Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 2, No. 3, 2022

Multi-party Privacy Set Intersection protocol with an Untrusted Cloud Server

He Tian, Jing Zhang, Li Yang, Yongli Tang*, and Chunming Zha

School of Software, Henan Polytechnic University, Jiaozuo 454000, China

* **Corresponding author**: Yongli Tang (Email: yltang@hpu.edu.cn)

Abstract: The development of cloud computing has brought many conveniences to life, its advantages of low cost and high computing power bring convenience to our life, but the main problem is how to protect user data on untrustworthy cloud servers. To solve this problem, we have proposed a new private set intersection protocol, denoted as. The protocol uses a representation of sets with polynomial point-value pairs. All elements should be transmitted and computed in the protocol as ciphertext by introducing random polynomials to re-randomize the encryption elements. To reduce the computational burden of each party. The parties send the processed polynomials to the cloud server, which calculates the intersection. In addition, our protocol achieves fairness in the case of collusion between parties. Finally, the security of the protocol is demonstrated under the UC model. Comparing the performance of this protocol with other related protocols show that the protocol is better than the other associated protocols in terms of low computational overhead.

Keywords: Private set intersection; Secure multi-party computation; Fair computation; Oblivious Linear Function Evaluation.

1. Introduction

1.1. Background

Private Set Intersection (PSI) is an important branch of secure multi-party computation that allows parties to jointly compute the common information of their sets without revealing any additional privacy information. At the end of the PSI protocol, if the PSI protocol outputs the results to all parties, we say it's fair. Since Freedman proposed PSI in [1], PSI has not only become an important research problem in privacy, but also has specific applications in many scenarios. For example, in the process of genetic similarity testing, the participants only want to get the same gene fragment of both for themselves, while the rest of the gene fragments should be kept strictly confidential. Another typical example is private contact discovery, where WeChat makes friend account recommendations based on the cell phone address book. At the same time, the emergence of cloud server provides more convenient conditions for PSI[2,3]. The PSI protocol combined with cloud computing technology is also widely utilized, such as online advertising[4], genetic testing[5], contact tracing for infection detection[6,7].

PSI based on cloud outsourcing brings us a lot of convenience, but there are still some problems that we need to solve. protocols based on public-key cryptosystems often require additional management of keys which is too expensive. In addition, the cloud server it may not perform the computation honestly, so one of the basic requirements of outsourcing the computation is to ensure the correctness of the results. In other words, the user should have a way to verify the correctness of the cloud server output with absolute probability. Although the cloud server is not trusted, inappropriate behavior can occur on the client side. For instance, a malicious user may intentionally claim that the cloud server's outputs are incorrect, even though the cloud server performed the calculations honestly. Therefore, the results are best disclosed to all parties at the same time. These problems are also of concern to us.

1.2. Our Contributions

A new protocol $\prod_{MPSI}$ is proposed which is implemented by combining cloud servers and Oblivious Linear Function Evaluation (OLE). The protocol achieves fairness in multi-party PSI while protecting the privacy of all parties. The main contributions of this paper can be summarized in the following three aspects:

(a)The $\prod_{MPSI}$ protocol uses polynomials to improve the pre-processing of set elements by introducing common sets. Using OLE to blind the polynomials of the parties involved, only the polynomials need to be compared. The computation overhead of the entire protocol is reduced by reducing the number of times to determine whether the elements are equal.

(b)The $\prod_{MPSI}$ protocol further reduces the computational burden on all parties by shifting the computation of intersections to the server. In addition, multiple parties are able to run the $\prod_{MPSI}$ protocol in parallel with each other. In addition, the protocol ends with each party knowing the result of the intersection, achieving fairness of the $\prod_{MPSI}$ protocol.

(c)The $\prod_{MPSI}$ protocol resists collusion between the cloud server and $n-1$ parties. The $\prod_{MPSI}$ protocol also ensures that each party can check the integrity and correctness of the results to counteract malicious behavior of the cloud server.

2. Related work

Scholars have studied PSI extensively, and in this section, we present further related work. Generally speaking, these works could be categorized into the 3 types.

2.1. Public-key technology

The PSI protocol was proposed in [1] based on polynomial evaluation, where a polynomial is constructed from the elements in the set, the coefficients of the polynomial are found by interpolation, and the polynomial coefficients are

sent to the server in ciphertext. In this protocol, a larger number of polynomials will cause the higher computational costs in the homomorphic encryption process. In [8], authors constructed the oblivious pseudo-random function (OPRF) based on the inadvertent transmission expansion protocol to efficiently complete the privacy equivalence test and propose a PSI protocol. In [9] , authors introduced the cuckoo hash mapping technique based on [8] and proposes an efficient PSI protocol. In [10] a new PSI protocol with lower complexity is given. Based on this, [11] proposed an unbalanced PSI protocol is made resistant to malicious attacks by using OPRF in the initialization part of this protocol. In [12] , authors used a new inadvertent transfer expansion protocol based on the [9] for constructing inadvertent pseudo-random functions as well as PSI schemes. The most common technique is homomorphic encryption. The authors of [13] improved computational efficiency by using ElGamal encryption and reduced computational complexity by using the Cuckoo Hash technique [14]. Abadi proposed a point-value pair-based d-polynomial representation of the set method, which is accomplished by the Paillier encryption scheme, reducing the multiplicative complexity from $O(d^2)$ to $O(d)$ [15]. The PSI protocols based on public-key technology are generally suitable for models with high computational power, but communication bandwidth and time complexity are a big obstacle for practical applications.

