ARTIFACT EVALUATED
usenix ASSOCIATION
AVAILABLE

# Private Set Intersection and other Set Operations in the Third Party Setting

Foo Yee Yeo
*Seagate Technology*

Jason H. M. Ying
*Seagate Technology*

## Abstract

We present a collection of protocols to perform privacy-preserving set operations in the third-party private set intersection (PSI) setting. This includes several protocols for multi-party third party PSI. In this model, there are multiple input parties (or clients) each holding a private set of elements and the receiver is an external party (termed as third-party) with no inputs. Multi-party third party PSI enables the receiver to learn only the intersection result of all input clients' private sets while revealing nothing else to the clients and the receiver. Our solutions include constructions that are provably secure against an arbitrary number of colluding parties in the semi-honest model. Additionally, we present protocols for third-party private set difference and private symmetric difference, whereby the learned output by the inputless third-party is the set difference and symmetric difference respectively of two other input parties, while preserving the same privacy guarantees. The motivation in the design of these protocols stems from their utilities in numerous real-world applications. We implemented our protocols and conducted experiments across various input and output set sizes.

## 1 Introduction

Conventional private set intersection (PSI) allows two parties, each holding a private set of items, to learn the intersection of the sets and nothing else. Enormous strides have been made to advance the progress of two-party PSI over the years [2,4,11–15,17,21–24,33,35,38,40,41,47–55,62,65,66]. Multi-party PSI can be regarded as a generalization of conventional two-party PSI, whereby the intersection functionality is over sets held by multiple parties. Many approaches for two-party PSI do not extend to the multi-party setting. As such, other techniques have been developed to provide such solutions. The initial multi-party PSI protocols proposed by Freedman et al. [20] and Kissner & Song [37] utilize an additive homomorphic encryption such as the Paillier cryptosystem [45] which involves expensive public key operations. Since then, much progress have been made in the development of more efficient multi-party PSI protocols [3,6,10,27,32,34,39,44].

Recently, Yeo and Ying [63,64] introduced a new model for PSI, known as third party PSI (TP-PSI), in which the intersection of sets held by two different parties are privately computed and revealed to a third-party $Q$. This problem admits a natural generalization to more parties; namely, given datasets $S_1,\ldots,S_N$ held by $N$ different parties $P_1,\ldots,P_N$, we would like to privately compute the intersection of these datasets and reveal the result only to an inputless third-party $Q$. We shall call this problem multi-party third party PSI.

A naive construction for multi-party third party PSI is to apply a multi-party PSI solution whereby $Q$ is assigned a set containing the entire universe of possible elements. However, this is not feasible except when the universe is small, since the computation and communication cost of such a solution will at least be linear in the size of the universe. On the other hand, an efficient solution should depend polynomially on the sizes of the input sets and polylogarithmically on the size of the universe.

Multi-party third party PSI brings unique challenges that are different from those in conventional multi-party PSI. A significant difficulty to achieving efficient third party PSI is due to the fact that the third-party has no private information that allows him to constrain the possible elements in the intersection. In contrast, in the conventional setting, the receiver can use his own private input set to help determine the intersection since elements in the intersection lie in every private input set. This heavily restricts the possible intersection elements from the universe of all elements to a much smaller set, and gives rise to techniques of designing the protocol that is not possible in the third-party setting.

The solutions presented in [63] and [64] for third party PSI with 2 input parties rely on tools such as commutative ciphers and key exchange, and do not generalize to the multi-party setting. In this paper, we present novel solutions to this problem which works regardless of the number $N$ of input parties, including some solutions that are secure in the semi-honest model against any number of colluding parties.

There are numerous applications for multi-party third party PSI. For example, it can be applied to investigate cyber intrusions of several organizations suspected to be performed by the same attacker. In such an event, authorities can perform an initial investigation by engaging in a multi-party TP-PSI (MP-TP-PSI) with the affected parties whereby each input is a list of suspicious IP addresses captured within a specified period of interest.

There are two natural approaches to handle scenarios such as the above. The simplest solution is for each organization to submit a list of suspicious IP addresses to an investigating entity. The investigating entity can then simply perform a multi-set intersection of all lists in the clear to obtain the outcome. A critical drawback of this straightforward method is that collective lists of all IP addresses provided by the organizations are exposed to the investigating authority which results in an unnecessary breach of privacy. An alternative approach which does not expose the entire list of IP addresses to the investigation entity is for the participating parties to designate one party to be the receiver in a multi-party PSI setting. Existing conventional multi-party PSI protocols can then be applied to enable the designated receiver to obtain the outcome which can then be forwarded to the investigative entity. However, this is once again not ideal since the receiving party will then have access to the intersection list of IP addresses from all the other organizations. A motivation to develop MP-TP-PSI stems from achieving a solution to satisfy such tight privacy constraints.

Another use case application arises in marketing whereby a group of shop owners intends to collaboratively launch a promotional campaign. The input parties are the shop owners who each has a list of customers while the marketing agency is the third party. The marketing agency is able to obtain the list of common customers to target from the intersection output and the shop owners maintain the confidentiality of their customers from the rest of the competitors. More generally, MP-TP-PSI can be applied in settings whereby an inputless external party seeks to obtain only the necessary information of common items from the set of all other parties in a privacy-preserving manner.

We provide a series of protocols to achieve the multi-set intersection functionality in the third-party setting. We first show in Section 3 that it is feasible to construct protocols having low communication overhead with the assumption that only certain types of collusion are present. We proceed to present two of our main protocols in Section 4 and Section 5 which are secure against arbitrary collusion. Protocol 2 incurs a slightly lower communication cost compared to Protocol 3, while Protocol 3 has a significantly lower computational cost compared to Protocol 2.

We also introduce third-party private set difference (TP-PSD) and third-party private symmetric difference (TP-PSymD) and present protocols to achieve these desired functionalities. In these settings, parties $P_1$ and $P_2$ have datasets $S_1$ and $S_2$ respectively. Private set difference is the functionality of privately computing $S_1 \setminus S_2$, i.e. the set of elements that lie in $S_1$ but not in $S_2$, while symmetric difference is the functionality of privately computing $S_1 \triangle S_2$. In both models, the receiver is an inputless third-party $Q$.

In the conventional two-party setting, private set difference is equivalent to PSI with $P_1$ as receiver, since $S_1 \setminus S_2 = S_1 \setminus (S_1 \cap S_2)$, and also equivalent to private set union with $P_2$ as receiver since $S_1 \setminus S_2 = (S_1 \cup S_2) \setminus S_2$. However, there is no such equivalence between third-party PSI and third-party private set difference.

The formulation of TP-PSD is motivated by recent critical real-world applications. During a pandemic, individuals may hold location data of others who are within proximity for a certain period via contact tracing tokens. In the event an individual is known to be infected, the public health authority (PHA) is required to notify the close contacts of that individual on the risk of potential exposure. However, there is a time lapse between the onset of infection to diagnosis, especially for an asymptomatic individual. During this period, an asymptomatic individual might have come into contact with numerous people, many of whom might have already undergone testing during that time frame.

Ideally, the PHA should only need to notify close contacts of the infected individual who have not undergone testing during the specified time frame. In such settings, each individual assumes the role of $P_1$, while the entire list of close contacts in the relevant time frame corresponds to $S_1$. The testing centers assume the role of $P_2$, while the database of all individuals who have undergone testing during the same time frame corresponds to $S_2$. The PHA assumes the role of the third party which does not have any input. TP-PSD enables the PHA to achieve the required objective in a privacy-preserving manner.

The functionality of privately computing the symmetric difference $S_1 \triangle S_2$ also arises naturally in practical settings. We examine a scenario that occurs often in medical research, where a research institution seeks to obtain relevant patient information to study the effectiveness of a single round treatment procedure. In this scenario, specialized medical facilities $P_1$ and $P_2$ which can administer such treatment procedures hold a set of records $S_1$ and $S_2$ respectively. $S_1$ contains the set of patients who have undergone exactly one round of treatment procedure at $P_1$, while $S_2$ contains the set of patients who have undergone exactly one round of treatment procedure at $P_2$. The research institution is the third-party which assumes the role of $Q$. The private set symmetric difference outcome enables the research institution to obtain the required data in a privacy-preserving manner without gaining more information than required, while also safeguarding the patient records of each medical facility from the other.

Building on our third-party private set difference protocol, we present a TP-PSymD protocol that privately computes the symmetric difference $S_1 \triangle S_2$ in the third-party setting.

## 1.1 Organization

In Section 2, we formalize the functionality requirement in the third-party model, and review the cryptographic primitives that are used as building blocks in our constructions. In Section 3, we present a basic version of our multi-party third-party PSI protocol. Then, in Sections 4 and 5, we present our main multi-party private set intersection protocols which are secure against collusions of any subset of parties. The protocols to achieve the functionalities of private set difference and private symmetric difference are provided in Section 6 and Section 7 respectively. In Section 8, we report the performance evaluations of our protocols. Finally, we conclude in Section 9.

# 2 Preliminaries

## 2.1 Definitions

We start by giving definitions for the various types of protocols we will introduce in this paper.

**Definition 1** (Multi-party third party private set intersection protocol)**.** *In a multi-party third party private set intersection (MP-TP-PSI) protocol, $N$ parties $P_1, \ldots, P_N$ hold datasets $S_1, \ldots, S_N$ respectively with elements in $\{0,1\}^*$, while a third-party $Q$ has no input. At the end of the protocol, $Q$ outputs $\bigcap_{i=1}^{N} S_i$, and the other parties output $\perp$.*

**Definition 2** (Third-party private set difference protocol)**.** *In a third-party private set difference (TP-PSD) protocol, two parties $P_1$ and $P_2$ hold datasets $S_1$ and $S_2$ respectively with elements in $\{0,1\}^*$, while a third-party $Q$ has no input. At the end of the protocol, $Q$ outputs $S_1 \setminus S_2$, and the other parties output $\perp$.*

**Definition 3** (Third-party private symmetric difference protocol)**.** *In a third-party private symmetric difference (TP-PSymD) protocol, two parties $P_1$ and $P_2$ hold datasets $S_1$ and $S_2$ respectively with elements in $\{0,1\}^*$, while a third-party $Q$ has no input. At the end of the protocol, $Q$ outputs $S_1 \triangle S_2$, and the other parties output $\perp$.*

Ideal-world/real-world simulation-based definitions can be used to define the security of such protocols. For example, a MP-TP-PSI protocol is secure if it achieves the ideal functionality shown in Figure 1.

## 2.2 Oblivious PRFs

Oblivious PRFs (OPRFs) were introduced by Naor and Reingold in [43]. There are two parties in an oblivious PRF protocol, a sender $S$ with a key, and a receiver $R$ who holds a private input. An OPRF allows $R$ to obtain an evaluation of the PRF, without $S$ learning the input nor $R$ learning the key. The ideal functionality of an OPRF is described in Figure 2.

**Parameters:** The number of parties $N$ and the size $n$ of each dataset.
**Functionality:**

1. For $i \in [N]$, get $P_i$'s input set $S_i$ of size $n$.
2. Send $\bigcap_{i=1}^{N} S_i$ to $Q$.

Figure 1: Multi-party third party PSI ideal functionality

**Parameters:** A pseudorandom function $F : \mathcal{K} \times I \to O$ with key space $\mathcal{K}$, input space $I$ and output space $O$.
**Functionality:**

1. Get key $k \in \mathcal{K}$ from $S$.
2. Get input $x \in I$ from $R$.
3. Send $F(k,x)$ to $R$.

Figure 2: Oblivious PRF ideal functionality

Many other OPRF constructions have been proposed since its introduction in [43]. Some of these constructions [19, 38] achieve slightly weaker security guarantees, while other constructions [13, 47] allow for the evaluation of the PRF at multiple points at once. OPRFs have been successfully applied to PSI in several works [13, 38, 47, 55].

## 2.3 Homomorphic Encryption

Homomorphic encryption is a form of encryption that allows computations to be performed on encrypted data. It is broadly classified into partially homomorphic encryption, which supports the evaluation of circuits consisting of only one type of gate, and fully homomorphic encryption, which supports evaluation of arbitrary circuits (possibly of bounded size).

Partially homomorphic encryption schemes can be further classified into two broad types depending on the type of operation supported. Additively homomorphic encryption schemes, such as Paillier [45], support addition, while multiplicatively homomorphic encryption schemes, such as RSA or ElGamal [18], support multiplication.

An important property that greatly enhances the usefulness of a homomorphic encryption scheme is circuit privacy, the property where a ciphertext obtained via evaluations is statistically indistinguishable from a fresh ciphertext. Many partially homomorphic encryption schemes can be easily modified to achieve circuit privacy via re-randomization techniques. This includes Paillier, RSA and ElGamal, which have circuit privacy for the class of all circuits that can be evaluated by each respective encryption scheme.

Circuit privacy can also be achieved for many fully homomorphic encryption schemes using techniques such as noise

flooding [25], bootstrapping or adding a small noise in each step of homomorphic evaluation [7].

Finally, a weaker, but still useful notion, is that of input privacy where a ciphertext obtained via evaluations reveals no information about the inputs except for what can be deduced from the output (i.e. the decrypted ciphertext).

