. Each party P_i does the following:
   (a) Define the polynomial $\mathsf{p}_i(\mathrm{x}) := \prod_{a \in S_i}(\mathrm{x} - a)$ and randomize it by $\mathsf{p}'_i(\mathrm{x}) := \mathsf{p}_i(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ where $r_i \xleftarrow{\$} \mathbb{F}$.
   (b) Compute $e_{i,j} := \mathsf{p}'_i(j)$ for $j \in [2T+3]$ and $e'_i := \mathsf{p}'_i(z)$.
   (c) Send encrypted evaluations $[\![e_{i,j}]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e_{i,j})$ for all $j \in [2T+3]$ and $[\![e'_i]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e'_i)$ to P_1.
4. P_1 does the following:
   (a) Use the algorithm $\mathsf{TFHE.Eval}$ to homomorphically compute an encryption $[\![\mathsf{p}^*(\mathrm{x})]\!]$ by rational polynomial interpolation from encrypted evaluations $\left(j, \left[\!\!\left[\frac{e_{2,j}+\cdots+e_{n,j}}{e_{1,j}}\right]\!\!\right]\right)$ for $j \in [2T+3]$.
   (b) Homomorphically compute the encrypted predicate $[\![b]\!]$ where $b = 1$ if $\mathsf{p}^*(z) = \frac{e'_2+\cdots+e'_n}{e'_1}$ and 0 otherwise.
5. P_1 sends $[\![b]\!]$ to all parties who respond with $[\![b : \mathsf{sk}_i]\!] := \mathsf{TFHE.PartialDec}(\mathsf{sk}_i, [\![b]\!])$. P_1 broadcasts $b := \mathsf{TFHE.Combine}(\mathsf{pk}, \{[\![b : \mathsf{sk}_i]\!]\}_{i \in [n]})$ and all parties output $\mathsf{similar}$ if $b = 1$ and $\mathsf{different}$ otherwise.

Figure 5: Multi-party private intersection cardinality testing protocol $\Pi_{\mathsf{TFHE\text{-}CTest\text{-}int}}$ for $\mathcal{F}_{\mathsf{CTest\text{-}int}}$.

Proof. **Correctness.** We first prove the protocol is correct. By the correctness of the TFHE scheme, we only need to show that the computed predicate $b = 1$ if and only if $\forall i, |S_i \setminus I| \leq T$. First consider the case where the protocol should output $\mathsf{similar}$. Since

$$\mathsf{p}'(\mathrm{x}) = \frac{\mathsf{p}'_2(\mathrm{x}) + \cdots + \mathsf{p}'_n(\mathrm{x})}{\mathsf{p}'_1(\mathrm{x})} = \frac{\mathsf{p}_{2\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_2) + \cdots + \mathsf{p}_{n\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_n)}{\mathsf{p}_{1\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_1)},$$

the degree of each term $\mathsf{p}_{i\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ is at most $T+1$ and therefore the rational polynomial interpolation requires a total of $(2T+3)$ evaluation points. Therefore $\mathsf{p}^*(\mathrm{x}) = \mathsf{p}'(\mathrm{x})$ and $\mathsf{p}^*(z) = \mathsf{p}'(z) = \frac{e'_2+\cdots+e'_n}{e'_1}$. Thus $b = 1$ as required.

Now consider the case where the protocol should output $\mathsf{different}$, namely when $|I| < m - T$. Observe that $\gcd(\mathsf{p}_{1\setminus I}, \cdots, \mathsf{p}_{n\setminus I}) = 1$ by construction and therefore Lemma 2.4 states that

$$\gcd\left(\mathsf{p}'_{2\setminus I}(\mathrm{x}) + \cdots + \mathsf{p}'_{n\setminus I}(\mathrm{x}), \mathsf{p}'_{1\setminus I}(\mathrm{x})\right) = 1$$

except with negligible probability, where $\mathsf{p}'_{i\setminus I}(\mathrm{x}) := \mathsf{p}_{i\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_i)$.

Assuming $\gcd\left(\mathsf{p}'_{2\setminus I}(\mathrm{x}) + \cdots + \mathsf{p}'_{n\setminus I}(\mathrm{x}), \mathsf{p}'_{1\setminus I}(\mathrm{x})\right) = 1$, it then follows that the degree of the rational polynomial $\mathsf{p}'(\mathrm{x})$ is the degree of $\mathsf{p}'_{2\setminus I}(\mathrm{x}) + \cdots + \mathsf{p}'_{n\setminus I}(\mathrm{x})$ plus the degree of $\mathsf{p}'_{1\setminus I}(\mathrm{x})$. The

former must have a leading term with degree $(m - |I| + 1) > (T + 1)$. Similarly, the latter also has degree $(m - |I| + 1) > T + 1$. Hence the degree of $\mathsf{p}'(\mathrm{x})$ is at least $2T + 4$. The probability of $b = 1$ is $\Pr_z[\mathsf{p}'(z) = \mathsf{p}^*(z)]$ where $\mathsf{p}^*(\mathrm{x})$ is the polynomial interpolated by P_1 using $(2T + 3)$ evaluations. However, since the degree of $\mathsf{p}'(\mathrm{x})$ is at least $2T + 4$, $\Pr_z[\mathsf{p}'(z) = \mathsf{p}^*(z)] \leq \mathsf{negl}(\lambda)$.

Communication Cost. Each party sends $(2T+4)$ TFHE encryptions and one partial decryption to P_1 where each plaintext is a field element. P_1 sends one ciphertext to every other party. The size of each encryption and each partial decryption is $\mathsf{poly}(\lambda)$. Thus, the overall communication complexity is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ in a star network and the protocols runs in $O(1)$ rounds.

Security. Consider an environment $\mathcal{Z}$ who corrupts a set $\mathcal{S}^*$ of n^* parties where $n^* < n$. The simulator Sim has output $w \in \{\mathsf{similar}, \mathsf{different}\}$ from the ideal functionality. Sim sets a bit $b^* = 1$ if $w = \mathsf{similar}$ and $b^* = 0$ otherwise. Also, for each corrupt party P_i, Sim has as input the tuple (S_i, r_i) indicating the party's input and randomness for the protocol. The strategy of the simulator Sim for our protocol is described below.

1. Sim runs the distributed key generation algorithm $\mathsf{TFHE.DistSetup}(1^\lambda, i)$ of the TFHE scheme honestly on behalf of each honest party P_i as in the real world. Note that Sim also knows $(\{\mathsf{sk}_i\}_{i \in S^*})$ as it knows the randomness for the corrupt parties.

2. In Steps 2-4 of the protocol, Sim plays the role of the honest parties exactly as in the real world except that on behalf of every honest party P_i, whenever P_i has to send any ciphertext, compute $[\![0]\!] = \mathsf{TFHE.Enc}(0)$ using fresh randomness.

3. In Step 5, on behalf of each honest party P_i, instead of sending the value $[\![b : \mathsf{sk}_i]\!]$ by running the honest $\mathsf{TFHE.PartialDec}$ algorithm as in the real world, Sim computes the partial decryptions by running the simulator $\mathsf{TFHE.Sim}$ as follows: $\{[\![b : \mathsf{Sim}_i]\!]\}_{i \in [n] \setminus \mathcal{S}^*} \leftarrow \mathsf{TFHE.Sim}(\mathsf{C}, b^*, [\![b]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}^*})$ where the circuit C denotes the whole computation done by P_1 in the real world to evaluate bit b. On behalf of the honest party P_i the simulator sends $[\![b : \mathsf{Sim}_i]\!]$. This corresponds to the ideal world.

We now show that the above simulation strategy is successful against all environments $\mathcal{Z}$ that corrupt parties in a semi-honest manner. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hybrid_0 corresponds to the real world and the last hybrid Hybrid_2 corresponds to the ideal world.

- Hybrid_0 **- Real World:** In this hybrid, consider a simulator SimHyb that plays the role of the honest parties as in the real world.
- Hybrid_1 **- Simulate Partial Decryptions:** - In this hybrid, in Step 5, SimHyb simulates the partial decryptions generated by the honest parties as done in the ideal world. That is, the simulator calls $\{[\![b : \mathsf{Sim}_i]\!]\}_{i \in [n] \setminus \mathcal{S}} \leftarrow \mathsf{TFHE.Sim}(\mathsf{C}, b^*, [\![b]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}})$. On behalf of the honest party P_i the simulator sends $[\![b : \mathsf{Sim}_i]\!]$ instead of $[\![b : \mathsf{sk}_i]\!]$.
- Hybrid_2 **- Switch Encryptions:** In this hybrid, SimHyb now computes every ciphertext generated on behalf of any honest party as encryptions of 0 as done by Sim in the ideal world. This hybrid corresponds to the ideal world.

We now show that every pair of consecutive hybrids is computationally indistinguishable.

Lemma 5.2. *Assuming the simulation security of the threshold fully homomorphic encryption scheme,* Hybrid_0 *is computationally indistinguishable from* Hybrid_1.

Proof. The only difference between the two hybrids is that in Hybrid_0, the simulator SimHyb generates the partial decryptions of the TFHE scheme on behalf of the honest parties as in the real world while in Hybrid_1, they are simulated by running the simulator $\mathsf{TFHE.Sim}$. We now show that if there exists an environment $\mathcal{Z}$ that can distinguish between these two hybrids with some non-negligible probability ϵ, we will come up with a reduction $\mathcal{A}$ that can break the simulation security of the TFHE scheme.

$\mathcal{A}$ interacts with a challenger $\mathcal{C}$ in the simulation security game for TFHE and with the environment $\mathcal{Z}$ in the game between Hybrid_0 and Hybrid_1. $\mathcal{A}$ corrupts the same set of parties as done by $\mathcal{Z}$ in its game with $\mathcal{C}$. Further, $\mathcal{A}$ forwards the public key-secret key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ for the corrupt parties it receives from $\mathcal{Z}$ to the challenger and the public keys pk_i for the honest parties from $\mathcal{C}$ to $\mathcal{Z}$. $\mathcal{A}$ also forwards to $\mathcal{C}$ the set of messages to be encrypted along with the randomness for the ones encrypted by the adversary, received from $\mathcal{Z}$. Similarly, it forwards the ciphertexts received from $\mathcal{C}$ to $\mathcal{Z}$. Finally, $\mathcal{A}$ sends the circuit C that denotes the whole computation done by P_1 in the real world to evaluate bit b and receives a set of partial decryptions on behalf of each honest party which it forwards to $\mathcal{A}$. It continues interacting with $\mathcal{Z}$ as in Hybrid_0 in the rest of its interaction. It is easy to see that if $\mathcal{C}$ sent honestly generated partial decryptions, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_0 and if the partial decryptions were simulated, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_1. Thus, if $\mathcal{Z}$ can distinguish between the two hybrids with non-negligible probability ϵ, $\mathcal{A}$ can break the simulation security of the TFHE scheme with the same probability ϵ which is a contradiction. □

Lemma 5.3. *Assuming the semantic security of the threshold fully homomorphic encryption scheme,* Hybrid_1 *is computationally indistinguishable from* Hybrid_2.

Proof. The only difference between the two hybrids is that in Hybrid_1, the simulator SimHyb generates the encryptions of the TFHE scheme on behalf of the honest parties as in the real world while in Hybrid_2, they are generated as encryptions of 0. We now show that if there exists an adversarial environment $\mathcal{Z}$ that can distinguish between these two hybrids with some non-negligible probability ϵ, we will come up with a reduction $\mathcal{A}$ that can break the semantic security of the TFHE scheme.