2.2. Oblivious transfer technology

Traditional OT protocols limit the security and efficiency of PSI protocols due to the large number of OTs that need to be used, but this bottleneck can be effectively solved by OT extension technology (OTE). Phase hashing and permutation hashing methods were applied to reduce computational effort and memory in [16]. A protocol based on OT extensions that gives a lightweight protocol for security and communication networks was proposed in[17]. In [18], Pinkas analyzed the existing PSI protocols and constructed a new PSI protocol using hashing techniques and OPRF to optimize the PSI protocol. In [19], a PSI protocol based on string detection with Paxos (Paxos) data structure was proposed that not only has linear communi-cation and computational complexity, but also resistant to malicious attacks Dong cons-tructed a semi-honest PSI protocol based on IKNP03-OT [20] and Bloom filters [21]. Later, Pinkas introduced ALSZ13-OT [22] based on [21], which led to a 50% reduction in the amount of communication between the parties [8]. To make the protocols more computationally efficient, most OT protocols are implemented using symmetric cryptosystems. These protocols have low communication complexity and time complexity. However, these protocols require operations such as key management.

2.3. Cloud computing.

Cloud computing technology provides users with flexible and economical computing resources. Using cloud servers to assist in computing privacy set intersection is a feasible approach, however, cloud servers bring convenience to users but also raise new security issues, and untrustworthy cloud servers may violate data privacy confidentiality agreements to steal private data [23,24]. Therefore, a number of cloud server-based privacy intersection solutions have emerged. In [25], authours constructed a series of PSI protocols for cloud computing environments that allow two-party clients to use the computational power and storage space of the cloud server to assist in computing PSI of both parties without leaking private information to the cloud server, but are limited to solving the two-party privacy intersection computation problem. In [26,27], authours proposed privacy set intersection protocols supporting multiple parties respectively, but their computation and communication are done by each party, and when the party's equipment is limited, it is not possible to securely use cloud services to assist the computation. In [28], A fair two-party PSI protocol is proposed, which combines cloud servers with cuckoo hashing to achieve a fair PSI.

The variation between the above protocols are shown in Table 1, and we have mainly analyzed structure and adversary model.

Table 1. Variation between the above protocols.

Type	Reference	Structure	Adversary model
Public Key	[8]	OPRF+GBF	Semihonest
	[9]	OPRF+Hash	Semihonest
	[10]	HE	Semihonest
	[11]	OPRF+HE	Malicious
	[12]	OPRF+ IKNP-OT	Semihonest
	[13]	HE+Hash	Malicious
	[15]	HE	Semihonest
Oblivious transfer	[16]	GC+OPRF	Semihonest
	[17]	OPRF+Hash	Semihonest
	[18]	OOS-OT+Hash	Malicious
	[19]	IKNP-OT	Semihonest
	[20]	BF+ IKNP-OT	Malicious
Cloud computing	[24]	Hash+ point-value polynomial	Semihonest
	[25]	OPPRF+OT	Semihonest
	[26]	KSO15-OT	Malicious
	[28]	ElGamal +Hash	Semihonest

3. Preliminaries

In this section, we give the representation of polynomials with sets and the related definitions. We also provide the OLE algorithm and security model used to construct our scheme.

3.1. Polynomial point-value pairs represent sets

In order to achieve the conversion from set to polynomial. The representation illustrates that a random point evaluation polynomial can be used to represent the set.

Definition 1. Polynomial point-value pairs represent sets.

Given a set $X=\{x_1,\cdots,x_d\}$,we define its characteristic polynomial as

$$f(x)=(x-x_1)\times\cdots\times(x-x_d) \tag{1}$$

to determine whether a set contains an element x , it is only necessary to determine whether $f(x)=0$ holds.

Definition 2. Suppose X and Y are two sets of the same degree d and denoted by polynomials $F(x),G(x)$ .We can represent the intersection $F(x)\bigcap G(x)$ by finding the roots of this polynomial when it is equal to 0,

$$F(x)H_1(x)+G(x)H_2(x) \tag{2}$$

where $H_1(x)$ and $H_2(x)$ are randomly chosen d-degree polynomials.

Definition 3. Let $\Omega_0(R)$ denote the $\forall x\in\Omega_0(R),R(x)=0$. Let $d,d'\in poly(\log|\mathbb{F}|)$. Suppose $r\in\mathbb{F}[x],\deg(r)=d$ is a

random polynomial. For all non-trivial $p \in \mathbb{F}[x]$, $\deg(p) = d'$,

$$\Pr[(\Omega_0(r) \cap \Omega_0(p)) \neq \varnothing] \leq negl(|\mathbb{F}|) \quad (3)$$

3.2. Oblivious Linear Function Evaluation.

Intuitively, Oblivious Linear Function Evaluation (OLE) is a generalisation of OT to the larger domain by [29]. The ideal functionality of F_{OLE} allows the sender and receiver to jointly compute a linear function $c(x) = ax + b$. In the F_{OLE}, the receiver inputs x and obtains $c(x)$ and the sender inputs a,b and obtains nothing. The OLE guarantees that the receiver has no knowledge of a,b other than the result $c(x)$, and the sender has no knowledge of x. The ideal functionality of F_{OLE} is shown in Figure 1.