# 3 A Basic Multi-Party Third Party PSI Protocol

## 3.1 An overview

In this section, we will present a basic version of our multi-party third party PSI protocol, which is secure against collusions of any subset of parties that do not include $Q$, or against a corrupt $Q$.

Our protocol combines the zero-sharing technique introduced by Kolesnikov et al. [39] with the technique of encoding intersection elements into a polynomial introduced in [63]. Furthermore, we improve the analysis of the error probability in [63] for our protocol, thus allowing us to select smaller parameters and achieve a linear communication complexity.[1]

The main idea behind our protocol is for the parties to create shares of zero for each element in the intersection. Once the parties exchange some keys, these shares of zero can be created by evaluating a PRF.

Each party $P_i$ then encodes his share of zero (corresponding to an element $s$) into a polynomial $p_i$ at the point $s$. The polynomials $p_i$ from the various parties $P_i$ are then sent to $Q$, who can determine the desired intersection by finding the points where the sum of the $p_i$'s evaluate to 0.

However, the above will fail to produce the correct output when the parties have identical sets, as the sum of the $p_i$'s will then be the zero polynomial, and hence evaluate to 0 at all elements. To handle this edge case, we use a trick introduced in [64], and have the parties choose a polynomial with degree that is one higher than is seemingly necessary. This additional randomness means that the sum of the $p_i$'s will be a non-zero polynomial with high probability, thus allowing $Q$ to correctly compute the intersection.

As discussed in the introduction, it can be more difficult to design a multi-party PSI protocol in the third-party setting since $Q$ has no private information that can be used to constrain the intersection elements. In fact, this is why the technique of encoding intersection elements as roots of a polynomial is used here so that $Q$, who does not possess any information about any of the private input sets, can still determine the intersection using root finding.

Note that, even with this technique of encoding intersection elements as roots of a polynomial, efficiency also differs between the conventional setting and third-party setting. Indeed, if we were to give $P_1$ a polynomial which has roots at the intersection elements, he can simply use a multi-point evaluation to evaluate the elements of his private input set $S_1$ at this polynomial to determine the intersection, and this is significantly cheaper than root finding.

## 3.2 Details of the protocol

Suppose there are $N$ parties $P_1,\dots,P_N$, each with a dataset $S_i \subseteq \{0,1\}^\ell$ of size $n$. Let $\lambda > 0$ be the correctness parameter and let $\mathbb{F}$ be a finite field with $|\mathbb{F}| > 2^{\ell+\lambda}$. We fix an injective map $\iota : \{0,1\}^\ell \hookrightarrow \mathbb{F}$ with image $S$, fix some $a_0 \in \mathbb{F} \setminus S$ and let $F : \mathcal{K} \times S \to \mathbb{F}$ be a PRF. For ease of notation, we implicitly identify $\{0,1\}^\ell$ with its image $S \subseteq \mathbb{F}$ under the map $\iota$.

1. For each $i, j \in [N]$ with $j \neq i$, $P_i$ generates a key $k_{i,j} \in \mathcal{K}$, and sends $k_{i,j}$ to $P_j$.
2. For $i \in [N]$, $P_i$ picks a random $r_i \leftarrow \mathbb{F}$ and computes the polynomial $p_i(X)$ of degree $\leq n$ such that $p_i(a_0) = r_i$ and
$$p_i(s) = \sum_{j \in [N] \setminus \{i\}} (F(k_{i,j}, s) - F(k_{j,i}, s))$$
for all $s \in S_i$, and sends $p_i(X)$ to $Q$.
3. $Q$ solves $\sum_{i=1}^N p_i(X) = 0$ and outputs $\{s \in S : \sum_{i=1}^N p_i(s) = 0\}$.

Protocol 1: A semi-honest MP-TP-PSI protocol

The communication complexity of Protocol 1 is

$$O\big(N^2 \log|\mathcal{K}| + Nn \log|\mathbb{F}|\big) = O\big(N^2 \log|\mathcal{K}| + Nn(\ell+\lambda)\big).$$

As for the computation cost, we note that each party $P_i$ generates $(N-1)$ keys for the PRF $F$, invokes the PRF $2n(N-1)$ times and interpolates a polynomial of degree $n$, while $Q$ needs to find the roots of a single polynomial of degree $n$. Since the computational complexities of certain operations depend on whether $\mathbb{F}$ has prime order, we shall assume that $|\mathbb{F}|$ is prime when stating the computational complexities.[2]

Under this assumption, multiplication in $\mathbb{F}$ has $O(\log|\mathbb{F}| \log\log|\mathbb{F}|)$ bit complexity, hence polynomial interpolation has a bit complexity of

$$O\big(n(\log n)^2(\log\log n)\log|\mathbb{F}|\log\log|\mathbb{F}|\big),$$

while Kedlaya-Umans has a bit complexity of

$$O\Big(n^{1.5+o(1)}\log^{1+o(1)}|\mathbb{F}| + n^{1+o(1)}\log^{2+o(1)}|\mathbb{F}|\Big).$$

Thus, ignoring the generation of keys (which has insignificant costs) and using Kedlaya-Umans [36] for root finding, this gives a total computational complexity of

$$O\big(N^2 n c_{\mathsf{comp},F} + Nn(\log n)^2(\log\log n)\log|\mathbb{F}|\log\log|\mathbb{F}|$$

[1] The communication complexity for the protocol in [63] is quasilinear.

[2] By Bertrand's postulate, there always exists a finite field $\mathbb{F}$ of prime order such that $2^{\ell+\lambda} < |\mathbb{F}| < 2^{\ell+\lambda+1}$.

$$+n^{1.5+o(1)}\log^{1+o(1)}|\mathbb{F}|+n^{1+o(1)}\log^{2+o(1)}|\mathbb{F}|\Big)$$

where $c_{\mathsf{comp},F}$ is the cost of a single invocation of $F$.

### 3.3 Correctness and security

**Lemma 1.** *In Protocol 1, $\sum_{i=1}^N p_i(X)$ is indistinguishable from a polynomial chosen uniformly from the set*

$$\left\{r(X)\in\mathbb{F}[X]:\deg(r)\le n \text{ and } r(s)=0 \text{ for all } s\in\bigcap_{i=1}^N S_i\right\}.$$

*Proof.* Let the above set be $U$, $p(X)=\sum_{i=1}^N p_i(X)$ and $I=\bigcap_{i=1}^N S_i$. If $s\in I$, then

$$\begin{aligned}p(s)=\sum_{i=1}^N p_i(s)&=\sum_{i=1}^N\sum_{j\in[N]\setminus\{i\}}(F(k_{i,j},s)-F(k_{j,i},s))\\&=\sum_{i,j\in[N],i\neq j}(F(k_{i,j},s)-F(k_{j,i},s))=0,\end{aligned}$$

hence $p(X)\in U$.

Next, we show that $p(X)$ is indistinguishable from a uniformly random element of $U$. To do so, we first change how the polynomials $p_i(X)$ are defined. For each $s\in\bigcup_{i=1}^N S_i$, we generate uniformly random values $\alpha_{s,i}\in\mathbb{F}$ for $i\in[N]$ subject to the constraint $\sum_{i=1}^N\alpha_{s,i}=0$. Then we pick independent and uniformly random $r_1,\dots,r_N\in\mathbb{F}$, and let $p_i(X)$ be the unique polynomial of degree $\le n$ such that $p_i(a_0)=r_i$ and $p_i(s)=\alpha_{s,i}$ for all $s\in S_i$. Since $F$ is a PRF, the joint distributions of $p_i(X)$ after this change is indistinguishable from that of the real protocol.

We claim that, after this change, $p(X)=\sum_{i=1}^N p_i(X)$ is a uniformly random element of $U$. Let $S_1\setminus I=\{t_1,\dots,t_{n-|I|}\}$. Since a polynomial of degree $\le n$ is uniquely determined by its values at any $n+1$ points, it suffices to prove that $p(t_1),\dots,p(t_{n-|I|}),p(a_0)$ are jointly uniformly random (recall that $p(X)$ evaluates to 0 at the elements of $I$).

Since $t_1,\dots,t_{n-|I|}\notin I$, for each $k\in[n-|I|]$, there exists $i_k$ such that $t_k\notin S_{i_k}$, which means that $p_{i_k}(X)$ is independent of $\alpha_{t_k,i_k}$. Therefore, $p_1(t_k)=\alpha_{t_k,1}$ is independent of $p_2(t_k),\dots,p_N(t_k)$, which proves that $p(t_k)=\sum_{i=1}^N p_i(t_k)=\alpha_{t_k,1}+\sum_{i=2}^N p_i(t_k)$ is uniformly random.

Furthermore, if $k\neq k'$, then $\alpha_{t_k,i}$ and $\alpha_{t_{k'},i}$ are independent by the definition of $\alpha_{s,i}$, from which it follows that $p(t_1),\dots,p(t_{n-|I|})$ are jointly uniformly random. It is clear also that $p(a_0)$ is uniformly random and independent of $p(t_1),\dots,p(t_{n-|I|})$, proving the claim. □

**Proposition 2.** *In Protocol 1, $Q$ outputs $\bigcap_{i=1}^N S_i$ except with probability negligible in $\lambda$.*

*Proof.* Let $I=\bigcap_{i=1}^N S_i$ and $p(X)=\sum_{i=1}^N p_i(X)$. By Lemma 1, $p(X)$ is indistinguishable from a uniformly random element of the set

$$\{r(X)\in\mathbb{F}[X]:\deg(r)\le n \text{ and } r(s)=0 \text{ for all } s\in I\},$$

so $p(X)=q(X)\prod_{s\in I}(X-s)$ where $q(X)$ is indistinguishable from a uniformly random polynomial of degree $\le n-|I|$.

If $s\notin I$, then the probability that $q(s)$, or equivalently $p(s)$, equals 0 is negligibly close to $1/|\mathbb{F}|$. By the union bound, the probability that $p(s)=0$ for some $s\notin I$ is bounded above by

$$\frac{n-|I|}{|\mathbb{F}|}\le\frac{n}{2^{\ell+\lambda}}\le\frac{2^\ell}{2^{\ell+\lambda}}=2^{-\lambda},$$

which is negligible in $\lambda$, as required. □

**Proposition 3.** *Protocol 1 is secure in the semi-honest model against collusion of any number of parties $P_i$.*

*Proof.* This is clear. The only messages exchanged between the parties $P_i$ are the keys $k_{i,j}$, which are independent of the inputs $S_1,\dots,S_N$. □

**Proposition 4.** *Protocol 1 is secure in the semi-honest model against $Q$.*

*Proof.* By the proof of Lemma 1, if $s\notin\bigcap_{i=1}^N S_i$, the values $p_i(s)$ are indistinguishable from uniformly random, and if $s\in\bigcap_{i=1}^N S_i$, the values $p_i(s)$ are indistinguishable from uniformly random values that satisfy the constraint $\sum_{i=1}^N p_i(s)=0$.

Hence, we can simulate the messages received by $Q$ as follows. For each $s\in\bigcap_{i=1}^N S_i$, pick random values $\alpha_{s,1},\dots,\alpha_{s,N}$ that sum to zero. Then, for each $i\in[N]$, pick a random $q_i(X)$ of degree $\le n$ such that $q_i(s)=\alpha_{s,i}$ for all $s\in\bigcap_{i=1}^N S_i$. The message received by $Q$ from $P_i$ is then simulated using the polynomial $q_i(X)$. □

However, we note that Protocol 1 will not be secure against a collusion of some party $P_i$ with $Q$. Suppose, for some $i$, the parties $P_i$ and $Q$ collude. Then, the corrupt parties will know whether an element $s\in\bigcap_{j\in[N]\setminus\{i\}}S_j$ by checking if

$$\sum_{j\in[N]\setminus\{i\}}p_j(s)+\sum_{j\in[N]\setminus\{i\}}(F(k_{i,j},s)-F(k_{j,i},s))=0.$$

## 4 An Improved Multi-Party Third Party PSI Protocol

### 4.1 An overview

In this section, we introduce a multi-party third party PSI protocol that is secure in the semi-honest model against collusions of any subset of parties.

We noted previously, in Section 3.3, that Protocol 1 is insecure against a collusion of some party $P_i$ and $Q$. This is because $P_i$ can determine if an element $s\in S$ lies in $\bigcap_{j\in[N]\setminus\{i\}}S_i$ even when $s\notin S_i$. A more careful look at the attack reveals that it works as $P_i$ can easily compute $F(k_{j,i},s)$ for any $s\in S$.

We modify Protocol 1 so that each $P_i$ can only obtain exactly $n$ evaluations of $F$ under the key $k_{j,i}$, and we achieve this with the use of an OPRF. This change not only disables the above attack, but it also makes the protocol secure against any collusion.

## 4.2 Details of the protocol

We use the same setup as Section 3.2. Furthermore, let $\mathcal{F}_{\mathsf{OPRF}}^{F}$ be an OPRF protocol for $F$.