$\mathcal{A}$ interacts with a challenger $\mathcal{C}$ in the semantic security game for TFHE and with the environment $\mathcal{Z}$ in the game between Hybrid_1 and Hybrid_2. $\mathcal{A}$ corrupts the same set of parties as done by $\mathcal{Z}$ in its game with $\mathcal{C}$. Further, $\mathcal{A}$ forwards the public key-secret key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ for the corrupt parties it receives from $\mathcal{Z}$ to the challenger and the public keys pk_i for the honest parties from $\mathcal{C}$ to $\mathcal{Z}$. $\mathcal{A}$ also forwards the pair of 0 and the set of honestly generated plaintexts to be encrypted, to the challenger and receives back a ciphertext for each of them which it uses in its interaction with $\mathcal{Z}$. It continues interacting with $\mathcal{Z}$ as in Hybrid_1 in the rest of its interaction. It is easy to see that if $\mathcal{C}$ sent honestly generated ciphertexts, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_1 and if the ciphertexts were generated as encryptions of 0, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_2. Thus, if $\mathcal{Z}$ can distinguish between the two hybrids with non-negligible probability ϵ, $\mathcal{A}$ can break the semantic security of the TFHE scheme with the same probability ϵ which is a contradiction. □

□

5.2 Protocol for Functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$

This protocol will compute the cardinality predicate b where $b = 1$ if and only if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. The core idea behind the protocol is that P_1 (the star of the network) and P_i first run a protocol to compute an encryption (via TFHE) of their set differences $D_{1,i} = S_1 \setminus S_i$ and $D_{i,1} = S_i \setminus S_1$ with $O(T)$ communication complexity if $|S_1 \setminus S_i| \leq T$. Before we describe how this is achieved, notice that at this point, the protocol enables P_1 to reconstruct an encryption of $(\bigcup_{i=1}^{n} S_i) \setminus I = \bigcup_{i \in [n] \setminus \{1\}} (D^*_{1,i} \cup D^*_{i,1})$ and a predicate b where $b = 1$ if and only if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. P_1 can then send this encryption to all parties to run threshold decryption.

We now describe in more detail how the encryption of $D_{1,i}$ and $D_{i,1}$ are computed. The idea follows from the two-party protocol of Ghosh and Simkin [GS19a]. Each party P_i encodes their set S_i as $\mathsf{p}_i(\mathrm{x}) := \Pi_{a \in S_i}(\mathrm{x} - a) \in \mathbb{F}[x]$. P_i then computes $e_{i,j} := \mathsf{p}_i(j)$ for $j \in [2T+1]$ and $e'_i := \mathsf{p}_i(z)$ on a special random point $z \in \mathbb{F}$ (picked uniformly at random by P_1). Party P_i encrypts these values as $[\![e_{i,j}]\!], [\![e'_i]\!]$ and sends them to P_1. Party P_1 considers the rational polynomial

$$\widetilde{\mathsf{p}}_i(\mathrm{x}) = \frac{\mathsf{p}_i(\mathrm{x})}{\mathsf{p}_1(\mathrm{x})} = \frac{\mathsf{p}_{i\setminus 1}(\mathrm{x})}{\mathsf{p}_{1\setminus i}(\mathrm{x})}$$

and homomorphically computes $2T+1$ encrypted evaluations $\left(j, \left[\!\left[\frac{e_{i,j}}{e_{1,j}}\right]\!\right]\right)$ for $j = [2T+1]$. Using these encrypted evaluations, P_1 homomorphically computes an encrypted rational polynomial $[\![\widetilde{\mathsf{p}}^*_i(\mathrm{x})]\!]$ using rational polynomial interpolation. P_1 then homomorphically reconstructs the roots of $\mathsf{p}_{i\setminus 1}(\mathrm{x})$ and $\mathsf{p}_{1\setminus i}(\mathrm{x})$ from $\widetilde{\mathsf{p}}^*_i$ to obtain $\left[\!\left[D^*_{i,1}\right]\!\right], \left[\!\left[D^*_{1,i}\right]\!\right]$. Note that $\widetilde{\mathsf{p}}^*_i(\mathrm{x}) = \widetilde{\mathsf{p}}_i(\mathrm{x})$ if $\widetilde{\mathsf{p}}_i(\mathrm{x})$ has degree at most $2T$, in which case $D^*_{i,1} = D_{i,1}$ and $D^*_{1,i} = D_{1,i}$.

In the final protocol, P_1 homomorphically computes encrypted predicates b_i where $b_i = 1$ iff $\widetilde{\mathsf{p}}^*_i(z) = \frac{e'_i}{e'_1}$ for each $i \in [n] \setminus \{1\}$ and encrypted predicate b' where $b' = 1$ iff $\left|\bigcup_{i \in [n]\setminus\{1\}} (D^*_{1,i} \cup D^*_{i,1})\right| \leq T$. The output predicate b is homomorpically computed as $[\![b]\!] = \left[\!\left[b' \cdot \prod_{i \in [n]\setminus\{1\}} b_i\right]\!\right]$ and jointly decrypted by all the parties. The protocol is formally described in Figure 6.

Theorem 5.4. *Assuming threshold FHE with distributed setup, protocol* $\Pi_{\mathsf{TFHE\text{-}CTest\text{-}diff}}$ *(Figure 6) securely realizes* $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ *(Figure 4).*

Proof. **Correctness.** We first prove the protocol is correct. By the correctness of the TFHE scheme, we only need to show that the computed predicate $b = 1$ if and only if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. First consider the case where the protocol should output similar. Since

$$\widetilde{\mathsf{p}}_i(\mathrm{x}) = \frac{\mathsf{p}_i(\mathrm{x})}{\mathsf{p}_1(\mathrm{x})} = \frac{\mathsf{p}_{i\setminus 1}(\mathrm{x})}{\mathsf{p}_{1\setminus i}(\mathrm{x})},$$

both the numerator and denominator have degree at most T and therefore the rational polynomial interpolation requires at most $(2T+1)$ evaluation points. Hence $\widetilde{\mathsf{p}}^*_i(\mathrm{x}) = \widetilde{\mathsf{p}}_i(\mathrm{x})$ and $\widetilde{\mathsf{p}}^*_i(z) = \widetilde{\mathsf{p}}_i(z) = \frac{e'_i}{e'_1}$, thus $b_i = 1$. Since the roots of $\mathsf{p}_{i\setminus 1}$ is simply the set difference $D_{i,1} = S_i \setminus S_1$, we have $D^*_{i,1} = D_{i,1} = S_i \setminus S_1$. Similarly $D^*_{1,i} = S_1 \setminus S_i$. Since $\left|\bigcup_{i \in [n]\setminus\{1\}} (D^*_{1,i} \cup D^*_{i,1})\right| = |(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$, we have $b' = 1$. Hence the protocol will output $b = 1$.

Now consider the case where the protocol should output different, namely $|(\bigcup_{i=1}^{n} S_i) \setminus I| > T$. There are two possible cases. In the first case, $|S_i \setminus S_1| > T$ for some i. Then $\widetilde{\mathsf{p}}_i$ has degree at least

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$. $\mathbb{F}$ is a finite field where $|\mathbb{F}| = \Omega(2^\lambda)$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}$ for all $j \in [m]$.

Output: Each party P_i receives $\mathsf{similar}$ if $|(\bigcup_{i=1}^n S_i) \setminus I| \leq T$ and $\mathsf{different}$ otherwise where $I = \bigcap_{i=1}^n S_i$.

Protocol:

1. Each party P_i generates $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TFHE.DistSetup}(1^\lambda, i)$ and sends pk_i to P_1. Then P_1 sends $\mathsf{pk} = (\mathsf{pk}_1 \| \ldots \| \mathsf{pk}_n)$ to all the other parties.
2. P_1 picks a random value $z \in \mathbb{F}$ and sends it to all parties.
3. Each party P_i does the following:
   (a) Define the polynomial $\mathsf{p}_i(\mathrm{x}) := \prod_{a \in S_i}(\mathrm{x} - a)$.
   (b) Compute $e_{i,j} := \mathsf{p}_i(j)$ for $j \in [2T+1]$ and $e'_i := \mathsf{p}_i(z)$.
   (c) Send encrypted evaluations $[\![e_{i,j}]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e_{i,j})$ for $j \in [2T+1]$ and $[\![e'_i]\!] := \mathsf{TFHE.Enc}(\mathsf{pk}, e'_i)$ to P_1.
4. P_1 does the following:
   (a) For each $i \in [n] \setminus \{1\}$, use the algorithm $\mathsf{TFHE.Eval}$ to homomorphically compute an encryption $[\![\widetilde{\mathsf{p}}_i^*(\mathrm{x})]\!]$ by rational polynomial interpolation from $2T+1$ encrypted evaluations $\left(j, \left[\!\!\left[\frac{e_{i,j}}{e_{1,j}}\right]\!\!\right]\right)$ for $j \in [2T+1]$.
   (b) For each $i \in [n] \setminus \{1\}$, homomorphically compute the encrypted predicate $[\![b_i]\!]$ where $b_i = 1$ if $\widetilde{\mathsf{p}}_i^*(z) = \frac{e'_i}{e'_1}$ and 0 otherwise.
   (c) For each $i \in [n] \setminus \{1\}$, homomorphically compute the encrypted roots $[\![D_{i,1}^*]\!]$, $[\![D_{1,i}^*]\!]$ of the numerator and denominator of $\widetilde{\mathsf{p}}_i^*(\mathrm{x})$, respectively.
   (d) Homomorphically compute the encrypted predicate $[\![b']\!]$ where $b' = 1$ if $\left|\bigcup_{i \in [n]\setminus\{1\}}(D_{1,i}^* \cup D_{i,1}^*)\right| \leq T$ and 0 otherwise.
5. P_1 sends $[\![b]\!] = \left[\!\!\left[b' \cdot \prod_{i \in [n]\setminus\{1\}} b_i\right]\!\!\right]$ to all parties who respond with $[\![b : \mathsf{sk}_i]\!] := \mathsf{TFHE.PartialDec}(\mathsf{sk}_i, [\![b]\!])$. P_1 broadcasts $b := \mathsf{TFHE.Combine}(\mathsf{pk}, \{[\![b : \mathsf{sk}_i]\!]\}_{i \in [n]})$ and all parties output $\mathsf{similar}$ if $b = 1$ and $\mathsf{different}$ otherwise.

Figure 6: Multi-party private intersection cardinality testing protocol $\Pi_{\mathsf{TFHE\text{-}CTest\text{-}diff}}$ for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$

$2T+2$ but $\widetilde{\mathsf{p}}_i^*$ is interpolated from $2T+1$ evaluation points, hence $b'_i = 0$ with all but negligible probability. In the second case, $|S_i \setminus S_1| \leq T$ for all $i \in [n] \setminus \{1\}$. Then $D_{i,1}^* = D_{i,1} = S_i \setminus S_1$, $D_{1,i}^* = S_1 \setminus S_i$, and $b_i = 1$ for all i. Since $|(\bigcup_{i=1}^n S_i) \setminus I| > T$, $b' = 0$. In both cases, we have $b = b' \cdot \prod_{i \in [n]\setminus\{1\}} b_i = 0$ with all but negligible probability.

Communication Cost. Each party sends $(2T+2)$ TFHE encryptions and one partial decryption to P_1 where each plaintext is a field element. P_1 sends one ciphertext to every other party. The size of each encryption and each partial decryption is $\mathsf{poly}(\lambda)$. Thus, the overall communication complexity is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ in a star network and the protocols runs in $O(1)$ rounds.

Security. The proof of security is identical to the proof of Theorem 5.1. We describe it below for the sake of completeness.

Consider an environment $\mathcal{Z}$ who corrupts a set $\mathcal{S}^*$ of n^* parties where $n^* < n$. The simulator Sim has input $w \in \{\mathsf{similar}, \mathsf{different}\}$ from the ideal functionality. Sim sets a bit $b^* = 1$ if $w = \mathsf{similar}$ and $b^* = 0$ otherwise. Also, for each corrupt party P_i, Sim has as input the tuple (S_i, r_i) indicating the party's input and randomness for the protocol. The strategy of the simulator Sim for our protocol is described below.

1. Sim runs the distributed key generation algorithm $\mathsf{TFHE.DistSetup}(1^\lambda, i)$ of the TFHE scheme honestly on behalf of each honest party P_i as in the real world. Note that Sim also knows $(\{\mathsf{sk}_i\}_{i \in S^*})$ as it knows the randomness for the corrupt parties.