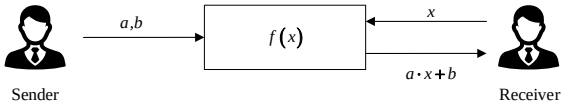

Figure 1. The ideal functionality of F_{OLE}

3.3. Security Model.

In the Universal Composability (UC) framework of [30] the $\prod_{MPSI}$ protocol is proved. Let π corresponds to the protocol in the real world and F corresponds to the ideal function in the ideal world. For π protocols in the real world, there is a PPT adversary $\mathcal{A}$ and will exist a simulator Sim in the ideal world. The protocol is secure if Equation 4 holds.

$$Real_{\pi,\mathcal{A}}(X,Y) \approx Ideal_{F,Sim}(X,Y) \quad (4)$$

where $Real_{\pi,\mathcal{A}}(X,Y)$ is the real output, $Ideal_{F,Sim}(X,Y)$ is the ideal output, the $\approx$ indicates computationally indistinguishable.

4. Our Protocol

In this section, we present the proposed PSI protocol. To make it easier to understand, we first introduce the overall idea of designing a protocol.

4.1. Overview

The protocol has n parties $P_i (1 \leq i \leq n)$ and a cloud server S are involved. P_i input his private set X_i and compute the intersection $\bigcap_{i=1}^{n} X_i$ through S without leakage information. We have proposed a new design idea based on [29], where F_{OLE} algorithm is run between P_i and $P_c (1 \leq i \leq n, c = (i+1) \bmod n)$. P_i obtains polynomial R_i, after which the party P_i sends the polynomial R_i to S. The cloud server S computes $R_{com} = \sum_{i=1}^{n} R_i$ and the set formed by the roots of the R_{com} is the intersection of the sets X_i.

However, we have discrepancy with the protocol of [29]. We have analyzed three possible scenarios of untrustworthy cloud servers.

(a) It can claim that all elements are not in the intersection set.

(b) It can claim that all elements are in the intersection set.

(c) The cloud server feeds only a portion of the intersection to the party.

We need to ensure that the results returned by the cloud server S cannot be empty and cannot directly return all the information entered by the parties [31]. In order to be able to realize this, the parties work collectively to generate a set $D_0, (|D_0| = (d - d_i)/2)$ and P_i generates the private set $D_i, (|D_i| = (d - d_i)/2)$. First, P_i chooses $D_0 + D_i$ to blind the set X_i, then P_i get $X_i' = X_i + D_0 + D_i$. Second, P_i expands its set X_i' with λ copies of all element. Prior to the start of the protocol, P_i need creates his new set $X_i^{\lambda} = \{(x_i^j \| 1, \cdots, x_i^j \| \lambda)\}$ for all $x_i^j \in X_i'$, meanwhile, define $(X_i^{\lambda})^{-\lambda} = X_i'$.

However, there is still a need to consider data privacy during the interaction between the two sets of information while correctly computing the intersection. We use OLE to solve this problem. In our protocol, the parties P_i holds sets $X_i^{\lambda}, (|X_i^{\lambda}| = \lambda d)$ and constructs a polynomial $R_i^0(x)$ based on X_i^{λ}. P_i randomly generates two polynomials $R_i^1(x)$ and $R_i^2(x)$, where $|R_i^1(x)| = |R_i^2(x)| = \lambda d / 2$. P_i and $P_c (1 \leq i \leq n, c = (i+1) \bmod n)$ run the F_{OLE} algorithm twice, P_i get $R_i'(x) = R_c^1(x) \bullet R_i^0(x) + R_c^2(x)$ and P_c get $R_c'(x) = R_i^1(x) \bullet R_c(x) + R_i^2(x)$. If $R_i'(x)$ is sent directly to the cloud server S, the polynomial $R_i^0(x)$ of P_i is compromised if P_c and the cloud server S are colluded. To avoid this problem, we require the $P_i (1 \leq i \leq n)$ to do another blinded operation before sending.

$$\begin{aligned} R_i^{F_{OLE}} &= R_i'(x) - R_i^2(x) + R_i^0(x) \bullet R_i^2(x) \\ &= R_i^0(x) \bullet (R_c^1(x) + R_i^2(x)) + R_c^2(x) - R_i^2(x) \end{aligned} \quad (5)$$

P_i sends $R_i^{F_{OLE}}$ to the cloud server S. The cloud server S derives $R_{com} = \sum_{i=1}^{n} R_i^{F_{OLE}}$. After that, the cloud server send R_{com} to the P_i.

4.2. $\prod_{MPSI}$ Protocol

We give a detailed description of our $\prod_{MPSI}$ proposed protocol in this section. we define the set $X_i \subseteq \mathcal{U}$ as $X_i = \{x_i^1, \cdots x_i^{j_i}, \cdots, x_i^{d_i}\}, j_i \in [1, d_i]$. The new protocol $\prod_{MPSI}$ is shown in Algorithm 1.

Inputs: P_i holds the set X_i, where $i \in [1, n], |X_i| = d_i$. Let $d = \max\{|X_i|\} + 1$.

Outputs: $X_{Set-\text{int}} = \bigcap_{i=1}^{d} X_i$.

1. Preparation Phase.

(1) All parties run a pseudo-random function generator F_{PRF} to generate the set D_0, where $|D_0| = (d - d_i)/2$.