1. For each $i, j \in [N]$ with $j \neq i$, $P_i$ generates a key $k_{i,j} \in \mathcal{K}$.
2. For each $i, j \in [N]$ with $j \neq i$, and each $s \in S_j$, $P_i$ and $P_j$ invoke $\mathcal{F}_{\mathsf{OPRF}}^{F}$ where
   - $P_i$ is sender with input $k_{i,j}$,
   - $P_j$ is receiver with input $s$.
3. For $i \in [N]$, $P_i$ picks a random $r_i \leftarrow \mathbb{F}$, computes the polynomial $p_i(X)$ of degree $\leq n$ such that $p_i(a_0) = r_i$ and
$$p_i(s) = \sum_{j \in [N] \setminus \{i\}} (F(k_{i,j}, s) - F(k_{j,i}, s))$$
for all $s \in S_i$, and sends $p_i(X)$ to $Q$.
4. $Q$ solves $\sum_{i=1}^{N} p_i(X) = 0$ and outputs $\{s \in S : \sum_{i=1}^{N} p_i(s) = 0\}$.

Protocol 2: A semi-honest MP-TP-PSI protocol secure against any collusion

The communication complexity of Protocol 2 is

$$O\big(N^2 n c_{\mathsf{comm},\mathcal{F}} + Nn \log |\mathbb{F}|\big) = O\big(N^2 n c_{\mathsf{comm},\mathcal{F}} + Nn(\ell + \lambda)\big)$$

where $c_{\mathsf{comm},\mathcal{F}}$ is the communication cost of the OPRF protocol $\mathcal{F}_{\mathsf{OPRF}}^{F}$. For the computation cost, we note that each party $P_i$ invokes $\mathcal{F}_{\mathsf{OPRF}}^{F}$ a total of $(N-1)n$ times as sender and $(N-1)n$ times as receiver, and interpolates a single polynomial of degree $n$. As in Protocol 1, $Q$ needs to find the roots of a single polynomial of degree $\leq n$. So the total computational complexity is

$$O\big(N^2 n c_{\mathsf{comp},\mathcal{F}} + Nn(\log n)^2 (\log\log n) \log |\mathbb{F}| \log\log |\mathbb{F}| + n^{1.5+o(1)} \log^{1+o(1)} |\mathbb{F}| + n^{1+o(1)} \log^{2+o(1)} |\mathbb{F}|\big)$$

where $c_{\mathsf{comp},\mathcal{F}}$ is the computation cost of $\mathcal{F}_{\mathsf{OPRF}}^{F}$.

## 4.3 Correctness and security

**Proposition 5.** *In Protocol 2, $Q$ outputs $\bigcap_{i=1}^{N} S_i$ except with probability negligible in $\lambda$.*

*Proof.* For each $i \in [N]$, the polynomial $p_i(X)$ sent by $P_i$ to $Q$ in Protocol 2 is identical to that in Protocol 1. The correctness of Protocol 2 follows. □

**Proposition 6.** *Protocol 2 is secure in the semi-honest model against collusion of any number of parties $P_i$ with $Q$.*

*Proof.* Let $H = \{i \in [N] : P_i \text{ is honest}\}$ be the set of indices of the honest parties, and $C = \{i \in [N] : P_i \text{ is corrupt}\}$ be the set of indices of the corrupt parties except $Q$. We show how to simulate the view of the corrupt parties. First, the simulator chooses the keys $k_{i,j}$ for the honest parties (i.e. for $i \in H$), and follows step 2 of the protocol.

Note that, for any $s \in \{0,1\}^{\ell}$,

$$\sum_{i \in H} \sum_{j \in [N] \setminus \{i\}} (F(k_{i,j}, s) - F(k_{j,i}, s)) + \sum_{i \in C} \sum_{j \in [N] \setminus \{i\}} (F(k_{i,j}, s) - F(k_{j,i}, s)) = 0.$$

We now consider 3 separate cases.

*Case 1: $s \in \bigcap_{i=1}^{N} S_i$.* In this case, the above constraint is equivalent to

$$\sum_{i \in H} p_i(s) + \sum_{i \in C} p_i(s) = 0,$$

hence the values $p_i(s)$ for $i \in H$ are indistinguishable from uniformly random values that satisfy the constraint $\sum_{i \in H} p_i(s) = -\sum_{i \in C} p_i(s)$.

*Case 2: $s \notin S_{i_0}$ for some $i_0 \in H$.* For any $i \in H$ such that $s \notin S_i$, $P_i$ did not interpolate a value for $p_i(X)$ at $s$, hence $p_i(s)$ is indistinguishable from uniformly random. On the other hand, for any $i \in H$ with $s \in S_i$, $F(k_{i,i_0}, s)$ and hence $p_i(s)$ are indistinguishable from uniformly random to the corrupt parties.

*Case 3: $s \notin \bigcap_{i=1}^{N} S_i$ but $s \in S_i$ for all $i \in H$.* In this case, there must exist some $i_0 \in C$ such that $s \notin S_{i_0}$. The above constraint now becomes

$$\sum_{i \in H} p_i(s) + \sum_{i \in C} \sum_{j \in [N] \setminus \{i\}} (F(k_{i,j}, s) - F(k_{j,i}, s)) = 0.$$

For all $j \in H$, the values of $F(k_{j,i_0}, s)$ are unknown to the corrupt parties, hence $p_i(s)$ for $i \in H$ are again indistinguishable from uniformly random values.

Therefore, we can simulate the polynomials $p_i(X)$ for $i \in H$ as follows. Fix any $j \in H$. Now, for $i \in H \setminus \{j\}$, pick a polynomial $p_i(X)$ of degree $\leq n$ uniformly at random. Finally, let $p_j(X)$ be a random polynomial of degree $\leq n$ such that

$$p_j(s) = -\sum_{i \in [N] \setminus \{j\}} p_i(s)$$

for all $s \in \bigcap_{i=1}^{N} S_i$. □

# 5 A Quasilinear Multi-Party Third-Party PSI Protocol

## 5.1 An overview

While the protocol in the previous section has a communication complexity that is linear in $n$, its computational complexity is slightly higher at $O(n^{1.5+o(1)})$. In this section, we further

improve the protocol to achieve a quasilinear computational complexity.

Recall that, in Protocol 2, $Q$ uses the information obtained from the parties $P_i$ to form a polynomial which has roots at the intersection elements. However, this polynomial is of degree $n$ and thus has other irreducible factors, which are almost always non-linear. Finding the roots of such a polynomial is significantly more costly than finding the roots of a polynomial which splits into distinct linear factors.

Indeed, this is because many polynomial factorization algorithms such as the Cantor-Zassenhaus algorithm [9] work via a three-step process, namely square-free factorization, distinct-degree factorization and equal-degree factorization. Square-free factorization factorizes the input polynomial into powers of square-free polynomials, distinct degree factorization then takes as input a monic square-free polynomial and factorizes it into polynomials where the $i$-th polynomial is a product of degree $i$ irreducible polynomials (or is equal to 1), and finally equal-degree factorization takes as input a polynomial whose irreducible factors are degree $i$ polynomials for some fixed $i$ and outputs its irreducible factors.

Of these three steps, the second step of distinct-degree factorization is the most costly. Kedlaya-Umans [36] achieves a complexity which is near-linear in the degree of the polynomial for square-free factorization and equal-degree factorization, but which however is non-linear for distinct-degree factorization. Furthermore, factorization of a polynomial which splits into distinct linear factors has even lower computational complexity. In fact, the Cantor-Zassenhaus algorithm [9] has quasilinear complexity in this special case.

Therefore, we introduce a technique that allows $Q$ to cheaply obtain a polynomial which splits into distinct linear factors, where each linear factor corresponding to an intersection element. This lets $Q$ find the roots of the polynomial, and hence obtain the intersection result, in quasilinear time. We achieve this by allowing $Q$ to obtain two different random polynomials $q_1$ and $q_2$, both of which have roots at the intersection elements. By taking the greatest common divisor of $q_1$ and $q_2$ (the gcd can be computed in quasilinear time), $Q$ then obtains a polynomial $q$ which has no extraneous factors, and thus can be solved in quasilinear time.

Furthermore, for a fixed error probability, this new technique allows the use of a smaller field (see Proposition 7 and the following discussion), and this also contributes to the communication and computational efficiency of the protocol.

## 5.2 Details of the protocol

We use a similar setup as Section 4.2, but replacing $F$ by a PRF $F : \mathcal{K} \times S \to \mathbb{F}^2$. Let $\pi_i : \mathbb{F}^2 \to \mathbb{F}$ (for $i = 1,2$) be the projection onto the $i$-th coordinate, and let $F_i = \pi_i \circ F$. As before, we let $\mathcal{F}^F_{\mathsf{OPRF}}$ be an OPRF protocol for $F$.

1. For each $i, j \in [N]$ with $j \neq i$, $P_i$ generates a key $k_{i,j} \in \mathcal{K}$.
2. For each $i, j \in [N]$ with $j \neq i$, and each $s \in S_j$, $P_i$ and $P_j$ invoke $\mathcal{F}^F_{\mathsf{OPRF}}$ where
   - $P_i$ is sender with input $k_{i,j}$,
   - $P_j$ is receiver with input $s$.
3. For $i \in [N]$, $P_i$ picks random $r_{i,1}, r_{i,2} \leftarrow \mathbb{F}$, and computes the polynomials $p_{i,1}(X)$ and $p_{i,2}(X)$, each of degree $\leq n$, such that $p_{i,h}(a_0) = r_{i,h}$ and
$$p_{i,h}(s) = \sum_{j \in [N] \setminus \{i\}} (F_h(k_{i,j}, s) - F_h(k_{j,i}, s))$$
   for all $s \in S_i$, and sends $p_{i,1}(X)$ and $p_{i,2}(X)$ to $Q$.
4. $Q$ computes the polynomial
$$q(X) = \gcd\left(\sum_{i=1}^{N} p_{i,1}(X), \sum_{i=1}^{N} p_{i,2}(X)\right).$$
5. $Q$ solves $q(X) = 0$ and outputs $\{s \in S : q(s) = 0\}$.

Protocol 3: A quasilinear semi-honest MP-TP-PSI protocol secure against any collusion

Note that, although the PRF $F$ used in this protocol has an output length that is twice as long as the PRF used in Protocol 2, running $\mathcal{F}^F_{\mathsf{OPRF}}$ can be, and is often, significantly cheaper than two evaluations of an OPRF with output space $\mathbb{F}$. This means that, although Protocol 3 has increased computational and communication costs for the parties $P_i$ as compared to Protocol 2, these costs are likely less than twice the corresponding costs for Protocol 2.

Furthermore, as alluded to earlier, when compared to Protocol 2, we can use a smaller field in Protocol 3 while achieving the same error probability, thus leading to a further reduction in the computational and communication costs in practice.

The communication complexity of Protocol 3 is
$$O\big(N^2 n c_{\mathsf{comm},\mathcal{F}} + Nn(\ell + \lambda)\big)$$
which is linear in $n$, where $c_{\mathsf{comm},\mathcal{F}}$ is the communication cost of the OPRF protocol $\mathcal{F}^F_{\mathsf{OPRF}}$. Using the Cantor-Zassenhaus algorithm [9] with fast integer multiplication [31] and fast polynomial multiplication [30] for the root finding step, its computational complexity is
$$O\Big(n \log(n \log|\mathbb{F}|) \log n \log|\mathbb{F}| (\log n + \log|\mathbb{F}|) \cdot 4^{\max(0, \log^* n - \log^*|\mathbb{F}|)} + n \log|\mathbb{F}| (\log\log|\mathbb{F}|)^2\Big)$$
where $\log^*$ is the iterated logarithm. Hence, the computational complexity of Protocol 3 is
$$O\Big(n \log(n \log|\mathbb{F}|) \log n \log|\mathbb{F}| (\log n + \log|\mathbb{F}|)$$

$$\cdot 4^{\max(0,\log^* n-\log^*|\mathbb{F}|)} + n\log|\mathbb{F}|(\log\log|\mathbb{F}|)^2 + Nn(\log n)^2(\log\log n)\log|\mathbb{F}|\log\log|\mathbb{F}| + N^2 n c_{\mathsf{comp},\mathcal{F}}\Big)$$

which is quasilinear in $n$, where $c_{\mathsf{comp},\mathcal{F}}$ is the computation cost of $\mathcal{F}^F_{\mathsf{OPRF}}$.

### 5.3 Correctness and security

**Proposition 7.** *In Protocol 3, $Q$ outputs $\bigcap_{i=1}^N S_i$ except with probability negligible in $\lambda$.*

*Proof.* Let $I = \bigcap_{i=1}^N S_i$ and $p_h(X) = \sum_{i=1}^N p_{i,h}(X)$ for $h = 1,2$. By Lemma 1,

$$p_h(X) = q_h(X)\prod_{s\in I}(X-s)$$

where $q_1(X)$ and $q_2(X)$ are independent and uniformly random polynomials of degree $\le n-|I|$.