2. In Steps 2-4 of the protocol, Sim plays the role of the honest parties exactly as in the real world except that on behalf of every honest party P_i, whenever P_i has to send any ciphertext, compute $[\![0]\!] = \mathsf{TFHE.Enc}(0)$ using fresh randomness.

3. In Step 5, on behalf of each honest party P_i, instead of sending the value $[\![b : \mathsf{sk}_i]\!]$ by running the honest $\mathsf{TFHE.PartialDec}$ algorithm as in the real world, Sim computes the partial decryptions by running the simulator $\mathsf{TFHE.Sim}$ as follows: $\{[\![b : \mathsf{Sim}_i]\!]\}_{i \in [n] \setminus \mathcal{S}^*} \leftarrow \mathsf{TFHE.Sim}(\mathsf{C}, b^*, [\![b]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}^*})$ where the circuit C denotes the whole computation done by P_1 in the real world to evaluate bit b. On behalf of the honest party P_i the simulator sends $[\![b : \mathsf{Sim}_i]\!]$. This corresponds to the ideal world.

We now show that the above simulation strategy is successful against all environments $\mathcal{Z}$ that corrupt parties in a semi-honest manner. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hybrid_0 corresponds to the real world and the last hybrid Hybrid_2 corresponds to the ideal world.

- Hybrid_0 **- Real World:** In this hybrid, consider a simulator SimHyb that plays the role of the honest parties as in the real world.
- Hybrid_1 **- Simulate Partial Decryptions:** - In this hybrid, in Step 5, SimHyb simulates the partial decryptions generated by the honest parties as done in the ideal world. That is, the simulator calls $\{[\![b : \mathsf{Sim}_i]\!]\}_{i \in [n] \setminus \mathcal{S}} \leftarrow \mathsf{TFHE.Sim}(\mathsf{C}, b^*, [\![b]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}})$. On behalf of the honest party P_i the simulator sends $[\![b : \mathsf{Sim}_i]\!]$ instead of $[\![b : \mathsf{sk}_i]\!]$.
- Hybrid_2 **- Switch Encryptions:** In this hybrid, SimHyb now computes every ciphertext generated on behalf of any honest party as encryptions of 0 as done by Sim in the ideal world. This hybrid corresponds to the ideal world.

We now show that every pair of consecutive hybrids is computationally indistinguishable.

Lemma 5.5. *Assuming the simulation security of the threshold fully homomorphic encryption scheme,* Hybrid_0 *is computationally indistinguishable from* Hybrid_1.

Proof. This is identical to the proof of Lemma 5.2. □

Lemma 5.6. *Assuming the semantic security of the threshold fully homomorphic encryption scheme,* Hybrid_1 *is computationally indistinguishable from* Hybrid_2.

Proof. This is identical to the proof of Lemma 5.3. □

□

6 TAHE-Based Protocol for $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$

In this section, we present a multi-party protocol for private intersection cardinality testing for functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ based on threshold additive homomorphic encryption with distributed setup. That is, the parties learn whether their sets satisfy $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. Our protocol works in the star network communication model where P_1 is the central party.

In our construction, we need a secure multi-party computation (MPC) protocol that tests the singularity of a specific Hankel matrix (defined later), which we discuss in Section 6.1. Using this, we present our complete protocol in Section 6.2.

6.1 Singularity Testing of Hankel Matrices

In Section 6.2, we will see that intersection cardinality testing can be reduced to determining whether the determinant of a specific matrix is 0 or not. The latter problem can be reduced to computing the so-called "Half-GCD" of two specific polynomials. In this section, we present a summary of the various results that go into these reductions and refer the reader to the cited works for further details.

Half-GCD Problem. Consider the ring of polynomials $\mathbb{F}[\mathrm{x}]$. Note that since $\mathbb{F}[\mathrm{x}]$ is a Euclidean domain, Euclid's GCD algorithm can be applied to polynomials as well. Consider $\mathsf{p}_0, \mathsf{p}_1 \in \mathbb{F}[\mathrm{x}]$ with $d = \deg(\mathsf{p}_0) > \deg(\mathsf{p}_1) \geq 0$. The Euclidean algorithm can be viewed as a sequence of transformations of 2-vectors as below:

$$\begin{pmatrix} \mathsf{p}_0 \\ \mathsf{p}_1 \end{pmatrix} \xrightarrow{M_1} \begin{pmatrix} \mathsf{p}_1 \\ \mathsf{p}_2 \end{pmatrix} \xrightarrow{M_2} \cdots \xrightarrow{M_{h-1}} \begin{pmatrix} \mathsf{p}_{h-1} \\ \mathsf{p}_h \end{pmatrix} \xrightarrow{M_h} \begin{pmatrix} \mathsf{p}_h \\ 0 \end{pmatrix} \tag{2}$$

Here, $M_1, \ldots, M_h$ are 2×2 matrices, $\mathsf{p}_2, \ldots, \mathsf{p}_h \in \mathbb{F}[\mathrm{x}]$. For vectors U, V and a matrix M, we write $U \xrightarrow{M} V$ to denote $U = MV$.

Equation 2 can be correctly interpreted if we define

$$M_i = \begin{pmatrix} \mathsf{q}_i & 1 \\ 1 & 0 \end{pmatrix}.$$

We call such matrices *elementary matrices*, where q_i is a polynomial of positive degree. We also refer to q_i as the *partial quotient* in M_i. A *regular matrix* M is a product of zero or more elementary matrices, namely

$$M = M_1 M_2 \ldots M_k \qquad (k \geq 0)$$

where if $k = 0$, then M is defined to be the identity matrix of order 2.

We define the *half-GCD (HGCD)* problem for the polynomial ring $\mathbb{F}[\mathrm{x}]$ as follows. Given $\mathsf{p}_0, \mathsf{p}_1 \in \mathbb{F}[\mathrm{x}]$ with $d = \deg(\mathsf{p}_0) > \deg(\mathsf{p}_1) \geq 0$, compute a regular matrix

$$M = \mathtt{HGCD}(\mathsf{p}_0, \mathsf{p}_1)$$

such that if

$$\begin{pmatrix} \mathsf{p}_0 \\ \mathsf{p}_1 \end{pmatrix} \xrightarrow{M} \begin{pmatrix} \mathsf{p}_2 \\ \mathsf{p}_3 \end{pmatrix},$$

then

$$\deg(\mathsf{p}_2) \geq d/2 > \deg(\mathsf{p}_3).$$

We now recall the result of Thull and Yap [TY90] on the computational complexity of HGCD.

Imported Theorem 6.1. *Consider the polynomial ring* $\mathbb{F}[\mathrm{x}]$ *and the polynomials* $\mathsf{p}_0, \mathsf{p}_1 \in \mathbb{F}[\mathrm{x}]$ *with* $d = \deg(\mathsf{p}_0) > \deg(\mathsf{p}_1) \geq 0$. *The computational complexity of the HGCD problem is* $O(d \log^2 d)$.

Singularity Testing of Hankel Matrices. Next, we proceed to outline the results that enable us to use the HGCD problem to test singularity of Hankel matrices. A Hankel matrix is a matrix in which each ascending skew-diagonal from left to right is constant. We will be working with square Hankel matrices. In particular, a $(k+1) \times (k+1)$ Hankel matrix takes the form

$$H = \begin{pmatrix} a_0 & a_1 & \dots & a_k \\ a_1 & a_2 & \dots & a_{k+1} \\ \vdots & \vdots & \vdots & \vdots \\ a_k & a_{k+1} & \dots & a_{2k} \end{pmatrix}$$

where the $2k+1$ entries $a_0, a_1, \dots, a_{2k}$ define H. Define the two polynomials

$$\mathsf{p}_0(\mathrm{x}) = \mathrm{x}^{2k+1}$$
$$\mathsf{p}_1(\mathrm{x}) = a_0 + a_1\mathrm{x} + a_2\mathrm{x}^2 + \ldots + a_{2k}\mathrm{x}^{2k}$$

where $\mathsf{p}_0, \mathsf{p}_1 \in \mathbb{F}[\mathrm{x}]$. Let $M = \mathtt{HGCD}(\mathsf{p}_0, \mathsf{p}_1)$ and

$$\begin{pmatrix} \mathsf{p}_0 \\ \mathsf{p}_1 \end{pmatrix} \xrightarrow{M} \begin{pmatrix} \mathsf{p}_2 \\ \mathsf{p}_3 \end{pmatrix}.$$

Then we have

$$\deg(\mathsf{p}_2) \geq k+1 > \deg(\mathsf{p}_3).$$

We recall the setting and results of Brent, Gustavson and Yun [BGY80] that elegantly connect the singularity of H with the HGCD of $\mathsf{p}_0(\mathrm{x})$ and $\mathsf{p}_1(\mathrm{x})$.

Imported Theorem 6.2. *The Hankel matrix* H *is singular iff* $\deg(\mathsf{p}_3) < k$.

Putting Imported Theorems 6.1 and 6.2 together, we have the following theorem.

Imported Theorem 6.3. *The computational complexity of testing singularity of a* $(k+1)\times(k+1)$ *Hankel matrix is* $O(k \log^2 k)$.

Multi-Party Singularity Testing. Looking ahead, in our multi-party intersection cardinality testing protocol, we will need to test for the singularity of a Hankel matrix H which the parties have additive shares of, and the parties will run a secure multi-party computation (MPC) protocol to jointly test for the singulairty of H. The ideal functionality $\mathcal{F}_{\mathsf{SingTest}}$ for the multi-party minimal polynomial computation is defined in Figure 7. We will need an MPC protocol that realizes $\mathcal{F}_{\mathsf{SingTest}}$ with communication complexity at most $\widetilde{O}(k \cdot n \cdot \mathsf{poly}(\lambda))$. Any such protocol suffices, and we denote by Π_{SingTest} the MPC protocol realizing $\mathcal{F}_{\mathsf{SingTest}}$.

Here we describe two such protocols with communication complexity $\widetilde{O}(k \cdot n \cdot \mathsf{poly}(\lambda))$ based on TAHE. In the first protocol, after the TAHE setup, each party P_i sends $[\![H_i]\!]$ to P_1 and P_1 homomorphically computes $[\![H]\!]$. Afterwards P_1 can homomorphically evaluate a circuit C that computes a predicate $b \stackrel{?}{=} (\det(H) = 0)$, following the ideas from [FH96, CDN01]. Finally the

Parameters: Parties $P_1, \ldots, P_n$.

Inputs: Each party P_i inputs $2k+1$ field elements $a_{0,i}, a_{1,i}, \ldots, a_{2k,i} \in \mathbb{F}$.

Output: Let

$$H_i = \begin{pmatrix} a_{0,i} & a_{1,i} & \ldots & a_{k,i} \\ a_{1,i} & a_{2,i} & \ldots & a_{k+1,i} \\ \vdots & \vdots & \vdots & \vdots \\ a_{k,i} & a_{k+1,i} & \ldots & a_{2k,i} \end{pmatrix}$$

be the Hankel matrix defined by the inputs of party P_i for $i = 1, \ldots, n$, and let

$$H = \sum_{i=1}^{n} H_i.$$

Determine if the Hankel matrix H is singular. Each party receives 0 if H is singular, and 1 otherwise.

Figure 7: Ideal functionality $\mathcal{F}_{\mathsf{SingTest}}$ for multi-party singularity testing of a Hankel matrix.

parties jointly decrypt the encrypted output. Since the size and depth of C are both $O(k \log^2 k)$ by Imported Theorem 6.3, the total communication complexity of this protocol is $O(k \log^2 k \cdot n \cdot \mathsf{poly}(\lambda))$ and the round complexity is $O(k \log^2 k)$.