(2) P_i run F_{PRF} to generate three private sets D_i^1, D_i^2 and D_i^3, where $|D_i^1| = (d - d_i)/2$, $|D_i^2| = \lambda d / 2$ and $|D_i^3| = \lambda d / 2$. P_i uses $D_0 + D_i^1$ to blind the set X_i, then

$X_i' = X_i + D_0 + D_i^1$. For all $x_i^j \in X_i'$, P_i creates his new set $X_i^\lambda = \{(x_i^j \| 1, \cdots, x_i^j \| \lambda)\}$

(3) P_i generates three polynomials $R_i^0(x)$, $R_i^1(x)$ and $R_i^2(x)$, where $R_i^0(x) = \prod_k^{\lambda d}(x - x_k), x_k \in X_i^\lambda$, $R_i^1(x) = \prod_k^{\lambda d/2}(x - x_k), x_k \in D_i^2$ and $R_i^2(x) = \prod_k^{\lambda d/2}(x - x_k), x_k \in D_i^3$.

2. Execute $\mathcal{F}_{OLE}$ and Blindness.

(1) P_i as a receiver, inputs $R_i^0(x)$ into F_{OLE}. $P_c (1 \le i \le n, c = (i+1) \bmod n)$ as a sender, inputs $R_c^1(x)$ and $R_c^2(x)$ into F_{OLE}. P_i get $R_i' = R_i^0(x) \bullet R_c^1(x) + R_c^2(x)$, P_c get $\perp$.

(2) P_c as a receiver, inputs $R_c^0(x)$ into F_{OLE}. P_i as a sender, inputs $R_i^1(x)$ and $R_i^2(x)$ into F_{OLE}. P_c get $R_c' = R_c^0(x) \bullet R_i^1(x) + R_i^2(x)$, P_i get $\perp$.

(3) $P_i (1 \le i \le n)$ computes $R_i^{F_{OLE}} = R_i^0(x) \bullet \left(R_c^1(x) + R_i^2(x)\right) + R_c^2(x) - R_i^2(x)$.

3. Set intersection.

The cloud server S computes $R_{com} = \sum_{i=1}^n R_i^{F_{OLE}}$ and sends R_{com} to P_i.

4. Checking and verification.

(1) P_i aborts the protocol if any of the following occurs: $R_{com}(x) \ne 0, x \in D_0$ or $R_{com}(x) = 0, x \in D_i$ or $R_{com}\left(x_i^j \| \alpha\right) = 0, R_{com}\left(x_i^j \| \beta\right) \ne 0, x_i^j \in X_i', \alpha, \beta \in [1, \lambda]$.

(2)For $x \in X_i^\lambda$, if $R_{com}(x) = 0, x \in \bigcap_{i=1}^d X_i^\lambda$. Then P_i calculates $\bigcap_{i=1}^d X_i' = \left(\bigcap_{i=1}^d X_i^\lambda\right)^{-\lambda}$ and $X_{Set-\text{int}} = \bigcap_{i=1}^d X_i' - D_0$.

Algorithm 1. Malicious security Multi-party PSI

4.3. Analysis of correctness

Theorem 1. We say that $\prod_{MPSI}$ protocol does not correctly calculate the probability of intersection as :

$$\Pr\left[R_{com}(x) = 0, x \notin \bigcap_{i=1}^n X_i\right] \le negl(|\mathbb{F}|) \tag{6}$$

Proof. According to the $\prod_{MPSI}$, we get $R_i^{F_{OLE}} = R_i'(x) - R_i^2(x) = R_c^1(x) \bullet R_i^0(x) + R_c^2(x) - R_i^2(x)$, then

$$\begin{aligned} R_{com} &= \sum_{i=1}^n R_i^{F_{OLE}} \\ &= \sum_{i=1}^n \left(R_i'(x) - R_i^2(x)\right) \\ &= \left(R_1^0(x) \bullet \left(R_n^1(x) + R_1^1(x)\right)\right) \\ &\quad + \cdots + \left(R_n^0(x) \bullet \left(R_n^1(x) + R_1^1(x)\right)\right) \end{aligned} \tag{7}$$

According to the Definition 2, equation (7) is equivalent to

$$R_{com} = X_1^\lambda \cap \cdots \cap X_n^\lambda \tag{8}$$

then P_i calculates $\bigcap_{i=1}^d X_i' = \left(\bigcap_{i=1}^d X_i^\lambda\right)^{-\lambda}$, we can get

$$\begin{aligned} R_{com} &= X_1' \cap \cdots \cap X_n' \\ &= \left(X_1 + D_0 + D_1^1\right) \cap \cdots \cap \left(X_n + D_0 + D_n^1\right) \\ &= \left(X_1 + D_1^1\right) \cap \cdots \cap \left(X_n + D_n^1\right) \cup D_0 \end{aligned} \tag{9}$$

Let $R_{X_i}(x) = \prod_k^{d_i}(x - x_k), x_k \in X_i, (1 \le i \le n)$, $\Omega_0(\Psi_1) = \Omega_0\left(R_{X_i} + R_c^1\right)$ and $\Omega_0(\Psi_2) = \Omega_0\left(R_{X_c} + R_i^1\right)$ $(c = (i+1) \bmod n)$. equation (9) is equivalent to

$$\begin{aligned} \Omega_0(\Psi_1) \cap \Omega_0(\Psi_2) &= \Omega_0\left(R_{X_i} + R_c^1\right) \cap \Omega_0\left(R_{X_c} + R_i^1\right) \\ &= \Omega_0\left(R_{X_i} \cup R_c^1\right) \cap \Omega_0\left(R_{X_c} \cup R_i^1\right) \\ &= \Omega_0\left(R_{X_i} \cap R_{X_c}\right) \cup \Omega_0\left(R_{X_i} \cap R_i^1\right) \\ &\quad \cup \Omega_0\left(R_c^1 \cap R_{X_c}\right) \cup \Omega_0\left(R_c^1 \cap R_i^1\right) \end{aligned} \tag{10}$$