For any monic irreducible polynomial $r(X) \in \mathbb{F}[X]$, we have $r(X) \mid q_h(X)$ if and only if $q_h(X) = r(X)f(X)$ for some $f(X) \in \mathbb{F}[X]$ of degree $\le n-|I|-\deg(r)$. Therefore, writing $d = \deg(r)$,

$$\begin{aligned}
&\Pr[r(X) \mid \gcd(q_1(X), q_2(X))] \\
&= \Pr[r(X) \mid q_1(X) \text{ and } r(X) \mid q_2(X)] \\
&= \Pr[r(X) \mid q_1(X)]\Pr[r(X) \mid q_2(X)] \\
&= \left(\frac{|\mathbb{F}|^{n-|I|-d}}{|\mathbb{F}|^{n-|I|}}\right)^2 = \frac{1}{|\mathbb{F}|^{2d}}.
\end{aligned}$$

Note that $\gcd(q_1(X), q_2(X)) \neq 1$ if and only if there exists some monic irreducible polynomial $r(X)$ such that $r(X) \mid \gcd(q_1(X), q_2(X))$. Since there are at most $|\mathbb{F}|^d$ monic irreducible polynomials $r(X)$ of degree $d$, by the union bound,

$$\begin{aligned}
\Pr[\gcd(q_1(X), q_2(X)) \neq 1] &\le \sum_{d=1}^{n-|I|} \frac{|\mathbb{F}|^d}{|\mathbb{F}|^{2d}} < \sum_{d=1}^{\infty} \frac{1}{|\mathbb{F}|^d} \\
&= \frac{1}{|\mathbb{F}|-1} \le \frac{1}{2^{\ell+\lambda}-1},
\end{aligned}$$

which is negligible in $\lambda$.

In all other cases, we have $\gcd(q_1(X), q_2(X)) = 1$, hence

$$\begin{aligned}
q(X) &= \gcd(p_1(X), p_2(X)) \\
&= \gcd\left(q_1(X)\prod_{s\in I}(X-s), q_2(X)\prod_{s\in I}(X-s)\right) \\
&= \prod_{s\in I}(X-s).
\end{aligned}$$

□

Comparing the proofs of Propositions 5 and 7, we see that a smaller field can be used in Protocol 3 as compared to Protocol 2 while achieving the same error probability. Indeed, the error probability for Protocol 2 is bounded above by $2^{-\lambda}$, while that for Protocol 3 is bounded above by $(2^{\ell+\lambda}-1)^{-1}$. For example, with $\ell = 32$, in order to achieve an error probability of $2^{-40}$, we require $|\mathbb{F}| \gtrapprox 2^{72}$ in Protocol 2, but we only require $|\mathbb{F}| \gtrapprox 2^{40}$ in Protocol 3. Intuitively, this can be explained by the fact that it is much less likely for two random polynomials to both have the same root in $S$ than for a single random polynomial to have a root in $S$.

**Proposition 8.** *Protocol 3 is secure in the semi-honest model against collusion of any number of parties $P_i$ with $Q$.*

*Proof.* Let $H = \{i \in [N] : P_i \text{ is honest}\}$ be the set of indices of the honest parties. The proof is similar to that of Proposition 6, except that we need to simulate two sets of polynomials $p_{i,1}(X)$ and $p_{i,2}(X)$ for $i \in H$. This is done in a similar manner as in the proof of Proposition 6.

Fix any $j \in H$. For $i \in H \setminus \{j\}$, pick polynomials $p_{i,1}(X)$ and $p_{i,2}(X)$ of degree $\le n$ independently and uniformly at random. Finally, let $p_{j,1}(X)$ and $p_{j,2}(X)$ be independent random polynomials of degree $\le n$ such that

$$p_{j,1}(s) = -\sum_{i\in[N]\setminus\{j\}} p_{i,1}(s) \quad \text{and} \quad p_{j,2}(s) = -\sum_{i\in[N]\setminus\{j\}} p_{i,2}(s)$$

for all $s \in \bigcap_{i=1}^N S_i$. □

## 6 Third-Party Private Set Difference via Rational Functions

### 6.1 An overview

In this section, we introduce a protocol for third-party private set difference which is based on the works of Minsky et al. [42] and Ghosh and Simkin [26]. In particular, we build upon a threshold PSI protocol in [26] to obtain our TP-PSD protocol.

Let $p_{S_1}(X)$ and $p_{S_2}(X)$ be the polynomials with roots given by the sets $S_1$ and $S_2$ respectively. First, $P_1$ and $P_2$ choose random polynomials $r_1(X)$ and $r_2(X)$. Then, using an additively homomorphic encryption scheme, they provide $Q$ with evaluations of the rational function $f(X) = \frac{r_1(X)p_{S_1}(X)+r_2(X)p_{S_2}(X)}{p_{S_1}(X)}$ at various $\alpha_i$'s. With these values, $Q$ can then reconstruct the rational function $f(X)$, which has poles exactly at the elements of $S_1 \setminus S_2$.[3]

As compared to the protocol in [26], we have replaced their use of oblivious linear evaluation with an additive homomorphic encryption scheme. Furthermore, we introduced additional randomness to the evaluations of the polynomials defining the rational function so that the third-party $Q$ does not learn any additional information about $S_1$ beyond the elements of $S_1 \setminus S_2$.

[3] The poles of a rational function are the zeros of its denominator when the rational function is written in its reduced form.

## 6.2 Details of the protocol

Let the size of each of $P_1$ and $P_2$'s datasets be $n$, and let $S_1 = \{s_1, \ldots, s_n\} \subseteq \{0,1\}^\ell$ and $S_2 = \{t_1, \ldots, t_n\} \subseteq \{0,1\}^\ell$. Let $\lambda > 0$ be the correctness parameter. We embed $\{0,1\}^\ell$ into a finite field $\mathbb{F}$ with $|\mathbb{F}| \geq \max\left(2^\ell + 3n + 1, 2^\lambda\right)$ using an injective map $\iota : \{0,1\}^\ell \hookrightarrow \mathbb{F}$. Let $S$ be the image of $\{0,1\}^\ell$ under $\iota$.

Fix $3n+1$ points $\alpha_1, \ldots, \alpha_{3n+1} \in \mathbb{F} \setminus S$, and let $E : \mathbb{F} \to \mathcal{C}$ be an asymmetric additively homomorphic encryption scheme with input privacy. For any subset $T \subseteq \{0,1\}^\ell$, we denote by $p_T(X)$ the polynomial $p_T(X) = \prod_{t \in T}(X - \iota(t)) \in \mathbb{F}[X]$.

1. $Q$ generates a key pair $(\mathsf{sk}, \mathsf{pk})$ for $E$ and sends $\mathsf{pk}$ to $P_1$ and $P_2$.
2. $P_1$ and $P_2$ pick $\beta_1, \ldots, \beta_{3n+1} \in \mathbb{F}$ uniformly at random.
3. $P_1$ chooses a random polynomial $r_1(X)$ of degree $\leq n$, computes $\mathsf{ct}_{1,i} = E(\mathsf{pk}, \beta_i r_1(\alpha_i) p_{S_1}(\alpha_i))$ and sends $\mathsf{ct}_{1,i}$ to $P_2$ for $i \in [3n+1]$.
4. $P_2$ chooses a random polynomial $r_2(X)$ of degree $\leq n$ and computes $\mathsf{ct}_{2,i} = E(\mathsf{pk}, \beta_i r_2(\alpha_i) p_{S_2}(\alpha_i))$ for $i \in [3n+1]$.
5. For each $i \in [3n+1]$, $P_2$ performs homomorphic addition on $\mathsf{ct}_{1,i}$ and $\mathsf{ct}_{2,i}$ to obtain a ciphertext $\mathsf{ct}_{0,i}$, and sends $\mathsf{ct}_{0,i}$ to $Q$.
6. $P_1$ computes $d_i = \beta_i p_{S_1}(\alpha_i)$ and sends $d_i$ to $Q$, for $i \in [3n+1]$.
7. For each $i \in [3n+1]$, $Q$ decrypts $\mathsf{ct}_{0,i}$ to obtain $n_i$.
8. $Q$ uses rational function interpolation to find the unique rational function $f(X)$ with numerator of degree $\leq 2n$ and denominator of degree $\leq n$ such that $f(\alpha_i) = n_i/d_i$ for $i \in [3n+1]$.
9. Let $f(X) = q_1(X)/q_2(X)$ where $q_1(X)$ and $q_2(X)$ are coprime polynomials. $Q$ outputs
$$\{\iota^{-1}(x) : x \in \mathbb{F} \text{ such that } q_2(x) = 0\}.$$

Protocol 4: A semi-honest TP-PSD protocol

## 6.3 Communication and computational complexity

Assuming that the ciphertexts for $E$ are at most a constant factor larger than the plaintexts, we easily see from the description of Protocol 4 that it has a communication complexity of $O(n \log |\mathbb{F}|) = O\left(n \log(2^\ell + n + 2^\lambda)\right)$, which, as a function of $n$, is $O(n \log n)$, hence quasilinear.

For large $n$, the most computational expensive steps in the above protocol are steps 8 and 9, which involve rational function interpolation and finding the roots of a polynomial in $\mathbb{F}[X]$ respectively.

Rational function interpolation can be performed by solving a linear system of $3n+1$ equations in $3n+2$ variables [42]. It is well known that this problem has the same asymptotic complexity as matrix multiplication, i.e. its complexity is $O(n^{\omega+o(1)})$ $\mathbb{F}$-operations, or equivalently $O(n^{\omega+o(1)} \log |\mathbb{F}| \log\log |\mathbb{F}|)$ bit operations, where $\omega$ is the exponent of matrix multiplication. Trivially, we have $2 \leq \omega \leq 3$. A long line of work on fast matrix multiplication algorithms has gradually improved known upper bounds on $\omega$ [1, 16, 46, 57, 61], and it has recently been shown by Williams et al. [61] that $\omega \leq 2.371552$. In practice, depending on the size of $n$, we can either apply Gaussian elimination which has a complexity of $O(n^3)$ $\mathbb{F}$-operations, or Strassen's algorithm [57] which has a complexity of $O(n^{\log 7 + o(1)}) \approx O(n^{2.8074+o(1)})$ operations.

As for root finding, we note that $q_2(X)$ factorizes into a product of distinct linear factors (see the proof of Proposition 13), hence the Cantor-Zassenhaus algorithm [9] has a quasilinear complexity of

$$O\Big(n \log(n \log |\mathbb{F}|) \log n \log |\mathbb{F}| (\log n + \log |\mathbb{F}|) \cdot 4^{\max(0, \log^* n - \log^* |\mathbb{F}|)} + n \log |\mathbb{F}| (\log\log |\mathbb{F}|)^2\Big)$$

in this special case (refer to the discussion following Protocol 3 for more details). Noting that $|\mathbb{F}| > n$, the above is $O\left(n \log(n \log |\mathbb{F}|) \log n \log^2 |\mathbb{F}|\right)$, which is $O\left(n \log(n \log n) \log^3 n\right)$ as a function of $n$.

Therefore, the computational complexity of Protocol 4 is dominated by rational function interpolation, and it has a bit complexity in $n$ that is

$$O\left(n^{\omega+o(1)} \log n \log\log n\right) = O\left(n^{\omega+o(1)}\right).$$

## 6.4 Correctness and security

The following is a lemma of Kissner and Song [37], which shows that linear combinations of coprime polynomials of the same degree by uniformly random polynomials give rise to a uniformly random polynomial.

**Lemma 9.** *Let $\mathbb{F}$ be a finite field, $p_1(X), p_2(X) \in \mathbb{F}[X]$ be coprime polynomials of degree $d$. If $r_1(X), r_2(X) \in \mathbb{F}[X]$ are uniformly random polynomials of degree $\leq n$, and $d \leq n$, then $r_1(X)p_1(X) + r_2(X)p_2(X)$ is a uniformly random polynomial of degree $\leq d + n$.*

We also require the following lemma by Ghosh and Simkin, which is stated in [26] for finite prime fields. However, the lemma is true even if the field is not prime, and we shall state and prove the more general version here.

**Lemma 10.** *Let $d > 0$ be some fixed integer, and let $\mathbb{F}$ be a finite field with $\log |\mathbb{F}| = \Theta(\kappa)$. If $p(X) \in \mathbb{F}[X]$ is a fixed*

*non-zero polynomial and $r(X) \in \mathbb{F}[X]$ is a uniformly random polynomial of degree $\leq d$, then* $\Pr[\gcd(p(X), r(X)) \neq 1]$ *is negligible in* $\kappa$.

*Proof.* Let

$$p(X) = p_1(X) \cdots p_k(X)$$

be the factorization of $p(X)$ in $\mathbb{F}[X]$ into irreducible polynomials. Then

$$\begin{aligned} &\Pr[\gcd(p(X), r(X)) \neq 1] \\ =&\Pr[\gcd(p_i(X), r(X)) \neq 1 \text{ for some } i] \\ \leq&\sum_{i=1}^{k} \Pr[\gcd(p_i(X), r(X)) \neq 1]. \end{aligned}$$

Since $p_i(X)$ is irreducible in $\mathbb{F}[X]$,

$$\begin{aligned} &\gcd(p_i(X), r(X)) \neq 1 \\ \iff& p_i(X) \mid r(X) \\ \iff& r(X) = p_i(X) q_i(X) \text{ for some } q_i(X) \in \mathbb{F}[X], \end{aligned}$$

from which it follows that

$$\Pr[\gcd(p_i(X), r(X)) \neq 1] = \frac{|\mathbb{F}|^{d-\deg(p_i)+1}}{|\mathbb{F}|^{d+1}} = \frac{1}{|\mathbb{F}|^{\deg(p_i)}}.$$

Therefore,

$$\Pr[\gcd(p(X), r(X)) \neq 1] \leq \sum_{i=1}^{k} \frac{1}{|\mathbb{F}|^{\deg(p_i)}} \leq \frac{k}{|\mathbb{F}|} \leq \frac{\deg(p)}{2^{\Theta(\kappa)}},$$

which is negligible in $\kappa$, as required. □

Finally, the following lemma by Minsky et al. [42] shows that a large enough number of evaluations of a rational function uniquely determines the rational function.