As a second protocol, the parties jointly compute another C' that takes H and a random PRF key r as input and outputs a Yao's garbled circuit [Yao86] that computes C. This approach is inspired by the work of Damgård et al. [DIK+08]. Since both H and r are additively shared among all the parties, this MPC can be done similarly as in the previous protocol, namely P_1 first obtains $[\![H]\!]$ and $[\![r]\!]$ and then homomorphically evaluates C'. Since the size C' is $\widetilde{O}(k \cdot \mathsf{poly}(\lambda))$ and the depth of C' is constant assuming PRG is a circuit in NC^1 [AIK05], the total communication complexity of this protocol is $\widetilde{O}(k \cdot n \cdot \mathsf{poly}(\lambda))$ and the round complexity is $O(1)$.

Two-party case. Notice that for two parties, $\mathcal{F}_{\mathsf{SingTest}}$ can be instantiated via Yao's garbled circuits with communication complexity $\widetilde{O}(k \cdot \mathsf{poly}(\lambda))$.

6.2 Our Protocol

In this section we present our multi-party private intersection cardinality testing protocol. That is, the parties learn whether their sets satisfy $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$.

At a high level, our protocol first encodes each party P_i's set as a polynomial $\mathsf{p}_i(\mathrm{x}) = \sum_{j=1}^{m} \mathrm{x}^{a_j^i}$, and let $\mathsf{p}(\mathrm{x}) := (n-1)\mathsf{p}_1(\mathrm{x}) - \sum_{i=2}^{n} \mathsf{p}_i(\mathrm{x})$. Notice that a term x^a is cancelled out in the polynomial p if and only if the element a is in the set intersection I. Therefore, the number of monomials in p is exactly $|(\bigcup_{i=1}^{n} S_i) \setminus I|$.

To determine if the number of monomials in p is $\leq T$, we can apply the polynomial sparsity test of Grigorescu et al. [GJR10] similarly as in [GS19a]. In particular, pick a field $\mathbb{F}_q$, sample $u \xleftarrow{\$} \mathbb{F}_q$ uniformly at random, and compute the Hankel matrix

$$H = \begin{bmatrix} \mathsf{p}(u^0) & \mathsf{p}(u^1) & \ldots & \mathsf{p}(u^T) \\ \mathsf{p}(u^1) & \mathsf{p}(u^2) & \ldots & \mathsf{p}(u^{T+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{p}(u^T) & \mathsf{p}(u^{T+1}) & \ldots & \mathsf{p}(u^{2T}) \end{bmatrix}.$$

Determining if the number of monomials in p is $\leq T$ can be reduced to testing the singularity of H. In particular, we take the following theorem from [GJR10, Theorem 3] and [GS19a, Theorem 1].

Imported Theorem 6.4. *Let $q > T(T+1)(p-1)2^{\kappa}$ be a prime. If the number of monomials in p is $\leq T$, then $\Pr[\det(H) = 0] = 1$, and if the number of monomials in p is $> T$, then $\Pr[\det(H) = 0] \leq 2^{-\kappa}$,*

In our multi-party private intersection cardinality testing protocol, the parties will first compute additive shares of H and then run a multi-party minimal polynomial computation protocol to jointly test the singularity of H. The protocol is presented in Figure 8.

1. **Computing Shares of Hankel Matrix H.**
   (a) P_1 picks a uniform random $u \xleftarrow{\$} \mathbb{F}_q$ and sends to all other parties.
   (b) P_1 sets a polynomial $\mathsf{p}_1(\mathrm{x}) = \sum_{j=1}^{m}(n-1)\cdot \mathrm{x}^{a_j^1}$ in $\mathbb{F}_q[\mathrm{x}]$.
   (c) Each party P_i $(i = 2, 3, \ldots, n)$ sets a polynomial $\mathsf{p}_i(\mathrm{x}) = -\sum_{j=1}^{m}\mathrm{x}^{a_j^i}$ in $\mathbb{F}_q[\mathrm{x}]$.
   (d) Each party P_i $(i = 1, 2, 3, \ldots, n)$ computes the values $a_{j,i} = \mathsf{p}_i(u^j)$ for $j = 0, 1, \ldots, 2T$.
2. **Matrix Singularity Testing of H.** Parties invoke an instance of $\mathcal{F}_{\mathsf{SingTest}}$ where each party P_i inputs $a_{0,i}, \ldots, a_{2T,i}$ and obtains a bit b.
3. **Output.** Each party P_i outputs $\mathsf{similar}$ if $b = 0$ and $\mathsf{different}$ otherwise.

Figure 8: Multi-party private intersection cardinality testing protocol $\Pi_{\mathsf{CTest\text{-}diff}}$.

Theorem 6.5. *Let $q > T(T+1)(p-1)2^{\kappa}$ be a prime. Assuming threshold additive homomorphic encryption scheme with distributed setup, the protocol $\Pi_{\mathsf{CTest\text{-}diff}}$ (Figure 8) securely realizes $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ in the $\mathcal{F}_{\mathsf{SingTest}}$-hybrid model.*

Proof. **Correctness.** By the correctness of $\mathcal{F}_{\mathsf{SingTest}}$, in Step 2 all the parties learn a bit b and $b = 0$ if and only if H is singular, where H is the Hankel matrix $H = \sum_{i=1}^{n} H_i$ and each Hankel matrix H_i is defined by the inputs of party P_i as

$$H_i = \begin{pmatrix} a_{0,i} & a_{1,i} & \ldots & a_{T,i} \\ a_{1,i} & a_{2,i} & \ldots & a_{T+1,i} \\ \vdots & \vdots & \ddots & \vdots \\ a_{T,i} & a_{T+1,i} & \ldots & a_{2T,i} \end{pmatrix}$$

for $i = 1, \ldots, n$. By Imported Theorem 6.4, $b = 0$ if and only if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$ with all but negligible probability. Therefore the protocol is correct with all but negligible probability.

Communication Cost. The communication cost is the same as the protocol Π_{SingTest}. In particular, the round complexity is $O(1)$ in a star network and the total communication complexity is $\widetilde{O}(T \cdot n \cdot \mathsf{poly}(\lambda))$.

Security. We construct a PPT Sim which simulates the view of the corrupted parties. The simulator Sim gets the output $w \in \{\mathsf{similar}, \mathsf{different}\}$ from the ideal functionality. Sim sets a bit $b^* = 1$ if $w = \mathsf{similar}$ and $b^* = 0$ otherwise. Also, for each corrupt party P_i, Sim has as input the tuple (S_i, r_i) indicating the party's input and randomness for the protocol. The strategy of the simulator Sim for our protocol is described below.

1. Invoke the corrupted parties with their corresponding inputs and randomness.
2. Play the role of the honest parties as follows: Run the protocol honestly. Note that P_1 is the only party that ever sends a message, so this step in the simulation is trivial.
3. In Step 2, play the role of $\mathcal{F}_{\mathsf{SingTest}}$ and respond b^*.
4. Finally, output the view of the corrupted parties.

Next we argue that the view of the corrupted parties generated by Sim is computationally indistinguishable to their view in the real world from $\mathcal{Z}$'s point of view. The only difference between the real and ideal worlds is that in the ideal world, the output from $\mathcal{F}_{\mathsf{SingTest}}$ is replaced by 0 if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$ and 1 otherwise. This is computationally indistinguishable from the real world because of the correctness of the protocol. □

Corollary 6.6. *Assuming TAHE with distributed setup, protocol* $\Pi_{\mathsf{CTest\text{-}diff}}$ *(Figure 8) securely realizes* $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ *in the star network communication model with communication complexity* $\widetilde{O}(n \cdot T \cdot \mathsf{poly}(\lambda))$ *and round complexity* $O(1)$*.*

7 Threshold PSI for $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$

Recall that in a multi-party threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ defined in Figure 2, each party wishes to learn the intersection of all their sets if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$, that is, if the size of the union of all their sets minus the intersection is less than the threshold T. In this section, we describe our multi-party threshold PSI protocol based on any protocol for multi-party private intersection cardinality testing. We rely on TAHE with distributed setup.

Theorem 7.1. *Assuming threshold additive homomorphic encryption with distributed setup, protocol* $\Pi_{\mathsf{TPSI\text{-}diff}}$ *(Figure 9) securely realizes* $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ *in the* $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$*-hybrid model in the star network communication model. Our protocol is secure against a semi-honest adversary that can corrupt up to* $(n-1)$ *parties.*

The protocol runs in a constant number of rounds and the communication complexity is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ in the $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$-hybrid model. We then instantiate the $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$-hybrid with the two protocols from the previous sections: one based on TFHE from Section 5.2 that has round complexity $O(1)$ and $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ communication complexity and the other based on TAHE from Section 6 that has round complexity $O(1)$ and communication complexity $\widetilde{O}(n \cdot T \cdot \mathsf{poly}(\lambda))$. Formally, we get the following corollaries:

Corollary 7.2. *Assuming TFHE (resp. TAHE) with distributed setup, protocol* $\Pi_{\mathsf{TPSI\text{-}diff}}$ *(Figure 9) securely realizes* $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ *in the star network communication model with communication complexity* $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ *(resp.* $\widetilde{O}(n \cdot T \cdot \mathsf{poly}(\lambda))$*) and round complexity* $O(1)$*.*

Our threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ is almost identical and we defer the details to Appendix B.

7.1 Protocol

Consider n parties $P_1, \ldots, P_n$ with input sets $S_1, \ldots, S_n$ of size m and a star network where the central party is P_1. The parties first run the private intersection cardinality testing protocols for functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ from the previous sections and proceed if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$. Then, each party P_i encodes its set as a polynomial $\mathsf{p}'_i(\mathrm{x}) = (\mathrm{x} - r_i) \cdot \prod_{j=1}^{m} (\mathrm{x} - a_j^i)$ where r_i is picked uniformly at random. The parties then compute $(3T+4)$ evaluations of the following polynomial $\mathsf{V}(\cdot)$ on points $1, \ldots, (3T+4)$ using threshold additive homomorphic encryption: $\mathsf{V}(\mathrm{x}) = \sum_{i=1}^{n} (\mathsf{p}'_i(\mathrm{x}) \cdot \mathsf{R}_i(\mathrm{x}))$ where each $\mathsf{R}_i(\cdot)$ is a uniformly random polynomial of degree T that is computed as an addition of n random polynomials - one generated by each party. Then, each party P_i interpolates the degree $(3T+3)$ rational polynomial $\frac{\mathsf{V}(\cdot)}{\mathsf{p}'_i(\cdot)}$ using the $(3T+4)$ evaluations. Finally, each party outputs the intersection as $S_i \setminus D_i$ where D_i denotes the roots of the above interpolated polynomial. Our protocol is formally described in Figure 9.

Two-party case. For two parties Alice and Bob, we can rely on AHE alone, where Alice holds the secret key. In particular, define $\mathsf{V}(\mathrm{x}) := \mathsf{p}_A(\mathrm{x}) \cdot \left(\mathsf{R}_1^A(\mathrm{x}) + \mathsf{R}_1^B(\mathrm{x})\right) + \mathsf{p}_B(\mathrm{x}) \cdot \left(\mathsf{R}_2^A(\mathrm{x}) + \mathsf{R}_2^B(\mathrm{x})\right)$, where $(\mathsf{R}_1^A, \mathsf{R}_2^A)$ and $(\mathsf{R}_1^B, \mathsf{R}_2^B)$ are uniformly random polynomials of degree T generated by Alice and Bob, respectively. To obtain an evaluation of $\mathsf{V}(x)$, Alice first sends an encryption of $\mathsf{p}_A(x)$ and $\mathsf{R}_2^A(x)$ to Bob. Then Bob homomorphically computes an encryption of $r = \mathsf{p}_A(x) \cdot \mathsf{R}_1^B(x) + \mathsf{p}_B(x) \cdot \left(\mathsf{R}_2^A(x) + \mathsf{R}_2^B(x)\right)$ and sends it back. Alice can decrypt $[\![r]\!]$ and compute $\mathsf{V}(x) = \mathsf{p}_A(x) \cdot \mathsf{R}_1^A(x) + r$. The communication complexity is $O(T \cdot \mathsf{poly}(\lambda))$.