By Definition 3, we have

$$\begin{aligned} \Pr\left[\Omega_0\left(R_{X_i} \cap R_i^1\right) \ne \varnothing\right] &\le negl(|\mathbb{F}|) \\ \Pr\left[\Omega_0\left(R_c^1 \cap R_{X_c}\right) \ne \varnothing\right] &\le negl(|\mathbb{F}|) \\ \Pr\left[\Omega_0\left(R_c^1 \cap R_i^1\right) \ne \varnothing\right] &\le negl(|\mathbb{F}|) \end{aligned} \tag{11}$$

Then we can get

$$\begin{aligned} \Omega_0(\Psi_1) \cap \Omega_0(\Psi_2) &= \left(X_1' - D_0\right) \cap \cdots \cap \left(X_n' - D_0\right) \\ &= X_1 \cap \cdots \cap X_n \\ &= X_{Set-\text{int}} \end{aligned} \tag{12}$$

According to the above equations, we can conclude that $\Pr\left[R_{com}(x) = 0, x \notin \bigcap_{i=1}^n X_i\right] \le negl(|\mathbb{F}|)$, the proof is finished.

4.4. Security Analysis

The formal security proof of polynomial-based protocol is given.

Theorem 2. The $\prod_{MPSI}$ protocol is secure in the presence of a malicious adversary $\mathcal{A}$ if the F_{PRF} is the pseudo-random function.

Proof. This theorem is proved by taking into account the cases in which the parties are destroyed. By building a simulator in each case, only the inputs and outputs of the destroyed party are obtained and a view is generated. Further, the view must be computationally indistinguishable from the real view. Let $M = \{i | P_i \text{ is malicious}\}$ and $|M| = M_t \le n-1$. Let $H = \{j | P_j \text{ is honest}\}$ and $|H| = H_t \ge 1$ .We will only discuss the worst-case scenario here. Namely, the $|M| = M_t = n-1$.

Case 1: Cloud sever S and P_1 are honest, the other parties are malicious. In this case, we construct the simulator $Sim1$ as follows. It is given M input set X_M. $Sim1$ generates its own view based on the execution process of the protocol. For the OT, $Sim1$ generates P_1's random string $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$, In order to build the view, $Sim1$ runs the F_{OLE} as receiver and inputs $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ and outputs $\tilde{R}_1^{F_{OLE}}$. Then $Sim1$ sends $\tilde{R}_1^{F_{OLE}}$ to S. Finally $Sim1$ outputs M's view.

Consider the following hybrid games.

Game 0: M and P_1 output in the real protocol. This corresponds to $Real_{\Pi_{PSI},\mathcal{A}}(X_1, X_M)$. From the $\prod_{MPSI}$ protocol, we know $Real_{\Pi_{PSI},\mathcal{A}}(X_1, X_M) = \left(R_1^{F_{OLE}}, \cdots, R_n^{F_{OLE}}, X_{Set-\text{int}}\right)$.

Game 1: Identical to Game 0, except that on P_1 chooses

the random F_{PRF} key. The $Sim1$ aborts according to the Checking and verification in Algorithm 1.

If the environment can distinguish between Game 0 and Game 1, then the environment must manage to send $\tilde{R}_1^{F_{OLE}}$ such that $\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)\neq\left(R_2^2(x)-\tilde{R}_1^2(x)\right)$, while passing the check of Checking and verification in Algorithm 1 with non-negligible probability. Let $\tilde{R}_1^{F_{OLE}}=\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)+R_2^2(x)-\tilde{R}_1^2(x)$. Due to the fact that x is uniformly random, the probability of satisfying $\tilde{R}_1^{F_{OLE}}=0$ is $d/|\mathbb{F}|$. Therefore, Game 1 and Game 0 are computationally indistinguishable.

Game 2: The remainder is the same as Game 1 except for $Sim1$ generates the $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ according to the Algorithm 1 and adjustsd the output. This corresponds to $Ideal_{F,Sim1}\left(\tilde{X}_1,X_M\right)$.

The previous Games determined that the inputs $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are extracted correctly. Note that the received intersection $\tilde{R}_{com}=\tilde{R}_1^{F_{OLE}}+R_2^{F_{OLE}}+\cdots+R_n^{F_{OLE}}$, the adversary $\mathcal{A}$ cannot modify the element information in the intersection unless there is a negligible probability.

Further, As $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are uniform randomly over the choice, $\tilde{R}_1^{F_{OLE}}=\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)+R_2^2(x)-\tilde{R}_1^2(x)$ is also randomly distributed. Similarly, the $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are uniform randomly over the choice, $\tilde{R}_{com}$ is also randomly distributed. Due to the degree of $\tilde{R}_1^0(x)$ is d, the $R_2^1(x)$ and $R_2^2(x)$ are randomly. Therefore, the Game 2 and Game 1 are computationally indistinguishable.