**Lemma 11.** *Given $n_1 + n_2$ pairs $(x_i, y_i) \in \mathbb{F}^2$, there is at most one rational function $f(X) = f_1(X)/f_2(X)$ (up to equivalence) with $f_1(X), f_2(X) \in \mathbb{F}[X]$ monic polynomials of degrees $n_1$ and $n_2$ respectively, that satisfies $f(x_i) = y_i$ for all $i \in [n_1 + n_2]$.*

We now state the following lemma, which is a variant of, and follows from Lemma 11. The result in this lemma appears to have been used implicitly in [26], although it does not appear there.

**Lemma 12.** *Given $n_1 + n_2 + 1$ pairs $(x_i, y_i) \in \mathbb{F}^2$, there is at most one rational function $f(X) = f_1(X)/f_2(X)$ (up to equivalence) with $f_1(X), f_2(X) \in \mathbb{F}[X]$ of degrees $n_1$ and $n_2$ respectively, that satisfies $f(x_i) = y_i$ for all $i \in [n_1 + n_2 + 1]$.*

*Proof.* Suppose $f(X)$ is a rational function satisfying the above conditions. Note that $y_i \neq 0$ for some $i \in [n_1 + n_2 + 1]$ since $f(X)$ has at most $n_1$ roots in $\mathbb{F}$. We may assume, without loss of generality, that $y_1 \neq 0$.

The rational function $g(X) = f(X + x_1)$ satisfies $g(x_i') = y_i$, where $x_i' = x_i - x_1$. In particular, we have $g(0) = f(x_1) = y_1$. Writing

$$g(X) = \frac{\sum_{i=1}^{n_1} a_i X^i}{\sum_{i=1}^{n_2} b_i X^i},$$

it follows that $a_0/b_0 = y_1$. Replacing $g(X)$ by an equivalent rational function with $b_0 = 1$, we then have $a_0 = y_1$. Consider the rational function

$$h(X) = \frac{y_1^{-1} \sum_{i=1}^{n_1} a_i X^{n_1 - i}}{\sum_{i=1}^{n_2} b_i X^{n_2 - i}}.$$

Note that $h(X)$ is monic, and

$$\begin{aligned} h\big((x_i')^{-1}\big) &= \frac{y_1^{-1} \sum_{i=1}^{n_1} a_i (x_i')^{-(n_1 - i)}}{\sum_{i=1}^{n_2} b_i (x_i')^{-(n_2 - i)}} = \frac{y_1^{-1} \sum_{i=1}^{n_1} a_i (x_i')^{i}}{(x_i')^{n_1 - n_2} \sum_{i=1}^{n_2} b_i (x_i')^{i}} \\ &= (x_i')^{n_2 - n_1} y_1^{-1} g(x_i') = (x_i')^{n_2 - n_1} y_1^{-1} y_i \end{aligned}$$

for all $i \in [n_1 + n_2 + 1] \setminus \{1\}$.

Now, by Lemma 11, there is at most one such rational function $h(X)$, which proves the uniqueness of $g(X)$, and hence the uniqueness of $f(X)$. □

### 6.4.1 Correctness

**Proposition 13.** *In Protocol 4, $Q$ outputs $S_1 \setminus S_2$ except with probability negligible in $\lambda$.*

*Proof.* Since $E$ is additively homomorphic, for each $i \in [3n + 1]$, $\mathsf{ct}_{0,i}$ decrypts to

$$\beta_i(r_1(\alpha_i) p_{S_1}(\alpha_i) + r_2(\alpha_i) p_{S_2}(\alpha_i)).$$

Therefore, by the uniqueness in Lemma 12, we have

$$\begin{aligned} f(X) &= \frac{r_1(X) p_{S_1}(X) + r_2(X) p_{S_2}(X)}{p_{S_1}(X)} \\ &= \frac{r_1(X) p_{S_1 \setminus S_2}(X) p_{S_1 \cap S_2}(X) + r_2(X) p_{S_2 \setminus S_1}(X) p_{S_1 \cap S_2}(X)}{p_{S_1 \setminus S_2}(X) p_{S_1 \cap S_2}(X)} \\ &= \frac{r_1(X) p_{S_1 \setminus S_2}(X) + r_2(X) p_{S_2 \setminus S_1}(X)}{p_{S_1 \setminus S_2}(X)}. \end{aligned}$$

By Lemmas 9 and 10, over the random choices of $r_1(X)$ and $r_2(X)$, the polynomials $r_1(X) p_{S_1 \setminus S_2}(X) + r_2(X) p_{S_2 \setminus S_1}(X)$ and $p_{S_1 \setminus S_2}(X)$ are coprime except with probability negligible in $\lambda$. Hence, in step 9, we have $q_2(X) = p_{S_1 \setminus S_2}(X)$ (up to multiplication by an element of $\mathbb{F}^*$), from which the correctness of the protocol follows. □

From the proof of Proposition 13, we note that Protocol 4 always produces a correct output unless $\gcd\big(p_{S_1 \setminus S_2}(X), r(X)\big) \neq 1$, where $r(X) = r_1(X) p_{S_1 \setminus S_2}(X) + r_2(X) p_{S_2 \setminus S_1}(X)$.

Let $d = \deg(p_{S_1 \setminus S_2})$. Then $r(X)$ is a uniformly random polynomial of degree $\leq n+d$. An analysis following the same argument as in the proof of Lemma 10 shows that

$$\Pr\big[\gcd\big(p_{S_1\setminus S_2}(X), r(X)\big) \neq 1\big] \leq \frac{d}{|\mathbb{F}|} \leq \frac{n}{|\mathbb{F}|}.$$

For example, in order to obtain an error probability that is less than $2^{-40}$, it suffices to pick $\mathbb{F}$ such that $\log|\mathbb{F}| > 40 + \log n$.

### 6.4.2 Security

**Proposition 14.** *Protocol 4 is secure against a semi-honest $P_1$.*

*Proof.* This is clear since the only message received by $P_1$ is the public key $\mathsf{pk}$, which does not depend on $S_2$. □

**Proposition 15.** *Assume that $E$ is IND-CPA secure. Then Protocol 4 is secure against a semi-honest $P_2$.*

*Proof.* The only messages that $P_2$ receives are $\mathsf{ct}_{1,i} = E(\mathsf{pk}, \beta_i r_1(\alpha_i) p_{S_1}(\alpha_i))$ for $i \in [3n+1]$. Since $E$ is IND-CPA secure, we can simulate these by encryptions of any $3n+1$ arbitrarily chosen plaintexts. □

**Proposition 16.** *Assume that $E$ has input privacy. Then Protocol 4 is secure against a semi-honest $Q$.*

*Proof.* In a real protocol execution, $n_i = \beta_i(r_1(\alpha_i)p_{S_1}(\alpha_i) + r_2(\alpha_i)p_{S_2}(\alpha_i))$ and $d_i = \beta_i p_{S_1}(\alpha_i)$. Since $\beta_i \in \mathbb{F}$ is chosen uniformly at random, $n_i$ and $d_i$ are uniformly random elements that satisfy

$$\begin{aligned}\frac{n_i}{d_i} &= \frac{r_1(\alpha_i)p_{S_1}(\alpha_i) + r_2(\alpha_i)p_{S_2}(\alpha_i)}{p_{S_1}(\alpha_i)} \\ &= \frac{r_1(\alpha_i)p_{S_1\setminus S_2}(\alpha_i) + r_2(\alpha_i)p_{S_2\setminus S_1}(\alpha_i)}{p_{S_1\setminus S_2}(\alpha_i)}.\end{aligned}$$

By Lemma 9, $r_1(X)p_{S_1\setminus S_2}(X) + r_2(X)p_{S_2\setminus S_1}(X)$ is a uniformly random polynomial of degree $\leq n + |S_1 \setminus S_2|$. Hence, the simulator can simulate the messages received by $Q$ as follows. First, it picks a random polynomial $u(X)$ of degree $\leq n + |S_1 \setminus S_2|$ uniformly at random. It then picks elements $\gamma_1, \ldots, \gamma_{3n+1} \in \mathbb{F}$ uniformly at random, sets $d_i = \gamma_i p_{S_1\setminus S_2}(\alpha_i)$ and $\mathsf{ct}_{0,i}$ to be the result of applying homomorphic addition to $E(\mathsf{pk}, \gamma_i u(\alpha_i))$ and $E(\mathsf{pk}, 0)$. By input privacy, this interaction is indistinguishable from the real interaction. □

## 6.5 Implementation considerations

Protocol 4 requires an additively homomorphic encryption scheme $E : \mathbb{F} \to \mathcal{C}$ with plaintext space a finite field. However, the plaintext spaces for many additively homomorphic encryption schemes are rings (but not fields).

For example, the Paillier cryptosystem [45] has plaintext space $\mathbb{Z}/N\mathbb{Z}$, where $N$ is a product of two large primes. One possible way to make the Paillier cryptosystem work with our scheme is to choose $\mathbb{F}$ to be a finite prime field $\mathbb{F}_p$, and use the natural set inclusion $\mathbb{F}_p \hookrightarrow \mathbb{Z}/N$ to identify $\mathbb{F}_p$ as a subset of $\mathbb{Z}/N\mathbb{Z}$ (this map is, however, not a ring homomorphism).

In step 5 of Protocol 4, $P_2$ performs a homomorphic addition on two ciphertexts $\mathsf{ct}_{1,i}$, $\mathsf{ct}_{2,i}$, and the result $\mathsf{ct}_{0,i}$ is then sent to $Q$ for decryption. If $N \geq 2p-1$, the ciphertext received by $Q$ can indeed be decrypted to give the correct result in $\mathbb{F}_p$.

However, this approach does leak a small amount of information since it does reveal whether or not the two summands, when viewed as integers, sum to a value $< p$ or to a value $\geq p$. Therefore, while we are not aware that this translates into an actual attack of the protocol, we cannot recommend the use of the Paillier cryptosystem for our protocol.

Instead, although we only require an additively homomorphic encryption scheme, a better solution is to use a fully homomorphic encryption scheme for $E$.

For example, the BGV scheme [8], introduced by Brakerski, Gentry and Vaikuntanathan, has plaintext space $\mathbb{Z}_p[X]/\Phi_d(X)$, where $p$ is prime and $\Phi_d(X)$ is the $d$-th cyclotomic polynomial. Let $k$ be the order of $p$ modulo $d$ and let $\varphi$ be Euler's totient function. Then $\Phi_d(X)$ factorizes into $\varphi(d)/k$ distinct irreducible factors $s_1(X), \ldots, s_{d/k}(X)$, each of degree $k$, in $\mathbb{Z}_p[X]$. Hence, by the Chinese remainder theorem,

$$\frac{\mathbb{Z}_p[X]}{\Phi_d(X)} \cong \prod_{i=1}^{\varphi(d)/k} \frac{\mathbb{Z}_p[X]}{(s_i(X))} \approx \prod_{i=1}^{\varphi(d)/k} \mathbb{F}_{p^k}$$

is a product of finite fields.

Since our protocol does not require multiplication of ciphertexts, BGV should have reasonably good performance in practice. Furthermore, as the plaintext space of BGV is a product of $\varphi(d)/k$ copies of $\mathbb{F}_{p^k}$, a single homomorphic addition performed using BGV corresponds to addition of $\varphi(d)/k$ pairs of ciphertexts in step 5 of Protocol 4.

While BGV does not have input privacy in general, homomorphic addition of ciphertexts in BGV is input private, and thus using it does not affect the security of Protocol 4.

# 7 Third-Party Private Symmetric Difference via Rational Functions

## 7.1 An overview

In this section, we will present our third-party private symmetric difference (TP-PSymD) protocol that is based on rational functions.

Since $S_1 \triangle S_2 = (S_1 \setminus S_2) \cup (S_2 \setminus S_1)$, using the ideas introduced in Protocol 4, one obvious way to obtain a TP-PSymD protocol is as follows. First, we pick random polynomials $r_1(X)$ and $r_2(X)$, and have $P_1$ and $P_2$ jointly compute encrypted evaluations of the rational function $f_1(X) = \frac{r_1(X)p_{S_1}(X) + r_2(X)p_{S_2}(X)}{p_{S_1}(X)}$, which has poles exactly at the elements of $S_1 \setminus S_2$. Next, using the same evaluation points, they

jointly compute encrypted evaluations of the rational function $f_2(X) = \frac{r_3(X)p_{S_1}(X)+r_4(X)p_{S_2}(X)}{p_{S_2}(X)}$, where $r_3(X)$ and $r_4(X)$ are again random polynomials. The encrypted evaluations of $f_1(X)$ and $f_2(X)$ (at the same evaluation point) are then homomorphically multiplied before they are sent to $Q$, who can use them to reconstruct a rational function which, with high probability, has poles exactly at $S_1 \triangle S_2$.

While the above indeed correctly and privately computes the symmetric difference, it is expensive in practice as it requires the use of a fully homomorphic encryption scheme to perform both addition and multiplication of ciphertexts.