7.2 Security Proof

Correctness. If $|(\bigcup_{i=1}^{n} S_i) \setminus I| > T$, then the protocol terminates after the first step – private intersection cardinality testing. If, on the other hand, $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$, observe that polynomial $\mathsf{V}(\mathrm{x})$ can be rewritten as $\sum_{i=1}^{n} \mathsf{p}'_i(\mathrm{x}) \cdot U_i(\mathrm{x})$ where each U_i is a uniformly random polynomial of degree at most $T+1$. Now, from the correctness of the TAHE scheme, each party P_i learns $3T+4$ evaluations of the rational polynomial:

$$\mathsf{q}_i(\mathrm{x}) = \frac{\mathsf{V}(\mathrm{x})}{\mathsf{p}'_i(\mathrm{x})} = \frac{\sum_{i=1}^{n} \mathsf{p}'_i(\mathrm{x}) \cdot U_i(\mathrm{x})}{\mathsf{p}'_i(\mathrm{x})} = \frac{\sum_{i=1}^{n} \mathsf{p}_{i \setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_i) \cdot U_i(\mathrm{x})}{\mathsf{p}_{i \setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_i)}.$$

Since $|S_i - I| \leq T$ for each $i \in [n]$, the numerator is a polynomial of degree at most $2T+2$ and the denominator is a polynomial of degree at most $T+1$. Further, since each U_i is uniformly random, by Lemma 2.5, the numerator is a random degree $2T+2$ polynomial. By Imported Lemma 2.3, the gcd of the polynomials in the numerator and denominator is 1 and hence no other terms will get canceled out. Therefore, each party P_i can interpolate this rational polynomial using $3T+4$ evaluation points and thereby learn the numerator and denominator. Finally, observe that for each party P_i, the roots of the denominator contains the set $S_i \setminus I$ and a random r_i, from which P_i can easily compute the intersection I.

Communication Cost. The first phase of the protocol, namely private intersection cardinality testing, has a communication complexity of $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ when instantiated with the TFHE-based scheme in Section 5.2 and a communication complexity of $\widetilde{O}(n \cdot T \cdot \mathsf{poly}(\lambda))$ when instantiated with the TAHE-based scheme in Section 6.

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$. $\mathbb{F}$ is a finite field where $|\mathbb{F}| = \Omega(2^\lambda)$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}$ for all $j \in [m]$.

Output: Each party P_i receives the set intersection $I = \bigcap_{i=1}^n S_i$ if and only if $|(\bigcup_{i=1}^n S_i) \setminus I| \leq T$.

Protocol:

1. **Private Intersection Cardinality Testing.** The parties invoke $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ on inputs $S_1, \ldots, S_n$ and receive back $w \in \{\mathsf{similar}, \mathsf{different}\}$. If $w = \mathsf{different}$ then all parties output $\perp$.
2. **TAHE Key Generation.** Each party P_i generates $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TAHE.DistSetup}(1^\lambda, i)$ and sends pk_i to P_1. Then P_1 sends $\mathsf{pk} = (\mathsf{pk}_1 \| \ldots \| \mathsf{pk}_n)$ to all the other parties.
3. **Evaluations of Random Polynomial.** In this phase the parties will evaluate a polynomial
$$\mathsf{V}(\mathrm{x}) = \sum_{i=1}^{n} \left(\mathsf{p}_i'(\mathrm{x}) \cdot \left(\mathsf{R}_1(\mathrm{x}) + \ldots + \mathsf{R}_{i-1}(\mathrm{x}) + \widetilde{\mathsf{R}}_i(\mathrm{x}) + \mathsf{R}_{i+1}(\mathrm{x}) \ldots + \mathsf{R}_n(\mathrm{x}) \right) \right)$$
for $x \in [3T+4]$ where the terms are defined as follows.
   (a) Each party P_i defines $\mathsf{p}_i(\mathrm{x}) = \prod_{j=1}^m (\mathrm{x} - a_j^i)$ and $\mathsf{p}_i'(\mathrm{x}) = \mathsf{p}_i(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ where $r_i \xleftarrow{\$} \mathbb{F}$.
   (b) Each party P_i uniformly samples $\mathsf{R}_i, \widetilde{\mathsf{R}}_i \xleftarrow{\$} \mathbb{F}[\mathrm{x}]$ of degree $T+1$, computes $\mathsf{R}_i(x)$ for $x \in [3T+4]$ and sends encrypted $[\![\mathsf{R}_i(x)]\!]$ to P_1.
   (c) For each $i \in [n], x \in [3T+4]$, party P_1 sends $[\![e_{i,x}]\!] = \left[\!\!\left[\sum_{j \in [n] \setminus \{i\}} \mathsf{R}_j(x) \right]\!\!\right]$ to P_i.
   (d) For each $x \in [3T+4]$, each party P_i sends $[\![v_{i,x}]\!] = \left[\!\!\left[\mathsf{p}_i'(x) \cdot \left(e_{i,x} + \widetilde{\mathsf{R}}_i(x) \right) \right]\!\!\right]$ to P_1.
   (e) For each $x \in [3T+4]$, P_1 sends $[\![v_x]\!] = [\![\sum_{i=1}^n v_{i,x}]\!]$ to all P_i.
   (f) For each $x \in [3T+4]$, each party P_i sends $[\![v_x : \mathsf{sk}_i]\!] \leftarrow \mathsf{TAHE.PartialDec}(\mathsf{sk}_i, [\![v_x]\!])$ to P_1.
   (g) For each $x \in [3T+4]$, P_1 sends $\mathsf{V}(x) = \mathsf{TAHE.Combine}(\mathsf{pk}, \{[\![v_x : \mathsf{sk}_i]\!]\}_{i \in [n]})$ to all P_i.
4. **Computing Set Intersection.** Each party P_i does the following:
   (a) Interpolate $\mathsf{q}_i(\mathrm{x})$ to be the degree $3T+3$ rational polynomial such that $\mathsf{q}_i(x) = \frac{\mathsf{V}(x)}{\mathsf{p}_i'(x)}$ for $x \in [3T+4]$ and the gcd of the numerator and denominator is 1. Let D_i be the roots of the denominator of $\mathsf{q}_i(\mathrm{x})$.
   (b) Output the set intersection $I = S_i \setminus D_i$.

Figure 9: Multi-party threshold PSI protocol $\Pi_{\mathsf{TPSI\text{-}diff}}$ for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$.

We now analyze the communication cost for the second phase where the parties compute the concrete intersection. The TAHE key generation is independent of the set sizes and the threshold T and has a communication complexity of only $O(n \cdot \mathsf{poly}(\lambda))$. The bottleneck of the protocol is in Step 3, that is, evaluating the random polynomial. In Steps 3b, 3d, and 3f, every party sends $3T+4$ encryptions or partial decryptions to P_1 hence the cost for these steps is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$. In Steps 3c, 3e, and 3g, P_1 sends $3T+4$ ciphertexts or plaintexts to every other party so the cost of these steps is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$. Finally, the last stage, namely computing the set intersection, does not involve any communication. Thus, the overall communication cost for computing the intersection is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$.

Therefore, when the private intersection cardinality testing protocol is instantiated with the TFHE-based protocol, the overall communication complexity is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ and when instan-

tiated with the TAHE-based scheme, the overall communication complexity is $\widetilde{O}(n \cdot T \cdot \mathsf{poly}(\lambda))$ for some apriori fixed polynomial $\mathsf{poly}(\cdot)$ and is independent of the size of each input set m.

Security. Consider an environment $\mathcal{Z}$ who corrupts a set $\mathcal{S}^*$ of n^* parties where $n^* < n$. The simulator Sim has the output of the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$, namely the intersection set I or $\perp$. Sim sets $w = \mathsf{similar}$ if the output is I and $w = \mathsf{different}$ if the output is $\perp$. In addition, Sim has the tuple (S_i, r_i) for each corrupt party P_i indicating the party's input and randomness for the protocol. The strategy of the simulator Sim for our multi-party threshold PSI protocol is described below.

(a) Private Intersection Cardinality Testing: Sim plays the role of the ideal functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ and responds with w.

(b) TAHE Key Generation: Sim runs the distributed key generation algorithm $\mathsf{TAHE.DistSetup}(1^\lambda, i)$ of the TAHE scheme honestly on behalf of each honest party P_i as in the real world. Note that Sim also knows $(\{\mathsf{sk}_i\}_{i \in S^*})$ as it knows the randomness for the corrupt parties.

(c) Evaluations of Random Polynomial: Sim does the following:

1. Encode the intersection set $I = \{b_1, \ldots, b_{|I|}\}$ as a polynomial as follows: $\mathsf{p}_I(\mathrm{x}) = \Pi_{i=1}^{|I|}(\mathrm{x} - b_i)$.
2. Pick a random polynomial $U(\cdot)$ of degree $2T + 2$ and set the polynomial $\mathsf{V}(\mathrm{x})$ as follows: $\mathsf{V}(\mathrm{x}) = \mathsf{p}_I(\mathrm{x}) \cdot U(\mathrm{x})$.
3. In Steps 3b-3e, on behalf of every honest party P_i, whenever P_i has to send any ciphertext, send $[\![0]\!]$ using fresh randomness.
4. For each $x \in [3T + 4]$, let $[\![v_x]\!]$ denote the ciphertext that is sent to all the parties at the end of Step 3f.
5. In Step 3f, for each $j \in [3T + 4]$, on behalf of each honest party P_i, instead of computing $\{[\![v_x : \mathsf{sk}_i]\!]\}$ by running the honest $\mathsf{TAHE.PartialDec}$ algorithm as in the real world, Sim computes the partial decryptions by running the simulator $\mathsf{TAHE.Sim}$ as follows: $\{[\![v_x : \mathsf{sk}_i]\!]\} \leftarrow \mathsf{TAHE.Sim}(\mathsf{C}, \mathsf{V}(x), [\![v_x]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}^*})$, where C is the public linear circuit to compute $\mathsf{V}(x)$ by P_1.
6. Finally, in Step 3g, if P_1 is honest, send the evaluations of polynomial $\mathsf{V}(x)$ as in the real world description.

Hybrids. We now show that the above simulation strategy is successful against all environments $\mathcal{Z}$ that corrupt parties in a semi-honest manner. That is, the view of the corrupt parties along with the output of the honest parties is computationally indistinguishable in the real and ideal worlds. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hybrid_0 corresponds to the real world and the last hybrid Hybrid_4 corresponds to the ideal world.

- Hybrid_0 **- Real World:** In this hybrid, consider a simulator SimHyb that plays the role of the honest parties as in the real world.

- Hybrid_1 **- Private Intersection Cardinality Testing:** In this hybrid, SimHyb plays the role of the ideal functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ and responds with $\mathsf{similar}$ if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$ and $\mathsf{different}$ otherwise.
- Hybrid_2 **- Simulate Partial Decryptions:** In this hybrid, in the evaluations of random polynomial, SimHyb simulates the partial decryptions generated by the honest parties in Step 3f as done in the ideal world. That is, for each $j \in [3T+4]$, SimHyb computes the partial decryptions as $\{[\![v_x : \mathsf{sk}_i]\!]\} \leftarrow \mathsf{TAHE.Sim}(\mathsf{C}, \mathsf{V}(x), [\![v_x]\!], \{\mathsf{sk}_i\}_{i \in \mathcal{S}^*})$. Observe that the polynomial $\mathsf{V}(\cdot)$ is still computed as in the real world (and in Hybrid_2).
- Hybrid_3 **- Switch Polynomial Computation:** In this hybrid, the polynomial $\mathsf{V}(\cdot)$ is no longer computed as in the real world. Instead, SimHyb now picks a random polynomial $U(\cdot)$ of degree $2T+2$ and sets the polynomial $\mathsf{V}(\cdot)$ as follows: $\mathsf{V}(\mathrm{x}) = \mathsf{p}_I(\mathrm{x}) \cdot U(\mathrm{x})$.
- Hybrid_4 **- Switch Encryptions:** In this hybrid, in the evaluations of random polynomial, SimHyb now computes every ciphertext generated on behalf of any honest party as encryptions of 0 as done by Sim in the ideal world. This hybrid corresponds to the ideal world.