. Then we get $Ideal_{F,Sim1}\left(\tilde{X}_1,X_M\right)=\left(\tilde{R}_1^{F_{OLE}},\cdots,R_n^{F_{OLE}},\tilde{X}_{Set-\text{int}}\right)$. Combining the above, we can get that

$$Real_{\Pi_{PSI},\mathcal{A}}(X_1,X_M)\approx Ideal_{F,Sim1}\left(\tilde{X}_1,X_M\right) \quad (13)$$

Case 2: We assume that only P_1 is honest, the cloud server S and the other parties are malicious. Also, the malicious cloud server S can conspire with $n-1$ malicious parties. We construct the simulator $Sim2$ as follows. It is given M input set X_M. $Sim2$ generates its own view based on the execution process of the protocol. For the OT, $Sim2$ generates P_1's random string $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$, In order to build the view, $Sim2$ runs the F_{OLE} as receiver and inputs inputs $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ and outputs $\tilde{R}_1^{F_{OLE}}$. Then $Sim2$ sends $\tilde{R}_1^{F_{OLE}}$ to S. Finally $Sim1$ outputs M's view.

Consider the following series of hybrid games.

Game 0: $Real_{\Pi_{PSI},\mathcal{A}}(X_1,X_M)$. M and P_1 output in the real protocol. This corresponds to $Real_{\Pi_{PSI},\mathcal{A}}(X_1,X_M)$. From the $\prod_{MPSI}$ protocol, we know $Real_{\Pi_{PSI},\mathcal{A}}(X_1,X_M)=\left(R_1^{F_{OLE}},\cdots,R_n^{F_{OLE}},X_{Set-\text{int}}\right)$.

Game 1: Identical to Game 0, except that on P_1 chooses the random F_{PRF} key. The $Sim1$ aborts according to the Checking and verification in Algorithm 1.

If the environment can distinguish between Game 0 and Game 1,then the environment must manage to send $\tilde{R}_1^{F_{OLE}}$ such that $\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)\neq\left(R_2^2(x)-\tilde{R}_1^2(x)\right)$, while passing the check in Checking and verification in Algorithm 1 with non-negligible probability. Let $\tilde{R}_1^{F_{OLE}}=\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)+R_2^2(x)-\tilde{R}_1^2(x)$. Due to the fact that x is uniformly random, the probability of satisfying $\tilde{R}_1^{F_{OLE}}=0$ is $d/|\mathbb{F}|$. Therefore, Game 1 and Game 0 are computationally indistinguishable.

Game 2: The remainder is the same as Game 1, except that $Sim1$ generates the $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ according to the Algorithm 1 and adjustsd the output. This corresponds to $Ideal_{F,Sim1}\left(\tilde{X}_1,X_M\right)$.

The previous Games determined that the inputs $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are extracted correctly. Note that the received intersection $\tilde{R}_{com}=\tilde{R}_1^{F_{OLE}}+R_2^{F_{OLE}}+\cdots+R_n^{F_{OLE}}$, the adversary $\mathcal{A}$ cannot modify the element information in the intersection unless there is a negligible probability.

Further, As $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are uniform randomly over the choice, $\tilde{R}_1^{F_{OLE}}=\tilde{R}_1^0(x)\bullet\left(R_2^1(x)+\tilde{R}_1^2(x)\right)+R_2^2(x)-\tilde{R}_1^2(x)$ is also randomly distributed. Similarly, the $\tilde{R}_1^0(x)$, $\tilde{R}_1^1(x)$ and $\tilde{R}_1^2(x)$ are uniform randomly over the choice, $\tilde{R}_{com}$ is also randomly distributed. Therefore, the Game 2 and Game 1 are computationally indistinguishable.

Finally, the malicious cloud server S conspire with $n-1$ malicious parties. Then the adversary $\mathcal{A}$ obtains $R_1^{F_{OLE}}=R_1^0(x)\bullet\left(R_2^1(x)+R_1^2(x)\right)+R_2^2(x)-R_1^2(x)$. To the adversary $\mathcal{A}$, $R_1^1(x)$ and $R_1^2(x)$ are random and unknown, the adversary $\mathcal{A}$ cannot get the information about $R_1^0(x)$ and pass the check with absolute advantage. Then we get $Ideal_{F,Sim1}\left(\tilde{X}_1,X_M\right)=\left(\tilde{R}_1^{F_{OLE}},\cdots,R_n^{F_{OLE}},\tilde{X}_{Set-\text{int}}\right)$. Combining the above, we can get that

$$Real_{\Pi_{PSI},\mathcal{A}}(X_1,X_M)\approx Ideal_{F,Sim2}\left(\tilde{X}_1,X_M\right) \quad (14)$$

the proof is finished.

5. Efficiency Analysis

5.1. Theoretical Evaluation

The comparison between the $\prod_{MPSI}$ protocol and [29,32] is given in this section. In contrast to previous work that relied on exponential operations on domains, our protocol requires only addition and multiplication for computing privacy set intersections. Our $\prod_{MPSI}$ protocol requires one interaction between each party and also one interaction with the cloud server. The resulting communication complexity is $O\left(\left(n^2+n\log d\right)k\right)$. P_i and $P_c\,(1\le i\le n, c=(i+1)\bmod n)$ perform nd operations in the OLE algorithm and $n\log d$ operations in the encryption step. The cloud server computes PSI and no other operations are involved.

The detailed analysis of the relevant PSI protocols is presented in table 2.