In order to achieve an efficient TP-PSymD protocol, we would like to avoid the use of fully homomorphic encryption. Hence, we design a different rational function $f(X) = \frac{r_1(X)p_{S_1}^2(X)+r_2(X)p_{S_2}^2(X)}{p_{S_1}(X)p_{S_2}(X)}$ (where $r_1(X)$ and $r_2(X)$ are random polynomials) which has, with high probability, poles exactly at the elements of $S_1 \triangle S_2$. By having $P_1$ and $P_2$ provide $Q$ with evaluations of $f(X)$ at various $\alpha_i$'s, $Q$ can reconstruct $f(X)$, and hence determine $S_1 \triangle S_2$.

Observe that, due to the design of the rational function $f(X)$, fully homomorphic encryption is not needed to allow $Q$ to obtain encrypted evaluations of $f(X)$. Indeed, $P_1$ and $P_2$ can jointly compute evaluations of the numerator of $f(X)$ using an additively homomorphic encryption scheme, while they can jointly compute evaluations of the denominator using a multiplicatively homomorphic encryption scheme. Thus, by using two different homomorphic encryption schemes (which are, respectively, additively and multiplicatively homomorphic), we avoid the need to use an expensive fully homomorphic encryption scheme.

## 7.2 Details of the protocol

As in Section 6, let $S_1 = \{s_1, \ldots, s_n\} \subseteq \{0,1\}^\ell$, $S_2 = \{t_1, \ldots, t_n\} \subseteq \{0,1\}^\ell$, and $\lambda > 0$ be the correctness parameter. We embed $\{0,1\}^\ell$ into a finite field $\mathbb{F}$ with $|\mathbb{F}| \geq \max\left(2^\ell + 6n + 1, 2^\lambda\right)$ using a map $\iota : \{0,1\}^\ell \hookrightarrow \mathbb{F}$, and let $S$ be the image of $\{0,1\}^\ell$ under $\iota$. Fix $6n+1$ points $\alpha_1, \ldots, \alpha_{6n+1} \in \mathbb{F} \setminus S$, and let $E_a : \mathbb{F} \to \mathcal{C}$ (respectively, $E_m : \mathbb{F} \to \mathcal{C}'$) be an asymmetric additively homomorphic (respectively, multiplicatively homomorphic) encryption scheme with input privacy.

1. $Q$ generates key pairs $(\mathsf{sk}_a, \mathsf{pk}_a)$ and $(\mathsf{sk}_m, \mathsf{pk}_m)$ for $E_a$ and $E_m$ respectively, and sends $\mathsf{pk}_a$ and $\mathsf{pk}_m$ to $P_1$ and $P_2$.
2. $P_1$ and $P_2$ pick $\beta_1, \ldots, \beta_{6n+1} \in \mathbb{F}$ uniformly at random.
3. $P_1$ chooses a random polynomial $r_1(X)$ of degree $\leq 2n$, computes $\mathsf{ct}_{1,i} = E_a\left(\mathsf{pk}_a, \beta_i r_1(\alpha_i) p_{S_1}^2(\alpha_i)\right)$ and sends $\mathsf{ct}_{1,i}$ to $P_2$ for $i \in [6n+1]$.
4. $P_2$ chooses a random polynomial $r_2(X)$ of degree $\leq 2n$ and computes $\mathsf{ct}_{2,i} = E\left(\mathsf{pk}_a, \beta_i r_2(\alpha_i) p_{S_2}^2(\alpha_i)\right)$ for $i \in [6n+1]$.
5. For each $i \in [6n+1]$, $P_2$ performs homomorphic addition on $\mathsf{ct}_{1,i}$ and $\mathsf{ct}_{2,i}$ to obtain a ciphertext $\mathsf{ct}_{0,i}$, and sends $\mathsf{ct}_{0,i}$ to $Q$.
6. $P_1$ computes $\mathsf{ct}'_{1,i} = E_m(\mathsf{pk}_m, \beta_i p_{S_1}(\alpha_i))$ and sends $\mathsf{ct}'_{1,i}$ to $P_2$ for $i \in [6n+1]$.
7. $P_2$ computes $\mathsf{ct}'_{2,i} = E_m(\mathsf{pk}_m, p_{S_2}(\alpha_i))$ for $i \in [6n+1]$.
8. For each $i \in [6n+1]$, $P_2$ performs homomorphic multiplication on $\mathsf{ct}'_{1,i}$ and $\mathsf{ct}'_{2,i}$ to obtain a ciphertext $\mathsf{ct}'_{0,i}$, and sends $\mathsf{ct}'_{0,i}$ to $Q$.
9. For each $i \in [6n+1]$, $Q$ decrypts $\mathsf{ct}_{0,i}$ and $\mathsf{ct}'_{0,i}$ to obtain $n_i$ and $d_i$ respectively.
10. $Q$ uses rational function interpolation to find the unique rational function $f(X)$ with numerator of degree $\leq 4n$ and denominator of degree $\leq 2n$ such that $f(\alpha_i) = n_i/d_i$ for $i \in [6n+1]$.
11. Let $f(X) = q_1(X)/q_2(X)$ where $q_1(X)$ and $q_2(X)$ are coprime polynomials. $Q$ outputs
$$\{\iota^{-1}(x) : x \in \mathbb{F} \text{ such that } q_2(x) = 0\}.$$

Protocol 5: A semi-honest TP-PSymD protocol

## 7.3 Communication and computational complexity

Using an analysis similar to that in Section 6.3, we can show that Protocol 5 has the same communication and computational complexity as Protocol 4, i.e. its communication complexity is $O\left(n \log(2^\ell + n + 2^\lambda)\right)$, which is quasilinear in $n$, and its computational complexity is

$$O\left(n \log(n \log|\mathbb{F}|) \log n \log^2|\mathbb{F}| + n^{\omega+o(1)} \log|\mathbb{F}| \log\log|\mathbb{F}|\right),$$

which is $O(n^{\omega+o(1)})$ as a function of $n$.

## 7.4 Correctness and security

We now state the correctness and security guarantees of Protocol 5.

### 7.4.1 Correctness

**Proposition 17.** *In Protocol 5, $Q$ outputs $S_1 \triangle S_2$ except with probability negligible in $\lambda$.*

*Proof.* By Lemma 12,

$$\begin{aligned} f(X) &= \frac{r_1(X)p_{S_1}^2(X) + r_2(X)p_{S_2}^2(X)}{p_{S_1}(X)p_{S_2}(X)} \\ &= \frac{r_1(X)p_{S_1\setminus S_2}^2(X)p_{S_1\cap S_2}^2(X) + r_2(X)p_{S_2\setminus S_1}^2(X)p_{S_1\cap S_2}^2(X)}{p_{S_1\setminus S_2}(X)p_{S_2\setminus S_1}(X)p_{S_1\cap S_2}^2(X)} \\ &= \frac{r_1(X)p_{S_1\setminus S_2}^2(X) + r_2(X)p_{S_2\setminus S_1}^2(X)}{p_{S_1\setminus S_2}(X)p_{S_2\setminus S_1}(X)}. \end{aligned}$$

By Lemmas 9 and 10, $r_1(X)p_{S_1\setminus S_2}^2(X) + r_2(X)p_{S_2\setminus S_1}^2(X)$ and $p_{S_1\setminus S_2}(X)p_{S_2\setminus S_1}(X)$ are coprime except with probability negligible in $\lambda$. Hence, in step 9, we have $q_2(X) = p_{S_1\setminus S_2}(X)p_{S_2\setminus S_1}(X) = p_{S_1\triangle S_2}(X)$ (up to multiplication by an element of $\mathbb{F}^*$), as required. □

#### 7.4.2 Security

**Proposition 18.** *Protocol 5 is secure against a semi-honest $P_1$.*

*Proof.* This is clear since $\mathsf{pk}_a$ and $\mathsf{pk}_m$ are independent of $S_2$. □

**Proposition 19.** *Assume that $E_a$ and $E_m$ are IND-CPA secure. Then Protocol 5 is secure against a semi-honest $P_2$.*

*Proof.* The only messages that $P_2$ receives are $\mathsf{ct}_{1,i} = E_a\left(\mathsf{pk}_a, \beta_i r_1(\alpha_i)p_{S_1}^2(\alpha_i)\right)$ and $\mathsf{ct}'_{1,i} = E_m(\mathsf{pk}_m, \beta_i p_{S_1}(\alpha_i))$ for $i \in [6n+1]$. Since both $E_a$ and $E_m$ are IND-CPA secure, we can simulate these by encryptions of any $6n+1$ arbitrarily chosen plaintexts under $E_a$ and $E_m$ respectively. □

**Proposition 20.** *Assume that $E_a$ and $E_m$ have input privacy. Then Protocol 5 is secure against a semi-honest $Q$.*

*Proof.* We use a similar argument as in the proof of Proposition 16. Observe that, in the real interaction, $n_i$ and $d_i$ are uniformly random elements that satisfy

$$\begin{aligned} \frac{n_i}{d_i} &= \frac{r_1(\alpha_i)p_{S_1}^2(\alpha_i) + r_2(\alpha_i)p_{S_2}^2(\alpha_i)}{p_{S_1}(\alpha_i)p_{S_2}(\alpha_i)} \\ &= \frac{r_1(\alpha_i)p_{S_1\setminus S_2}^2(\alpha_i) + r_2(\alpha_i)p_{S_2\setminus S_1}^2(\alpha_i)}{p_{S_1\triangle S_2}(\alpha_i)}. \end{aligned}$$

Since $r_1(X)p_{S_1\setminus S_2}^2(X) + r_2(X)p_{S_2\setminus S_1}^2(X)$ is a uniformly random polynomial of degree $\leq 2(n + |S_1 \setminus S_2|)$ by Lemma 9, we can simulate the messages received by $Q$ as follows. First, pick a uniformly random polynomial $u(X)$ of degree $\leq 2(n+|S_1 \setminus S_2|)$ and pick uniformly random elements $\gamma_1, \ldots, \gamma_{6n+1} \in \mathbb{F}$. Let $\mathsf{ct}_{0,i}$ be the result of homomorphically adding $E_a(\mathsf{pk}_a, \gamma_i u(\alpha_i))$ and $E_a(\mathsf{pk}_a, 0)$, and $\mathsf{ct}'_{0,i}$ be the result of homomorphically multiplying $E_m(\mathsf{pk}_m, \gamma_i p_{S_1\triangle S_1}(\alpha_i))$ and $E_m(\mathsf{pk}_m, 1)$. By input privacy, this is indistinguishable from the real interaction. □

### 7.5 Implementation considerations

As in Protocol 4, we can use BGV for the additively homomorphic encryption scheme $E_a$. As for the multiplicatively homomorphic encryption scheme $E_m$, we can use ElGamal [18] over a finite field $\mathbb{K}$. For ElGamal to be secure, $|\mathbb{K}|$ needs to be large (for example, 3072 bits). However, in most cases, the field $\mathbb{F}$ used in Protocol 5 will be much smaller, and hence we must choose $\mathbb{K}$ to be a suitably large extension field of $\mathbb{F}$.

Another possible option is to also use BGV for the multiplicatively homomorphic encryption scheme $E_m$, and this has the advantage of being quantum-safe. Since BGV does not provide input privacy by default, it is necessary to augment it. One possible way to do so is to use noise flooding [25] to achieve circuit privacy. Augmenting BGV for input privacy, however, does result in an additional performance penalty.

## 8 Performance Evaluation

For multi-party third party PSI, we implemented Protocol 3, which is secure against arbitrary collusion, in C++ using the NTL library [56]. The NTL library provides various classes (`ZZ_p` and `ZZ_pE`) that support arithmetic over any finite field. However, for performance reasons, we use the `zz_p` class that only supports modular arithmetic for a modulus $p$ of up to 60 bits. For the implementation of Protocol 3, we used 32-bit elements and target an error probability of $< 2^{-40}$.

We give an outline of some of the optimizations used in our implemention of the protocol. First, we implemented a fast polynomial interpolation algorithm with quasilinear complexity. This is essential as the built-in polynomial interpolation from the NTL library quickly becomes impractical as it has complexity $O(n^2)$.

As for the OPRFs required by the protocol, we use the multi-point OPRF protocol introduced by Chase and Miao [13], which mostly uses symmetric key, bitwise operations and hashing, hence is very efficient in practice.

Next, in order to optimize the root finding step, we choose the prime $p = 180143985094819841 = 5 \cdot 2^{55} + 1$, which allows us to apply the tangent Graeffe method for root finding. The tangent Graeffe method is introduced by Grenet et al. [28] with important practical improvements by van der Hoeven and Monagan [58, 59]. This root finding method works over finite fields $\mathbb{F}_p$ for primes $p$ of the form $\sigma 2^k + 1$ with $\sigma$ small.[4] While the complexity of the tangent Graeffe method is quasilinear just like the Cantor-Zassenhaus algorithm, both its complexity and practical performance are superior to Cantor-Zassenhaus when $p$ is of this prescribed form.