We now show that every pair of consecutive hybrids is computationally indistinguishable.

Lemma 7.3. Hybrid_0 *is computationally indistinguishable from* Hybrid_1 *based on the correctness of our protocol.*

Proof. The only difference between the two hybrids is that in Hybrid_0, the simulator SimHyb calls $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ honestly while in Hybrid_1, SimHyb plays the role of the ideal functionality $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ responds with $\mathsf{similar}$ if $|(\bigcup_{i=1}^{n} S_i) \setminus I| \leq T$ and $\mathsf{different}$ otherwise. The output of $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ in Hybrid_1 is always correct. Based on the correctness of our protocol $\Pi_{\mathsf{TPSI\text{-}diff}}$, the output of $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ in Hybrid_0 is correct with overwhelming probability. Hence Hybrid_0 and Hybrid_1 are computationally indistinguishable. □

Lemma 7.4. *Assuming the simulation security of the threshold additive homomorphic encryption scheme,* Hybrid_1 *is computationally indistinguishable from* Hybrid_2.

Proof. The only difference between the two hybrids is that in Hybrid_1, the simulator SimHyb generates the partial decryptions of the TAHE scheme on behalf of the honest parties as in the real world while in Hybrid_2, they are simulated by running the simulator $\mathsf{TAHE.Sim}$. We now show that if there exists an adversarial environment $\mathcal{Z}$ that can distinguish between these two hybrids with some non-negligible probability ϵ, we will come up with a reduction $\mathcal{A}$ that can break the simulation security of the TAHE scheme.

$\mathcal{A}$ interacts with a challenger $\mathcal{C}$ in the simulation security game for TAHE and with the environment $\mathcal{Z}$ in the game between Hybrid_1 and Hybrid_2. $\mathcal{A}$ corrupts the same set of parties as done by $\mathcal{Z}$ in its game with $\mathcal{C}$. Further, $\mathcal{A}$ forwards the public key-secret key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ for the corrupt parties it receives from $\mathcal{Z}$ to the challenger and the public keys pk_i for the honest parties from $\mathcal{C}$ to $\mathcal{Z}$. $\mathcal{A}$ also forwards to $\mathcal{C}$ the set of messages to be encrypted along with the randomness for the ones encrypted by the adversary, received from $\mathcal{Z}$. Similarly, it forwards the ciphertexts received from $\mathcal{C}$ to $\mathcal{Z}$. Finally, $\mathcal{A}$ sends the circuit corresponding to the evaluation of polynomial $\mathsf{V}(\cdot)$ to $\mathcal{C}$ and receives a set of partial decryptions on behalf of each honest party which it forwards to $\mathcal{A}$. It continues interacting with $\mathcal{Z}$ as in Hybrid_1 in the rest of its interaction. It is easy to see that if $\mathcal{C}$ sent honestly generated partial decryptions, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds

to Hybrid_1 and if the partial decryptions were simulated, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_2. Thus, if $\mathcal{Z}$ can distinguish between the two hybrids with non-negligible probability ϵ, $\mathcal{A}$ can break the simulation security of the TAHE scheme with the same probability ϵ which is a contradiction. □

Lemma 7.5. Hybrid_2 *is statistically close to* Hybrid_3.

Proof. The only difference between the two hybrids is the way the polynomial $\mathsf{V}(\cdot)$ is computed. In Hybrid_3, $\mathsf{V}(\mathrm{x}) = \mathsf{p}_I(\mathrm{x}) \cdot U(\mathrm{x})$ where $U(\cdot)$ is a uniformly random polynomial of degree $2T+2$. In Hybrid_2,[8]

$$\begin{aligned}
\mathsf{V}(\mathrm{x}) =& \sum_{i=1}^{n} \left(\mathsf{p}'_i(\mathrm{x}) \cdot \left(\mathsf{R}_1(\mathrm{x}) + \ldots + \mathsf{R}_{i-1}(\mathrm{x}) + \widetilde{\mathsf{R}}_i(\mathrm{x}) + \mathsf{R}_{i+1}(\mathrm{x}) \ldots + \mathsf{R}_n(\mathrm{x}) \right) \right) \\
=& \mathsf{p}_I(\mathrm{x}) \cdot \sum_{i=1}^{n} \left(\mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\mathsf{R}_1(\mathrm{x}) + \ldots + \mathsf{R}_{i-1}(\mathrm{x}) + \widetilde{\mathsf{R}}_i(\mathrm{x}) + \mathsf{R}_{i+1}(\mathrm{x}) \ldots + \mathsf{R}_n(\mathrm{x}) \right) \right) \\
=& \mathsf{p}_I(\mathrm{x}) \cdot \left[\sum_{i \in \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\widetilde{\mathsf{R}}_i(\mathrm{x}) + \sum_{j \in \mathcal{S}^*} \mathsf{R}_j(\mathrm{x}) \right) + \sum_{i \in \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\sum_{j \in [n]\setminus \mathcal{S}^*} \mathsf{R}_j(\mathrm{x}) \right) \right. \\
& \left. + \sum_{i \in [n]\setminus \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\widetilde{\mathsf{R}}_i(\mathrm{x}) + \sum_{j \in [n]} \mathsf{R}_i(\mathrm{x}) \right) \right] \\
=& \mathsf{p}_I(\mathrm{x}) \cdot [A(\mathrm{x}) + B(\mathrm{x})]
\end{aligned}$$

where

$$\begin{aligned}
A(\mathrm{x}) &= \sum_{i \in \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\widetilde{\mathsf{R}}_i(\mathrm{x}) + \sum_{j \in \mathcal{S}^*} \mathsf{R}_j(\mathrm{x}) \right) \\
B(\mathrm{x}) &= \left(\sum_{i \in \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\sum_{j \in [n]\setminus \mathcal{S}^*} \mathsf{R}_i(\mathrm{x}) \right) \right) + \sum_{i \in [n]\setminus \mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\widetilde{\mathsf{R}}_i(\mathrm{x}) + \sum_{j \in [n]} \mathsf{R}_j(\mathrm{x}) \right).
\end{aligned}$$

Note that for all $i \in [n]$, $\mathsf{Deg}(\mathsf{p}_{i\setminus I}(\mathrm{x})) \leq T$, $\mathsf{Deg}(\mathsf{p}'_{i\setminus I}(\mathrm{x})) \leq T+1$, $\mathsf{Deg}(\mathsf{R}_i(\mathrm{x})) = \mathsf{Deg}(\widetilde{\mathsf{R}}_i(\mathrm{x})) = T+1$. Thus, $\mathsf{Deg}(A(\mathrm{x})) = \deg(B(\mathrm{x})) = 2T+2$. Now, suppose we prove that $B(\mathrm{x})$ is statistically close to a uniformly random polynomial of degree $2T+2$, then in Hybrid_2,

$$\begin{aligned}
\mathsf{V}(\mathrm{x}) &\equiv \mathsf{p}_I(\mathrm{x}) \cdot \Big[A(\mathrm{x}) + U_1(\mathrm{x}) \Big] \\
&\equiv \mathsf{p}_I(\mathrm{x}) \cdot U_2(\mathrm{x})
\end{aligned}$$

where $U_1(\cdot)$ and $U_2(\cdot)$ are uniformly random polynomials of degree $2T+2$. Thus, $\mathsf{V}(\mathrm{x})$ in Hybrid_2 is statistically close to the distribution of $\mathsf{V}(\mathrm{x})$ in Hybrid_3. Therefore, the only remaining step is to prove the below claim.

Claim 7.6. $B(\mathrm{x})$ *is statistically close to a uniformly random polynomial of degree* $2T+2$.

[8] Here, $\mathsf{p}'_{i\setminus I}(\mathrm{x}) = \frac{\mathsf{p}'_i(\mathrm{x})}{\mathsf{p}_I(\mathrm{x})} = \mathsf{p}_{i\setminus I}(\mathrm{x}) \cdot (\mathrm{x} - r_i)$.

Proof.

$$B(\mathrm{x}) = \left(\sum_{i\in\mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\sum_{j\in[n]\setminus\mathcal{S}^*} \mathsf{R}_i(\mathrm{x})\right)\right) + \sum_{i\in[n]\setminus\mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot \left(\widetilde{\mathsf{R}}_i(\mathrm{x}) + \sum_{j\in[n]} \mathsf{R}_j(\mathrm{x})\right)$$
$$= \left(\sum_{i\in\mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x})\right) \cdot \widetilde{U}(\mathrm{x}) + \sum_{i\in[n]\setminus\mathcal{S}^*} \left(\mathsf{p}'_{i\setminus I}(\mathrm{x}) \cdot U_i(\mathrm{x})\right).$$

where $\widetilde{U}(\mathrm{x}), U_i(\mathrm{x})$ are uniformly random polynomials of degree $T+1$.

Note that by the definition of the set intersection I,

$$\gcd(\mathsf{p}_{1\setminus I}(\mathrm{x}), \ldots, \mathsf{p}_{n\setminus I}(\mathrm{x})) = 1.$$

By a direct invocation of Lemma 2.4,

$$\gcd\left(\sum_{i\in\mathcal{S}^*} \mathsf{p}'_{i\setminus I}(\mathrm{x}), \mathsf{p}'_{j_1\setminus I}(\mathrm{x}), \mathsf{p}'_{j_2\setminus I}(\mathrm{x}), \ldots, \mathsf{p}'_{j_t\setminus I}(\mathrm{x})\right) = 1$$

except with negligible probability where $j_1, \ldots, j_t$ denotes the indices of the honest parties. Then, by Lemma 2.5, we can conclude that $B(\mathrm{x})$ is statistically close to a uniformly random polynomial of degree $2T+2$. □

□

Lemma 7.7. *Assuming the semantic security of the threshold additive homomorphic encryption scheme,* Hybrid_3 *is computationally indistinguishable from* Hybrid_4.

Proof. The only difference between the two hybrids is that in Hybrid_3, the simulator SimHyb generates the encryptions of the TAHE scheme on behalf of the honest parties as in the real world while in Hybrid_4, they are generated as encryptions of 0. We now show that if there exists an adversarial environment $\mathcal{Z}$ that can distinguish between these two hybrids with some non-negligible probability ϵ, we will come up with a reduction $\mathcal{A}$ that can break the semantic security of the TAHE scheme.

$\mathcal{A}$ interacts with a challenger $\mathcal{C}$ in the semantic security game for TAHE and with the environment $\mathcal{Z}$ in the game between Hybrid_3 and Hybrid_4. $\mathcal{A}$ corrupts the same set of parties as done by $\mathcal{Z}$ in its game with $\mathcal{C}$. Further, $\mathcal{A}$ forwards the public key-secret key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ for the corrupt parties it receives from $\mathcal{Z}$ to the challenger and the public keys pk_i for the honest parties from $\mathcal{C}$ to $\mathcal{Z}$. $\mathcal{A}$ also forwards the pair of 0 and the set of honestly generated plaintexts to be encrypted, to the challenger and receives back a ciphertext for each of them which it uses in its interaction with $\mathcal{Z}$. It continues interacting with $\mathcal{Z}$ as in Hybrid_3 in the rest of its interaction. It is easy to see that if $\mathcal{C}$ sent honestly generated ciphertexts, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_3 and if the ciphertexts were generated as encryptions of 0, the interaction between $\mathcal{A}$ and $\mathcal{Z}$ exactly corresponds to Hybrid_4. Thus, if $\mathcal{Z}$ can distinguish between the two hybrids with non-negligible probability ϵ, $\mathcal{A}$ can break the semantic security of the TAHE scheme with the same probability ϵ which is a contradiction. □

References

[AIK05] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Computationally private randomizing polynomials and their applications. In *CCC*, 2005.