We evaluated its performance in five aspects: total communication complexity, total computation complexity, client communication, client Computation and fairness. From

the Table 2, compared to literature [29] and [32], our $\prod_{MPSI}$ protocol has the following advantages:

(a) The total computational complexity, total communication complexity, client communication and client computation of our protocol are no greater than other protocols. The $\prod_{MPSI}$ protocol uses polynomials to perform comparisons. The computational efficiency of the $\prod_{MPSI}$ protocol is reduced by reducing the times of the set are compared. At the same time all parties can execute the protocol in parallel. This is why the protocol has a significantly lower computational and communication cost than other protocols.

(b) Only our protocol achieves fair multi-party PSI. The $\prod_{MPSI}$ protocol guaranteed that after the protocol is executed, each party can know the intersection result.

Table 2. Comparison with related PSI protocols

Protocal	Communication	Computation	Fairness
	$O(k(n^2+nd))$	$O(nd\log d)$	No
	$O(k(n^2+nd\log d))$	$O(n+nd\log d)$	No
Ours	$O(k(n^2+n\log d))$	$O(nd+n\log d)$	Yes

5.2. Experimental Evaluation

To more visually compare the runtime overhead of each protocol, simulation experiments and results analysis are conducted in this paper. It should be noted that the time consumed by the protocol refers to the average time consumed in 10 experiments. The experimental platform is Windows 10, Intel(R) Core(TM) i5-5200U CPU @ 2.20 GHz 2.20 GHz, 4.00 GB of RAM, and a compiled environment of Dev-C++5.11.

First consider the effect of the change in the number of elements contained in the set on the running time. Suppose $n=20$, $k=128$, $d=2^{10},2^{12},2^{14},2^{16},2^{18},2^{20}$ are selected for comparison experiments, and the relationship between the protocol running time and the number of elements contained in the set is shown in Figure 2.

From Figure 2, when the number of sets increases, the execution time of all three protocols grows. However, the time of $\prod_{MPSI}$ protocol increases the slowest when the fixed set cardinality is small. In addition, the $\prod_{MPSI}$ protocol has the slowest time growth rate.

In addition, the change in the number of partes in the protocol has an impact on the running time of the protocol. Assume that the total number of elements in the set $d=\sum_{i=1}^{n} d_i = 2^8$ and keep the remaining parameters constant. The number of parties $n=10^3,10^4,10^5,10^6$ is selected for comparison experiments, and the relations between the protocol running time and the number of parties is shown in Figure 3.

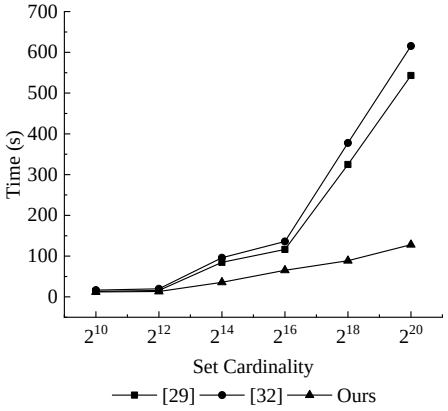

Figure 2. Running time v.s. set cardinality

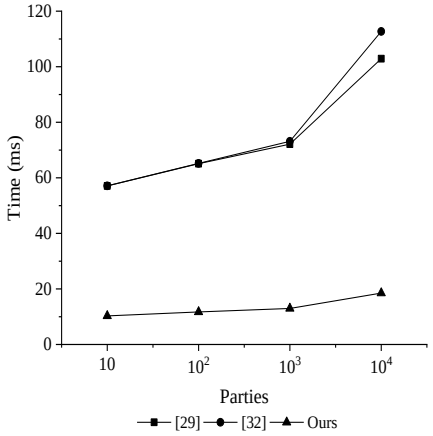

Figure 3. Running time v.s. the number of parties

From Figure 3, the running time of all protocols increases gradually with the number of parties. The time overheads of the $\prod_{MPSI}$ protocols are lower than those of the other protocols when n is fixed. In addition, the $\prod_{MPSI}$ protocol has the slowest time growth rate.

To sum up, the $\prod_{MPSI}$ protocol saves the time overhead of the multi-party PSI computation to a certain extent, improves the calculation efficiency, and realizes the fair multi-party PSI computation. Therefore, the overall efficiency of the $\prod_{MPSI}$ protocol is better than the compared protocols.

6. Conclusion

PSI is a popular field of study and has been applied in many domains. In this paper, we design a multi-party privacy intersection computation protocol for untrustworthy cloud computing environment, so that the multi-party holding the privacy set can use the computing and communication capabilities of untrustworthy cloud servers to assist the parties to complete the privacy set intersection computation without revealing any set information. This paper has a wider range of applications than existing schemes. The next work will focus on how to safely outsource party's data for reuse of processed privacy data.

Acknowledgments

This work was supported by the Support Plan of Scientific and Technological Innovation Team in Universities of Henan Province (No. 20IRTSTHN013), the Youth Talent Support

Program of Henan Association for Science and Technology (No. 2021HYTP008), the Fundamental Research Funds for the Universities of Henan Province (No. NSFRF210312), and the Project supported by the PhD Foundation of Henan Polytechnic University (No. B2021-41).

References

[1] Freedman, Michael J, Kobbi Nissim, and Benny Pinkas. "Efficient Private Matching and Set Intersection". International conference on the theory and applications of cryptographic techniques. 2004.

[2] Yang, Li, et al. "Achieving Privacy-Preserving Sensitive Attributes for Large Universe Based on Private Set Intersection." Information Sciences 582 pp.529-46.2022.