For third party private set difference and symmetric difference, we implement Protocols 4 and 5 using the NTL library and the HElib library [29]. The BGV encryption scheme is

[4] It is an open problem whether it is always possible to find a suitable prime of this form that is arbitrarily large. However, for practical purposes, this question is of little importance.

| Party Size $N$ | Set Size $n$ | Output Size $t$ | Running Time (s) | | | | Comm. Cost (MB) |
|---|---|---|---|---|---|---|---|
| | | | OPRF Eval. | Inter-polation | Root Finding | Total | |
| 2 | $2^{18}$ | 0 | 2.06 | 3.47 | 3.40 | 9.10 | 46.48 |
| | | $2^{17}$ | 2.08 | 3.43 | 5.68 | 11.36 | 46.48 |
| | | $2^{18}$ | 2.13 | 3.48 | 7.64 | 13.43 | 46.48 |
| | $2^{20}$ | 0 | 7.36 | 16.72 | 16.52 | 41.25 | 187.29 |
| | | $2^{19}$ | 7.34 | 16.76 | 25.40 | 50.06 | 187.29 |
| | | $2^{20}$ | 7.35 | 16.76 | 32.75 | 57.45 | 187.29 |
| | $2^{22}$ | 0 | 29.95 | 81.90 | 81.78 | 195.65 | 755.04 |
| | | $2^{21}$ | 29.47 | 82.05 | 115.12 | 228.72 | 755.04 |
| | | $2^{22}$ | 29.62 | 81.85 | 140.79 | 254.33 | 755.04 |
| 3 | $2^{18}$ | 0 | 2.49 | 3.45 | 3.39 | 9.62 | 127.45 |
| | | $2^{17}$ | 2.55 | 3.40 | 5.54 | 11.77 | 127.45 |
| | | $2^{18}$ | 2.64 | 3.43 | 7.63 | 14.00 | 127.45 |
| | $2^{20}$ | 0 | 8.93 | 16.80 | 16.49 | 43.11 | 513.88 |
| | | $2^{19}$ | 8.87 | 16.76 | 24.81 | 51.38 | 513.88 |
| | | $2^{20}$ | 8.66 | 16.73 | 32.67 | 58.96 | 513.88 |
| | $2^{22}$ | 0 | 35.49 | 81.70 | 81.73 | 202.72 | 2073.13 |
| | | $2^{21}$ | 35.85 | 83.57 | 114.76 | 235.75 | 2073.13 |
| | | $2^{22}$ | 37.25 | 82.26 | 140.59 | 263.40 | 2073.13 |
| 4 | $2^{18}$ | 0 | 2.97 | 3.47 | 3.40 | 10.20 | 246.89 |
| | | $2^{17}$ | 3.06 | 3.47 | 5.59 | 12.48 | 246.89 |
| | | $2^{18}$ | 3.04 | 3.42 | 7.65 | 14.45 | 246.89 |
| | $2^{20}$ | 0 | 10.99 | 16.71 | 16.55 | 45.60 | 995.76 |
| | | $2^{19}$ | 11.35 | 16.73 | 24.80 | 54.05 | 995.76 |
| | | $2^{20}$ | 11.09 | 16.74 | 32.66 | 61.79 | 995.76 |
| | $2^{22}$ | 0 | 42.68 | 81.73 | 82.02 | 211.66 | 4018.26 |
| | | $2^{21}$ | 45.58 | 81.47 | 112.63 | 245.12 | 4018.26 |
| | | $2^{22}$ | 46.54 | 81.85 | 140.02 | 273.51 | 4018.26 |
| 6 | $2^{18}$ | 0 | 3.80 | 3.43 | 3.41 | 11.23 | 601.23 |
| | | $2^{17}$ | 3.78 | 3.43 | 5.58 | 13.35 | 601.23 |
| | | $2^{18}$ | 3.82 | 3.41 | 7.70 | 15.50 | 601.23 |
| | $2^{20}$ | 0 | 14.18 | 16.85 | 16.30 | 49.43 | 2425.42 |
| | | $2^{19}$ | 14.22 | 16.76 | 24.66 | 57.77 | 2425.42 |
| | | $2^{20}$ | 14.20 | 16.80 | 32.45 | 65.44 | 2425.42 |
| | $2^{22}$ | 0 | 60.40 | 82.15 | 80.68 | 232.01 | 9789.66 |
| | | $2^{21}$ | 60.58 | 81.84 | 111.65 | 262.99 | 9789.66 |
| | | $2^{22}$ | 60.57 | 81.90 | 138.34 | 289.63 | 9789.66 |

Table 1: Running times and communication costs of Protocol 3 for $N = 2,3,4,6$

| Party Size $N$ | Set Size $n$ | Output Size $t$ | Running Time (s) | | | | Comm. Cost (MB) |
|---|---|---|---|---|---|---|---|
| | | | OPRF Eval. | Inter-polation | Root Finding | Total | |
| 8 | $2^{18}$ | 0 | 5.28 | 3.51 | 3.41 | 12.95 | 1109.49 |
| | | $2^{17}$ | 5.29 | 3.60 | 5.59 | 15.17 | 1109.49 |
| | | $2^{18}$ | 5.27 | 3.46 | 7.66 | 17.16 | 1109.49 |
| | $2^{20}$ | 0 | 19.88 | 16.93 | 16.27 | 55.97 | 4476.24 |
| | | $2^{19}$ | 19.89 | 16.75 | 24.72 | 64.34 | 4476.24 |
| | | $2^{20}$ | 19.88 | 16.74 | 32.52 | 72.13 | 4476.24 |
| | $2^{22}$ | 0 | 84.92 | 81.97 | 80.69 | 260.02 | 18069.28 |
| | | $2^{21}$ | 84.88 | 81.88 | 111.93 | 291.17 | 18069.28 |
| | | $2^{22}$ | 84.84 | 81.97 | 138.78 | 318.43 | 18069.28 |

Table 2: Running times and communication costs of Protocol 3 for $N = 8$

| Set Size $n$ | Output Size $t$ | Running Time (s) | | | | Comm. Cost (MB) |
|---|---|---|---|---|---|---|
| | | Cipher Operations | Interpolation | Root Finding | Total | |
| $2^{10}$ | 0 | 0.39 | 7.27 | 0 | 13.47 | 0.99 |
| | $2^{9}$ | 0.39 | 7.27 | 0.05 | 13.52 | 0.99 |
| | $2^{10}$ | 0.39 | 6.34 | 0.12 | 12.66 | 0.99 |
| $2^{11}$ | 0 | 0.78 | 55.25 | 0 | 62.03 | 1.34 |
| | $2^{10}$ | 0.78 | 54.79 | 0.12 | 61.70 | 1.34 |
| | $2^{11}$ | 0.78 | 47.21 | 0.30 | 54.31 | 1.34 |
| $2^{12}$ | 0 | 1.18 | 430.28 | 0 | 438.29 | 1.71 |
| | $2^{11}$ | 1.18 | 425.39 | 0.30 | 433.63 | 1.71 |
| | $2^{12}$ | 1.17 | 364.38 | 0.71 | 373.05 | 1.71 |

Table 3: Running times and communication costs of Protocol 4

| Set Size $n$ | Output Size $t$ | Running Time (s) | | | | Comm. Cost (MB) |
|---|---|---|---|---|---|---|
| | | Cipher Operations | Interpolation | Root Finding | Total | |
| $2^{9}$ | 0 | 50.16 | 7.26 | 0 | 63.10 | 5.94 |
| | $2^{8}$ | 50.24 | 7.39 | 0.02 | 63.31 | 5.94 |
| | $2^{9}$ | 49.99 | 7.27 | 0.05 | 62.99 | 5.94 |
| $2^{10}$ | 0 | 99.82 | 55.15 | 0 | 160.83 | 11.23 |
| | $2^{9}$ | 100.02 | 55.84 | 0.05 | 161.76 | 11.23 |
| | $2^{10}$ | 99.74 | 54.65 | 0.12 | 160.38 | 11.23 |
| $2^{11}$ | 0 | 199.72 | 428.58 | 0 | 634.85 | 21.49 |
| | $2^{10}$ | 199.72 | 433.25 | 0.12 | 639.66 | 21.49 |
| | $2^{11}$ | 199.43 | 423.35 | 0.29 | 629.65 | 21.49 |

Table 4: Running times and communication costs of Protocol 5

chosen as the additively homomorphic encryption scheme for both protocols and the ElGamal encryption scheme is chosen as the multiplicatively homomorphic encryption scheme for Protocol 5. Our implementations of both protocols use 32-bit elements and have an error probability of $< 2^{-40}$.

We briefly discuss the choice of parameters for our implementation. First, an ideal choice of $p$ for the finite field $\mathbb{F}_p$ will have $\approx 60$ bits to allow the use of the `zz_p` class, and yet result in a small probability of error for the protocols.

Recall that the BGV scheme has plaintext space $\mathbb{Z}_p[X]/\Phi_d(X)$, where $\Phi_d(X)$ is the $d$-th cyclotomic polynomial. For the BGV scheme, we would like to choose parameters $p$ and $d$ that provide at least 128 bits of security and has a large number of plaintext slots (which will mean that each ciphertext addition corresponds to many plaintext additions).

The level of security is roughly determined by the size of $d$, and we require a value of $d$ that is in the thousands to get our desired level of security. Since the number of plaintext slots is equal to $\varphi(d)/k$, where $k$ is the order of $p$ modulo $d$, we would like $k$ to be small. One way to achieve this is to choose $d$ such that $p \equiv 1 \pmod{d}$, i.e. such that $d \mid (p-1)$. Hence, we search for values of $p$ such that $p-1$ has a factor $d$ that is roughly of the correct size.

Next, for ElGamal, we would like to use an extension field $\mathbb{F}_{p^\alpha}$ of $\mathbb{F}_p$ that has size at least 3072 bits. Based on the size of $p$, we choose $\alpha = 53$. In order to efficiently check if an element $g \in \mathbb{F}^*_{p^{53}}$ is a generator for the group, it is necessary to know the prime factorization of $p^{53}-1 = (p-1)(p^{52}+p^{51}+\cdots+p+1)$, which in general, is a difficult computational problem. We solve this by choosing $p$ such that $\frac{p^{53}-1}{p-1}$ is prime. Based on all the above considerations, we perform a search for suitable values of $p$ and $d$ which yields the choice of $p = 576460752303765851$ and $d = 7015$.

Our benchmarks for Protocol 3 are run on two identical machines, each with two Intel Xeon Silver 4214 processors.[5] For 4 or fewer input parties, all parties (including $Q$) run on a single machine. For larger numbers of input parties, parties $P_i$ for $i \geq 5$ run on the second machine. The input parties $P_i$ are assigned four cores each while $Q$ runs on a single core.

For Protocols 4 and 5, a single machine is used. All parties run on a single core for Protocol 4, while for Protocol 5, each party (inclusive of the third-party) is assigned four cores.

The total running times and communication costs of Protocols 3, 4 and 5 are presented in Tables 1–2, Table 3 and Table 4 respectively. The times taken for several key steps are also presented. The results are recorded based on an average of 5 runs. The number of parties, not including the receiver $Q$, is denoted as $N$, the set size of each input party is denoted as $n$, and the output size is denoted as $t$ (set at 50% and 100% of $n$).

The TP-PSI works of [63] and [64] are more theoretical in nature and do not come with any implementations or benchmarks. As such, we are unable to perform experimental comparisons with their works. Nevertheless, in the case of two input parties, we expect Protocol 3 to significantly outperform all these previous protocols due to their heavy use of public key operations. Indeed, all previous protocols require at least $2n$ public key operations (such as exponentiations), where $n$ is the size of each input dataset.

[5] Each processor has 12 cores and a base frequency of 2.20 GHz.

| Party Size $N$ | Set Size $n$ | Running Time (s) | Comm. Cost (MB) |
|---|---|---|---|
| 3 | $2^{18}$ | 11.10 | 556.45 |
| | $2^{20}$ | 43.66 | 2225.59 |
| | $2^{22}$ | 170.77 | 9582.31 |
| 4 | $2^{18}$ | 12.49 | 1252.00 |
| | $2^{20}$ | 46.87 | 5007.58 |
| | $2^{22}$ | 186.64 | 21560.20 |
| 6 | $2^{18}$ | 19.74 | 3477.77 |
| | $2^{20}$ | 76.71 | 13909.92 |
| | $2^{22}$ | 307.49 | 59889.42 |

Table 5: Running times and comm. costs of [39]

To better understand the performance of Protocol 3 in the context of other works, we will instead look at a state-of-the-art semi-honest multi party PSI protocol in the conventional setting that is secure against the maximal number of corrupt parties. Some of these works include [39], [5] and [60]. [5] is designed for the specific scenario of a large number of parties and small input set sizes, while [60] is optimized for the special case where the universe of all possible elements is small. Since both the use cases of [5] and [60] differ from ours, [39] emerged as the most suitable work to compare our protocol to. As above, we assign each party four cores and we record the results based on an average of 5 runs. We run the benchmark only for $N = 3, 4, 6$ as the code accompanying [39] does not support multiple machines. The results are presented in Table 5.

We note that the running times of Protocol 3 are competitive with the protocol in [39]. Indeed, although our protocol works in the slightly different third-party setting, which as explained, limits the breadth of techniques that we can apply and thus has the potential to adversely affect efficiency, all the run times of Protocol 3 are within $1.55\times$ of those listed in Table 5 that are achieved by [39]. Furthermore, for larger number of parties $N$ and smaller set sizes $n$, Protocol 3 actually achieves superior performance compared to [39]. The communication costs of Protocol 3 are also significantly better than that of [39].