[BDP21] Pedro Branco, Nico Döttling, and Sihang Pu. Multiparty cardinality testing for threshold private set intersection. In *PKC*, 2021.

[Ben94] Josh Benaloh. Dense probabilistic encryption. May 1994.

[BFK+19] Saikrishna Badrinarayanan, Rex Fernando, Venkata Koppula, Amit Sahai, and Brent Waters. Output compression, mpc, and io for turing machines. In *ASIACRYPT*, 2019.

[BGG+18] Dan Boneh, Rosario Gennaro, Steven Goldfeder, Aayush Jain, Sam Kim, Peter M. R. Rasmussen, and Amit Sahai. Threshold cryptosystems from threshold fully homomorphic encryption. In *CRYPTO*, 2018.

[BGY80] Richard P. Brent, Fred G. Gustavson, and David Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of padé approximants. *J. Algorithms*, 1(3):259–295, 1980.

[BJMS20] Saikrishna Badrinarayanan, Aayush Jain, Nathan Manohar, and Amit Sahai. Threshold multi-key fhe and applications to round-optimal mpc. 2020.

[BO15] Mark Braverman and Rotem Oshman. On information complexity in the broadcast model. In *PODC*, 2015.

[BPSW07] Justin Brickell, Donald E Porter, Vitaly Shmatikov, and Emmett Witchel. Privacy-preserving remote diagnostics. In *CCS*, 2007.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, 2001.

[CDN01] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In *EUROCRYPT*, 2001.

[CM20] Melissa Chase and Peihan Miao. Private set intersection in the internet setting from lightweight oblivious PRF. In *CRYPTO*, 2020.

[DCT10] Emiliano De Cristofaro and Gene Tsudik. Practical private set intersection protocols with linear complexity. In *FC*, 2010.

[DCW13] Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: an efficient and scalable protocol. In *CCS*, 2013.

[DIK+08] Ivan Damgård, Yuval Ishai, Mikkel Krøigaard, Jesper Buus Nielsen, and Adam D. Smith. Scalable multiparty computation with nearly optimal work and resilience. In *CRYPTO*, 2008.

[FH96] Matthew K. Franklin and Stuart Haber. Joint encryption and message-efficient secure computation. *J. Cryptology*, 1996.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *EUROCRYPT*, 2004.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, 1984.

[GJR10] Elena Grigorescu, Kyomin Jung, and Ronitt Rubinfeld. A local decision test for sparse polynomials. *Information Processing Letters*, 2010.

[GN19] Satrajit Ghosh and Tobias Nilges. An algebraic approach to maliciously secure private set intersection. In *EUROCRYPT*, 2019.

[GS19a] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. In *CRYPTO*, 2019.

[GS19b] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection, 2019. ia.cr/2019/175.

[HFH99] Bernardo A. Huberman, Matt Franklin, and Tad Hogg. Enhancing privacy and trust in electronic communities. In *In Proc. of the 1st ACM Conference on Electronic Commerce*, 1999.

[HOS17] Per A. Hallgren, Claudio Orlandi, and Andrei Sabelfeld. Privatepool: Privacy-preserving ridesharing. In *CSF*, 2017.

[HV17] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In *PKC*, 2017.

[HW15] Pavel Hubáček and Daniel Wichs. On the communication complexity of secure function evaluation with long output. In *ITCS*, 2015.

[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. 2017. ia.cr/2017/738.

[KKRT16] Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In *CCS*, 2016.

[KMP+17] Vladimir Kolesnikov, Naor Matania, Benny Pinkas, Mike Rosulek, and Ni Trieu. Practical multi-party private set intersection from symmetric-key techniques. In *CCS*, 2017.

[KMWF07] Eike Kiltz, Payman Mohassel, Enav Weinreb, and Matthew Franklin. Secure linear algebra using linearly recurrent sequences. In *TCC*, 2007.

[KS05] Lea Kissner and Dawn Song. Privacy-preserving set operations. In *CRYPTO*, 2005.

[MPR+20] Peihan Miao, Sarvar Patel, Mariana Raykova, Karn Seth, and Moti Yung. Two-sided malicious security for private intersection-sum with cardinality. In *CRYPTO*, 2020.

[MTZ03] Yaron Minsky, Ari Trachtenberg, and Richard Zippel. Set reconciliation with nearly optimal communication complexity. *IEEE Transactions on Information Theory*, 2003.

[MZ17] Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *IEEE S and P*, 2017.

[NMH+10] Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov. Botgrep: Finding p2p bots with structured graph analysis. In *USENIX security symposium*, 2010.

[OOS16] Michele Orrù, Emmanuela Orsini, and Peter Scholl. Actively secure 1-out-of-n OT extension with application to private set intersection. In *CT-RSA*, 2016.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, 1999.

[PRTY19] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. Spot-light: Lightweight private set intersection from sparse ot extension. In *CRYPTO*, 2019.

[PRTY20] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from paxos: Fast, malicious private set intersection. In *EUROCRYPT*, 2020.

[PSSZ15] Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. Phasing: Private set intersection using permutation-based hashing. In *USENIX*, 2015.

[PSWW18] Benny Pinkas, Thomas Schneider, Christian Weinert, and Udi Wieder. Efficient circuit-based PSI via cuckoo hashing. In *EUROCRYPT*, 2018.

[PSZ14] Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on ot extension. In *USENIX*, 2014.

[RR17] Peter Rindal and Mike Rosulek. Malicious-secure private set intersection via dual execution. In *CCS*, 2017.

[TPKC07] Juan Ramón Troncoso-Pastoriza, Stefan Katzenbeisser, and Mehmet Celik. Privacy preserving error resilient dna searching through oblivious automata. In *CCS*, 2007.

[TY90] Klaus Thull and Chee Yap. A unified approach to hgcd algorithms for polynomials and integers. *Manuscript*, 1990.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, 1986.

[ZC18] Yongjun Zhao and Sherman SM Chow. Can you find the one for me? privacy-preserving matchmaking via threshold psi. 2018. ia.cr/2018/184.

A Linear Algebra Proofs

A.1 Proof of Lemma 2.5

This lemma is a generalization of Lemma 2 in the work of Kissner and Song [KS05]. They proved the lemma for two polynomials and we generalize it to multiple polynomials. We first extend [KS05, Lemma 2] for two polynomials to the case where p_1 and p_2 do not necessarily have the same degree. Formally, we first prove the following lemma.

Lemma A.1. *Let $\mathbb{F}$ be a finite field of prime order q. Fix any $n = O(\mathsf{poly}(\lambda))$. For any two polynomials $p_1(x), p_2(x) \in \mathbb{F}[x]$ of degrees α_1 and α_2 respectively, if $R_1(x) = \sum_{j=0}^{\beta_1} r_{1,j}x^j, R_2(x) = \sum_{j=0}^{\beta_2} r_{2,j}x^j \in \mathbb{F}[x]$ where $\alpha_1 + \beta_1 = \alpha_2 + \beta_2 \geq \alpha_1 + \alpha_2$ and $r_{i,j} \xleftarrow{\$} \mathbb{F}$ are sampled independently and uniformly at random. Let $S(x) = p_1(x) \cdot R_1(x) + p_2(x) \cdot R_2(x)$. Then $S(x) = \gcd(p_1(x), p_2(x)) \cdot U(x)$, where $U(x) = \sum_{j=0}^{\alpha_1+\beta_1-\gamma} u_j x^j$, in which γ is the degree of $\gcd(p_1(x), p_2(x))$, u_j's are distributed uniformly and independently over $\mathbb{F}$.*

Proof. Let $g = \gcd(p_1, p_2)$, and let $p_1(x) = g(x) \cdot q_1(x), p_2(x) = g(x) \cdot q_2(x)$. We know that g has degree γ, hence q_1, q_2 has degrees $\alpha_1 - \gamma$ and $\alpha_2 - \gamma$, respectively. In addition, $\gcd(q_1, q_2) = 1$. Since

$$S(x) = p_1(x) \cdot R_1(x) + p_2(x) \cdot R_2(x) = g(x) \cdot (q_1(x) \cdot R_1(x) + q_2(x) \cdot R_2(x)),$$

we only need to show that $q_1(x) \cdot R_1(x) + q_2(x) \cdot R_2(x) = \sum_{j=0}^{\alpha_1+\beta_1-\gamma} u_j x^j$ where u_j's are distributed uniformly and independently over $\mathbb{F}$.

Given any fixed polynomial U of degree $\alpha_1 + \beta_1 - \gamma$, we calculate the number of (R_1, R_2) pairs such that $q_1 \cdot R_1 + q_2 \cdot R_2 = U$. Let us assume for this particular U there exists at least one pair of (R_1, R_2) such that $q_1 \cdot R_1 + q_2 \cdot R_2 = U$. Then for any other satisfying pair (R_1', R_2'), we have that

$$\begin{aligned} q_1 \cdot (R_1 - R_1') + q_2 \cdot (R_2 - R_2') &= 0, \\ q_1 \cdot (R_1 - R_1') &= q_2 \cdot (R_2' - R_2). \end{aligned}$$

Since $\gcd(q_1, q_2) = 1$ and $\mathbb{F}$ is a finite field of prime order, we have $q_2 | (R_1 - R_1')$ and $q_1 | (R_2' - R_2)$. Let $R_1 - R_1' = q_2 \cdot h$, then $R_2' - R_2 = q_1 \cdot h$, where h is a polynomial with degree

$$\begin{aligned} d =& \deg(U) - \deg(q_1) - \deg(q_2) \\ =& (\alpha_1 + \beta_1 - \gamma) - (\alpha_1 - \gamma) - (\alpha_2 - \gamma) \\ =& \beta_1 - \alpha_2 + \gamma. \end{aligned}$$

Every pair of satisfying (R_1', R_2') decides a unique polynomial h and every polynomial h corresponds to a unique satisfying pair (R_1', R_2'). Hence the total number of (R_1', R_2') pairs equals the total number of degree-d polynomial h in $\mathbb{F}[x]$, which is $q^{\beta_1 - \alpha_2 + \gamma + 1}$.

Now consider the polynomial $(q_1 \cdot R_1 + q_2 \cdot R_2)$. We know that for any fixed polynomial U as a possible result, there are $q^{\beta_1 - \alpha_2 + \gamma + 1}$ pairs of (R_1, R_2) that lead to the result. Since there are a total number of $q^{\beta_1 + \beta_2 + 2}$ possible pairs of (R_1, R_2), the total number of possible resulting U is

$$\frac{q^{\beta_1 + \beta_2 + 2}}{q^{\beta_1 - \alpha_2 + \gamma + 1}} = q^{\alpha_2 + \beta_2 - \gamma + 1} = q^{\alpha_1 + \beta_1 - \gamma + 1},$$

which is exactly the total number of U. Therefore, for randomly generated R_1, R_2, each possible polynomial with degree $\alpha_1 + \beta_1 - \gamma$ will be the result with equal probability. □

Given Lemma A.1,

$$\begin{aligned}
&p_1(x) \cdot R_1(x) + p_2(x) \cdot R_2(x) \cdots + p_n(x) \cdot R_n(x) \\
&= \gcd(p_1(x), p_2(x)) \cdot U_2(x) + p_3(x) \cdot R_3(x) \cdots + p_n(x) \cdot R_n(x) \\
&= \gcd(p_1(x), p_2(x), p_3(x)) \cdot U_3(x) + p_4(x) \cdot R_4(x) \cdots + p_n(x) \cdot R_n(x) \\
&= \ldots \\
&= \gcd(p_1(x), \ldots, p_n) \cdot U_n(x),
\end{aligned}$$

where $U_i(x) = \sum_{j=0}^{\alpha+\beta-\gamma_i} u_{i,j} x^j$, in which γ_i is the degree of $\gcd(p_1(x), \ldots, p_i(x))$, $u_{i,j}$'s are distributed uniformly and independently over $\mathbb{F}$. Since $\gcd(p_1, \ldots, p_n) = 1$, we have $\gamma_n = 0$. Hence $\sum_{i=1}^{n}(p_i(x) \cdot R_i(x)) = \sum_{j=0}^{\alpha+\beta} u_j x^j$, then u_j's are distributed uniformly and independently over $\mathbb{F}$.