[3] Ruan, Ou, and Jianqiang Zeng. "A Delegated Offline Private Set Intersection Protocol for Cloud Computing Environments". Proceedings of the 2022 2nd International Conference on Control and Intelligent Robotics. 2022.

[4] Lv Siyi, Ye Jinhui, and Yin Sjie. "Unbalanced Private Set Intersection Cardinality Protocol with Low Communication Cost." Future Generation Computer Systems 102 pp.1054-61.2020.

[5] Yang, Yaxi, et al. "Prirange: Privacy-Preserving Range-Constrained Intersection Query over Genomic Data." IEEE Transactions on Cloud Computing.2022.

[6] Singh, Priyanka, et al. "Ppcontacttracing: A Privacy-Preserving Contact Tracing Protocol for Covid-19 Pandemic." arXiv preprint arXiv:.06648 20.8 pp.6648-57.2020.

[7] Duong, T., D. H. Phan, and T. Ni. "Catalic: Delegated Psi Cardinality with Applications to Contact Tracing". International Conference on the Theory and Application of Cryptology and Information Security. 2020.

[8] Pinkas, Benny, Thomas Schneider, and Michael Zohner. "Faster Private Set Intersection Based on Ot Extension". 23rd USENIX Security Symposium (USENIX Security 14). 2014.

[9] Kolesnikov, Vladimir, et al. "Efficient Batched Oblivious Prf with Applications to Private Set Intersection". Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. 2016.

[10] Chen, Hao, Kim Laine, and Peter Rindal. "Fast Private Set Intersection from Homomorphic Encryption." Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

[11] Chen, Hao, et al. "Labeled Psi from Fully Homomorphic Encryption with Malicious Security". Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018.

[12] Pinkas, Benny, et al. "Spot-Light: Lightweight Private Set Intersection from Sparse Ot Extension". Annual International Cryptology Conference. 2019.

[13] Freedman, Michael J., et al. "Efficient Set Intersection with Simulation-Based Security." Journal of Cryptology 29.1 pp.115-55.2016.

[14] Pagh, Rasmus. "On the Cell Probe Complexity of Membership and Perfect Hashing". Proceedings of the thirty-third annual ACM symposium on Theory of computing. 2001.

[15] Abadi, A., S. Terzis, and C. Dong. "O-Psi: Delegated Private Set Intersection on Outsourced Datasets". IFIP International Information Security and Privacy Conference. 2015.

[16] Pinkas, Benny, et al. "Phasing: Private Set Intersection Using Permutation-Based Hashing". 24th USENIX Security Symposium (USENIX Security 15). 2015.

[17] Dong, C., et al. "Fair Private Set Intersection with a Semi-Trusted Arbiter". Data and Applications Security and Privacy XXVII. 2013.

[18] Pinkas, Benny, Thomas Schneider, and Michael Zohner. "Scalable Private Set Intersection Based on Ot Extension." Acm Transactions on Privacy and Security (TOPS) 21.2 pp.1-35.2018.

[19] Pinkas, Benny, et al. "Psi from Paxos: Fast, Malicious Private Set Intersection". Annual International Conference on the Theory and Applications of Cryptographic Techniques. 2020.

[20] Ishai, Yuval, et al. "Extending Oblivious Transfers Efficiently". Annual International Cryptology Conference. 2003.

[21] Dong, Changyu, Liqun Chen, and Zikai Wen. "When Private Set Intersection Meets Big Data: An Efficient and Scalable Protocol". Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. 2013.

[22] Asharov, Gilad, et al. "More Efficient Oblivious Transfer and Extensions for Faster Secure Computation". Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. 2013.

[23] ZHANG, En, and Ganggang JIN. "Cloud Outsourcing Multiparty Private Set Intersection Protocol Based on Homomorphic Encryption and Bloom Filter." Journal of Computer Applications 38.8 pp.2256-60.2018.

[24] Tajima, Arisa, Hiroki Sato, and Hayato Yamana. "Outsourced Private Set Intersection Cardinality with Fully Homomorphic Encryption". 2018 6th International Conference on Multimedia Computing and Systems (ICMCS). 2018.

[25] Abadi, Aydin, et al. "Efficient Delegated Private Set Intersection on Outsourced Private Datasets." IEEE Transactions on Dependable and Secure Computing 16.4 pp.608-24.2017.

[26] Kolesnikov, Vladimir, et al. "Practical Multi-Party Private Set Intersection from Symmetric-Key Techniques". Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

[27] Zhang, En, et al. "Efficient Multi-Party Private Set Intersection against Malicious Adversaries". Proceedings of the 2019 ACM SIGSAC conference on cloud computing security workshop. 2019.

[28] Zhang, Jing, et al. "Outsourced Mutual Private Set Intersection Protocol for Edge-Assisted Iot." 2021.2021.

[29] Ghosh, Satrajit, and Tobias Nilges. "An Algebraic Approach to Maliciously Secure Private Set Intersection". Annual International Conference on the Theory and Applications of Cryptographic Techniques. 2019.

[30] Canetti, Ran. "Universally Composable Security: A New Paradigm for Cryptographic Protocols". Proceedings 42nd IEEE Symposium on Foundations of Computer Science. 2001.

[31] Kamara, S., et al. "Scaling Private Set Intersection to Billion-Element Sets". Springer Berlin Heidelberg. 2014.

[32] Hazay, Carmit, and Muthuramakrishnan Venkitasubramaniam. "Scalable Multi-Party Private Set-Intersection". IACR international workshop on public key cryptography. 2017.