We interpret the experimental results in the context of the previously described applications, such as for investigating cyber intrusions. Indeed, the results in Tables 1–2 demonstrate the practicality of Protocol 3 for input sizes of up to several million, which is more than sufficient for these applications.

As our protocol achieves performance close to that of [39],

and with better performance in certain cases, our protocol is the ideal choice when the desired functionality is that of third-party PSI, due to the additional privacy advantages afforded by our solution compared to running a multi-party PSI protocol with one of the input parties as the receiver.

## 9 Conclusion

We design a series of protocols for performing privacy-preserving set operations in the third-party setting.

The various multi-party third party PSI protocols are designed to cater for different requirements, depending on variability of compute resources, communication cost and the likelihood of arbitrary collusion. Protocol 1 has low communication costs, but is secure only against certain collusions. On the other hand, Protocols 2 and 3 are secure against an arbitrary number of collusions, but comes at the cost of significantly more communication. Compared to Protocol 2, Protocol 3 significantly reduces the computational cost for $Q$ at the cost of additional communication and computation for the $P_i$'s. Overall, it greatly reduces the total computational costs. Hence all three protocols can be useful in practice depending on the specific use case.

In addition, we present protocols for private computations of the set difference functionality and the symmetric difference functionality in the third-party setting, and demonstrate the practicalities of these protocols for small sets.

## 10 Acknowledgements

We thank the shepherd and the reviewers for their valuable feedback.

## 11 Ethics Considerations

The datasets used in the experiments are randomly generated and do not infringe on confidentiality or privacy of any individual or organization. No human subjects were involved. One segment of our implementation in Protocol 3 adapts modifications of an OPRF code. We obtained approvals from the author of this code to modify and utilize in our performance benchmarking. Similarly, we also obtained approvals from the authors of the tangent Graeffe root finding code for use in our implementations.

## 12 Open Science

We are committed for our research results to be made available to the public. In addition, we are committed to openly share our research artifacts to enable reproducibility and replicability of our work. Artifacts include the source code used to benchmark Protocols 3, 4 and 5, and are available at https://zenodo.org/records/14729415.

## References

[1] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 522–539, 2021.

[2] Diego F. Aranha, Chuanwei Lin, Claudio Orlandi, and Mark Simkin. Laconic private set-intersection from pairings. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 111–124. ACM, 2022.

[3] Saikrishna Badrinarayanan, Peihan Miao, Srinivasan Raghuraman, and Peter Rindal. Multi-party threshold private set intersection with sublinear communication. In *IACR International Conference on Public-Key Cryptography*, pages 349–379. Springer International Publishing, 2021.

[4] Saikrishna Badrinarayanan, Peihan Miao, and Tiancheng Xie. Updatable private set intersection. *Proceedings on Privacy Enhancing Technologies*, (2):378–406, 2022.

[5] Aslı Bay, Zekeriya Erkin, Jaap-Henk Hoepman, Simona Samardjiska, and Jelle Vos. Practical multi-party private set intersection protocols. *IEEE Transactions on Information Forensics and Security*, 17:1–15, 2022.

[6] Aner Ben-Efraim, Olga Nissenbaum, Eran Omri, and Anat Paskin-Cherniavsky. PSImple: Practical multiparty maliciously-secure private set intersection. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pages 1098–1112. ACM, 2022.

[7] Florian Bourse, Rafaël Del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In *Advances in Cryptology – CRYPTO 2016*, pages 62–89. Springer Berlin Heidelberg, 2016.

[8] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. Association for Computing Machinery, 2012.

[9] David G. Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587–592, 1981.

[10] Nishanth Chandran, Nishka Dasgupta, Divya Gupta, Sai Lakshmi Bhavana Obbattu, Sruthi Sekar, and Akash Shah. Efficient linear multiparty PSI and extensions to circuit/quorum PSI. In *Proceedings of the 2021 ACM*

*SIGSAC Conference on Computer and Communications Security*, pages 1182–1204. ACM, 2021.

[11] Nishanth Chandran, Divya Gupta, and Akash Shah. Circuit-PSI with linear complexity via relaxed batch OPPRF. In *Proceedings on Privacy Enhancing Technologies, vol. 2022, no. 1*, pages 353–372, 2022.

[12] Melissa Chase, Sanjam Garg, Mohammad Hajiabadi, Jialin Li, and Peihan Miao. Amortizing rate-1 OT and applications to PIR and PSI. In *Theory of Cryptography: 19th International Conference, TCC 2021*, pages 126–156. Springer International Publishing, 2021.

[13] Melissa Chase and Peihan Miao. Private set intersection in the internet setting from lightweight oblivious PRF. In *Advances in Cryptology – CRYPTO 2020*, pages 34–63. Springer International Publishing, 2020.

[14] Hao Chen, Kim Laine, and Peter Rindal. Fast private set intersection from homomorphic encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1243–1255. Association for Computing Machinery, 2017.

[15] Wutichai Chongchitmate, Yuval Ishai, Steve Lu, and Rafail Ostrovsky. PSI from ring-OLE. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 531–545. ACM, 2022.

[16] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.

[17] Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: an efficient and scalable protocol. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 789–800. ACM, 2013.

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

[19] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In *Theory of Cryptography*, pages 303–324. Springer Berlin Heidelberg, 2005.

[20] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology - EUROCRYPT 2004*, pages 1–19. Springer, Berlin, Heidelberg, 2004.

[21] Gayathri Garimella, Payman Mohassel, Mike Rosulek, Saeed Sadeghian, and Jaspal Singh. Private set operations from oblivious switching. In *IACR International Conference on Public-Key Cryptography*, pages 591–617. Springer International Publishing, 2021.

[22] Gayathri Garimella, Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. Oblivious key-value stores and amplification for private set intersection. In *Advances in Cryptology – CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021*, pages 395–425. Springer International Publishing, 2021.

[23] Gayathri Garimella, Mike Rosulek, and Jaspal Singh. Structure-aware private set intersection, with applications to fuzzy matching. In *Advances in Cryptology – CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022*, pages 323–352. Springer Nature Switzerland, 2022.

[24] Gayathri Garimella, Mike Rosulek, and Jaspal Singh. Malicious secure, structure-aware private set intersection. In *Advances in Cryptology – CRYPTO 2023: 43rd Annual International Cryptology Conference, CRYPTO 2023*, pages 577–610. Springer Nature Switzerland, 2023.

[25] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[26] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II*, pages 3–29. Springer-Verlag, 2019.

[27] S. Dov Gordon, Carmit Hazay, and Phi Hung Le. Fully secure PSI via MPC-in-the-head. *Proceedings on Privacy Enhancing Technologies*, (3):291–313, 2022.

[28] Bruno Grenet, Joris van der Hoeven, and Grégoire Lecerf. Randomized root finding over finite FFT-fields using tangent Graeffe transforms. ISSAC '15, pages 197–204. Association for Computing Machinery, 2015.

[29] Shai Halevi and Victor Shoup. Algorithms in HElib. In *Advances in Cryptology – CRYPTO 2014*, pages 554–571. Springer Berlin Heidelberg, 2014.

[30] David Harvey and Joris van der Hoeven. Faster polynomial multiplication over finite fields using cyclotomic coefficient rings. *Journal of Complexity*, 54, 2019.

[31] David Harvey and Joris van der Hoeven. Integer multiplication in time $O(n \log n)$. *Annals of Mathematics*, 193:563–617, 2021.

[32] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In *IACR international workshop on public key cryptography*, pages 175–203. Springer Berlin Heidelberg, 2017.

[33] Jingwei Hu, Junyan Chen, Wangchen Dai, and Huaxiong Wang. Fully homomorphic encryption-based protocols for enhanced private set intersection functionalities. *IACR Cryptology ePrint Archive, Paper 2023/1407*, 2023. https://eprint.iacr.org/2023/1407.

[34] Roi Inbar, Eran Omri, and Benny Pinkas. Efficient scalable multiparty private set-intersection via garbled bloom filters. In *International Conference on Security and Cryptography for Networks*, pages 235–252. Springer International Publishing, 2018.

[35] Ferhat Karakoç and Alptekin Küpçü. Linear complexity private set intersection for secure two-party protocols. In *International Conference on Cryptology and Network Security*, pages 409–429. Springer International Publishing, 2020.

[36] Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM Journal on Computing*, 40(6):1767–1802, 2011.

[37] Lea Kissner and Dawn Song. Privacy-preserving set operations. In *Advances in Cryptology – CRYPTO 2005*, pages 241–257. Springer Berlin Heidelberg, 2005.

[38] Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS'16)*, pages 818–829. ACM, 2016.

[39] Vladimir Kolesnikov, Naor Matania, Benny Pinkas, Mike Rosulek, and Ni Trieu. Practical multi-party private set intersection from symmetric-key techniques. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1257–1272. Association for Computing Machinery, 2017.

[40] Jack P. K. Ma and Sherman S. M. Chow. Friendly private set intersection from oblivious compact graph evaluation. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pages 1086–1097. ACM, 2022.

[41] Peihan Miao, Sarvar Patel, Mariana Raykova, Karn Seth, and Moti Yung. Two-sided malicious security for private intersection-sum with cardinality. In *Advances in Cryptology – CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020*, pages 3–33. Springer International Publishing, 2020.

[42] Y. Minsky, A. Trachtenberg, and R. Zippel. Set reconciliation with nearly optimal communication complexity. *IEEE Transactions on Information Theory*, 49(9):2213–2218, 2003.

[43] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudorandom functions. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 458–467, 1997.

[44] Ofri Nevo, Ni Trieu, and Avishay Yanai. Simple, fast malicious multiparty private set intersection. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1151–1165. ACM, 2021.

[45] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology — EUROCRYPT '99*, pages 223–238. Springer Berlin Heidelberg, 1999.

[46] V. Ya. Pan. Strassen's algorithm is not optimal: Trilinear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations. In *19th Annual Symposium on Foundations of Computer Science (SFCS 1978)*, pages 166–176, 1978.

[47] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. SpOT-light: lightweight private set intersection from sparse OT extension. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III 39*, pages 401–431. Springer International Publishing, 2019.

[48] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from PaXoS: fast, malicious private set intersection. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 739–767. Springer International Publishing, 2020.

[49] Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. Phasing: Private set intersection using permutation-based hashing. In *24th USENIX Security Symposium (USENIX Security '15)*, pages 515–530, 2015.

[50] Benny Pinkas, Thomas Schneider, Oleksandr Tkachenko, and Avishay Yanai. Efficient circuit-based PSI with linear communication. In *Advances in Cryptology – EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 122–153. Springer International Publishing, 2019.

[51] Benny Pinkas, Thomas Schneider, Christian Weinert, and Udi Wieder. Efficient circuit-based PSI via cuckoo hashing. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 125–157. Springer International Publishing, 2018.

[52] Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on OT extension. In *23rd USENIX Security Symposium (USENIX Security '14)*, pages 797–812, 2014.

[53] Benny Pinkas, Thomas Schneider, and Michael Zohner. Scalable private set intersection based on OT extension. *ACM Transactions on Privacy and Security (TOPS)*, 21(2):1–35, 2018.

[54] Srinivasan Raghuraman and Peter Rindal. Blazing fast PSI from improved OKVS and subfield VOLE. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2505–2517. ACM, 2022.

[55] Peter Rindal and Phillipp Schoppmann. VOLE-PSI: Fast OPRF and circuit-PSI from vector-OLE. In *Advances in Cryptology – EUROCRYPT 2021*, pages 901–930. Springer International Publishing, 2021.

[56] Victor Shoup. NTL: A library for doing number theory. https://libntl.org/.

[57] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.

[58] Joris van der Hoeven and Michael Monagan. Implementing the tangent Graeffe root finding method. *Mathematical Software — ICMS 2020*, pages 482–492, 2020.

[59] Joris van der Hoeven and Michael Monagan. Computing one billion roots using the tangent Graeffe method. *ACM Commun. Comput. Algebra*, 54(3):65–85, 2021.

[60] Jelle Vos, Mauro Conti, and Zekeriya Erkin. Fast multi-party private set operations in the star topology from secure ANDs and oRs. *IACR Cryptology ePrint Archive, Paper 2022/721*, 2022. https://eprint.iacr.org/2022/721.

[61] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. *arXiv preprint arXiv:2307.07970*, 2023. https://arxiv.org/abs/2307.07970.

[62] Yaxi Yang, Jian Weng, Yufeng Yi, Changyu Dong, Leo Yu Zhang, and Jianying Zhou. Predicate private set intersection with linear complexity. In *International Conference on Applied Cryptography and Network Security*, pages 143–166. Springer Nature Switzerland, 2023.

[63] Foo Yee Yeo and Jason H. M. Ying. Third-party private set intersection. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 1633–1638. IEEE, 2023.

[64] Foo Yee Yeo and Jason H. M. Ying. A round-optimal near-linear third-party private set intersection protocol. *IACR Cryptology ePrint Archive, Paper 2024/566*, 2024. https://eprint.iacr.org/2024/566.

[65] Yongjun Zhao and Sherman S. M. Chow. Are you the one to share? Secret transfer with access structure. *Proceedings on Privacy Enhancing Technologies*, 2017(1):149–169, 2017.

[66] Yongjun Zhao and Sherman S. M. Chow. Can you find the one for me? In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*, pages 54–65, 2018.