A.2 Proof of Lemma 2.4

Fix any i such that $1 \leq i < n$. Let's denote $p_i^*(x) := p_1'(x) + \ldots + p_i'(x)$. Observe that $\gcd(p_i^*, p_{i+1}', \ldots, p_n') = \gcd(p_i^*, \gcd(p_{i+1}', \ldots, p_n')) = \gcd(p_i^*, \gcd(p_{i+1}, \ldots, p_n))$ since the probability that for $j > i$, r_j is a root of p_k', where $k > i, k \neq j$ is negligible.

Consider any root v of $\gcd(p_{i+1}, \ldots, p_n)$. We now analyze the event: $\Pr_{r_j}[p_i^*(v) = 0]$. First, note that since $\gcd(p_1, \ldots, p_n) = 1$, there exists $k \leq i$ s.t. $(x - v) \nmid p_k(x)$. Further, since r_k is picked uniformly at random, $\Pr_{r_k}[r_k = v] \leq \mathsf{negl}(\lambda)$. Therefore, except with negligible probability, there exists $k <= i$ s.t. $(x - v) \nmid p_k'(x)$.

As such it must hold that

$$\begin{aligned}
p_1'(v) + \ldots + p_{k-1}'(v) + p_{k+1}'(v) + \ldots + p_i'(v) &= -p_k'(v), \\
\frac{p_1'(v) + \ldots + p_{k-1}'(v) + p_{k+1}'(v) + \ldots + p_i'(v)}{p_k(v)} + v &= r_k.
\end{aligned}$$

Since $r_k \in \mathbb{F}$ is picked uniformly at random, the probability of this event is $1/q = \mathsf{negl}(\lambda)$. Taking a union bound over all the roots of $\gcd(p_{i+1}, \ldots, p_n)$ yields

$$\Pr_{r_j}[\gcd(p_i^*, \gcd(p_{i+1}, \ldots, p_n)) \neq 1] \leq \mathsf{negl}(\lambda)$$

and this completes the proof.

B Threshold PSI for Functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$

In this section, we give a multiparty threshold PSI protocol for the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$. Recall that in a multi-party threshold PSI protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ defined in Figure 1, each party P_i wishes to learn the intersection of all the sets if $|S_i \setminus I| \leq T$, that is, if the size of its own set minus the intersection is less than the threshold T. Our protocol is almost identical to the protocol from Section 7 for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ with the only difference being in the first step of the protocol, we run the multiparty private intersection cardinality testing protocol for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ instead of $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$. The rest of the protocol is the same. We elaborate on the details here for completeness.

As before, we formally prove the following theorem:

Theorem B.1. *Assuming the existence of threshold additive homomorphic encryption, protocol* $\Pi_{\mathsf{TPSI\text{-}int}}$ *(Figure 10) securely realizes* $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ *in the* $\mathcal{F}_{\mathsf{CTest\text{-}int}}$*-hybrid model in the star network communication model. Our protocol is secure against a semi-honest adversary that can corrupt up to* $(n-1)$ *parties.*

The protocol runs in a constant number of rounds and the communication complexity is $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ in the $\mathcal{F}_{\mathsf{CTest\text{-}int}}$-hybrid model. We then instantiate the $\mathcal{F}_{\mathsf{CTest\text{-}int}}$-hybrid with the protocol based on TFHE from Section 5.1 that has constant round complexity and $O(n \cdot T \cdot \mathsf{poly}(\lambda))$ communication complexity. Formally, we get the following corollary:

Corollary B.2. *Assuming TFHE with distributed setup, protocol* $\Pi_{\mathsf{TPSI\text{-}int}}$ *(Figure 10) securely realizes* $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ *in the star network communication model with communication complexity* $O(n \cdot T \cdot \mathsf{poly}(\lambda))$*.*

Protocol. Our protocol, which is almost identical to the protocol from Section 7 is described in Figure 10. The change is highlighted in red.

Correctness. If $|S_i \setminus I| > T$, then the protocol terminates after the first step - the private intersection cardinality testing. Note that since $|S_i|$ is the same for all i, the protocol either terminates for all the parties or proceeds for all the parties. If, on the other hand, $|S_i \setminus I| \leq T$, the rest of the correctness analysis is identical to the one performed for the protocol in Section 7.

Communication Complexity. The communication complexity analysis is identical to the one performed for the protocol in Section 7.

Security Proof. The proof is identical to the proof of Theorem 7.1 from Section 7.

C Other Communication Models

Throughout the paper we focus on the communication lower bounds in point-to-point networks and design protocols in the star network (which can be implemented on point-to-point network). In this section, we initiate the study of multiparty threshold PSI on networks with broadcast channels. We give a new lower bound for the functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$ in the broadcast model of communication. Note that the lower bound of $\Omega(n \cdot T)$ from Section 4 for the point-to-point network does not necessarily apply to the broadcast setting.

At a high level, we reduce the problem of multiparty set disjointness to multiparty threshold PSI for the ideal functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$. In the multiparty set disjointness problem, there are n parties each holding a set $X_i \subseteq [N]$. The parties' task is to determine if $\bigcap_{i=1}^{n} X_i = \emptyset$. It is shown by Braverman and Oshman [BO15] that the communication complexity of multiparty set disjointness in the broadcast model is $\Omega(N \log n + n)$. Given this result, we prove the communication lower bound for multiparty threshold PSI:

Theorem C.1. *For any multiparty threshold private set intersection protocol for functionality* $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$*, the communication complexity in the broadcast model is* $\Omega(T \log n + n)$ *where* n *is the number of parties and* T *is the threshold parameter.*

Parameters: Parties $P_1, \ldots, P_n$. Each party has a set of m elements. Threshold $T \in \mathbb{N}$. $\mathbb{F}$ is a finite field where $|\mathbb{F}| = \Omega(2^\lambda)$.

Inputs: Party P_i has an input set $S_i = \{a_1^i, \ldots, a_m^i\}$ where $a_j^i \in \mathbb{F}$ for all $j \in [m]$.

Output: Each party P_i receives the set intersection $I = \bigcap_{i=1}^n S_i$ if and only if $|(\bigcup_{i=1}^n S_i) \setminus I| \leq T$.

Protocol:

1. **Private Intersection Cardinality Testing.** The parties invoke $\mathcal{F}_{\mathsf{CTest\text{-}diff}}$ on inputs $S_1, \ldots, S_n$ and receive back $w \in \{\mathsf{similar}, \mathsf{different}\}$. If $w = \mathsf{different}$ then all parties output $\perp$.
2. **TAHE Key Generation.** Each party P_i generates $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{TAHE.DistSetup}(1^\lambda, i)$ and sends pk_i to P_1. Then P_1 sends $\mathsf{pk} = (\mathsf{pk}_1 \| \ldots \| \mathsf{pk}_n)$ to all the other parties.
3. **Evaluations of Random Polynomial.** In this phase the parties will evaluate a polynomial
$$\mathsf{V}(\mathrm{x}) = \sum_{i=1}^{n} \left(\mathsf{p}_i'(\mathrm{x}) \cdot \left(\mathsf{R}_1(\mathrm{x}) + \ldots + \mathsf{R}_{i-1}(\mathrm{x}) + \widetilde{\mathsf{R}}_i(\mathrm{x}) + \mathsf{R}_{i+1}(\mathrm{x}) \ldots + \mathsf{R}_n(\mathrm{x}) \right) \right)$$
for $x \in [3T+4]$ where the terms are defined as follows.
   (a) Each party P_i defines $\mathsf{p}_i(\mathrm{x}) = \prod_{j=1}^m (\mathrm{x} - a_j^i)$ and $\mathsf{p}_i'(\mathrm{x}) = \mathsf{p}_i(\mathrm{x}) \cdot (\mathrm{x} - r_i)$ where $r_i \xleftarrow{\$} \mathbb{F}$.
   (b) Each party P_i uniformly samples $\mathsf{R}_i, \widetilde{\mathsf{R}}_i \xleftarrow{\$} \mathbb{F}[\mathrm{x}]$ of degree $T+1$, computes $\mathsf{R}_i(x)$ for $x \in [3T+4]$ and sends encrypted $[\![\mathsf{R}_i(x)]\!]$ to P_1.
   (c) For each $i \in [n], x \in [3T+4]$, party P_1 sends $[\![e_{i,x}]\!] = \left[\!\!\left[\sum_{j \in [n] \setminus \{i\}} \mathsf{R}_j(x) \right]\!\!\right]$ to P_i.
   (d) For each $x \in [3T+4]$, each party P_i sends $[\![v_{i,x}]\!] = \left[\!\!\left[\mathsf{p}_i'(x) \cdot \left(e_{i,x} + \widetilde{\mathsf{R}}_i(x) \right) \right]\!\!\right]$ to P_1.
   (e) For each $x \in [3T+4]$, P_1 sends $[\![v_x]\!] = [\![\sum_{i=1}^n v_{i,x}]\!]$ to all P_i.
   (f) For each $x \in [3T+4]$, each party P_i sends $[\![v_x : \mathsf{sk}_i]\!] \leftarrow \mathsf{TAHE.PartialDec}(\mathsf{sk}_i, [\![v_x]\!])$ to P_1.
   (g) For each $x \in [3T+4]$, P_1 sends $\mathsf{V}(x) = \mathsf{TAHE.Combine}(\mathsf{pk}, \{[\![v_x : \mathsf{sk}_i]\!]\}_{i \in [n]})$ to all P_i.
4. **Computing Set Intersection.** Each party P_i does the following:
   (a) Interpolate $\mathsf{q}_i(\mathrm{x})$ to be the degree $3T+3$ rational polynomial such that $\mathsf{q}_i(x) = \frac{\mathsf{V}(x)}{\mathsf{p}_i'(x)}$ for $x \in [3T+4]$ and the gcd of the numerator and denominator is 1. Let D_i be the roots of the denominator of $\mathsf{q}_i(\mathrm{x})$.
   (b) Output the set intersection $I = S_i \setminus D_i$.

Figure 10: Multi-party threshold PSI protocol $\Pi_{\mathsf{TPSI\text{-}int}}$ for functionality $\mathcal{F}_{\mathsf{TPSI\text{-}int}}$.

Proof. We prove the theorem by giving a reduction from multiparty set disjointness to multiparty threshold PSI. Assume there is a multiparty threshold PSI protocol $\Pi_{\mathsf{TPSI\text{-}int}}$ that achieves communication complexity $o(T \log n + n)$ in the broadcast model, then we can solve multiparty set disjointness with communication complexity $o(N \log n + n)$, which leads to a contradiction. We show the reduction in the following.

Given an instance of multiparty set disjointness where there are n parties each holding a set $X_i \subseteq [N]$, we construct an instance of multiparty threshold PSI as follows. There are n parties. Each party P_i has a set of size N. We set the sets as $S_i := X_i \cup U_i$, where $|S_i| = N$, and U_i consists of unique elements that can only appear in S_i. Notice that $\bigcap_{i=1}^n X_i \neq \emptyset$ if and only if $\bigcap_{i=1}^n S_i \neq \emptyset$, namely $|S_i - \bigcap_{i=1}^n S_i| \leq (N-1)$. The n parties in the multiparty set disjointness problem run the multiparty threshold PSI protocol $\Pi_{\mathsf{TPSI\text{-}int}}$ where each party P_i inputs set S_i and the threshold is

set to be $T := N - 1$. If $\Pi_{\mathsf{TPSI\text{-}int}}$ outputs the set intersection, then the parties output $\bigcap_{i=1}^{n} X_i \neq \emptyset$; otherwise the parties output $\bigcap_{i=1}^{n} X_i = \emptyset$. □

We leave further exploration in the broadcast model including a lower bound for the other functionality $\mathcal{F}_{\mathsf{TPSI\text{-}diff}}$ as well as designing more efficient protocols as interesting open problems.