The ISC Int'l Journal of Information Security

ISeCure

July 2018, Volume 10, Number 2 (pp. 117–127)

http://www.isecure-journal.org

On the Security of O-PSI a Delegated Private Set Intersection on Outsourced Datasets[☆]

Mehdi Mahdavi Oliaee[1], Mahshid Delavar[1], Mohammad Hassan Ameri[1,*], Javad Mohajeri[1], and Mohammad Reza Aref[2]

[1] *Electronics Research Institute of Sharif university of Technology, Tehran, Iran*

[2] *Department of Electrical Engineering, Sharif university of Technology, Tehran, Iran*

ARTICLE INFO.

Article history:

Received: 27 February 2018

First Revised: 5 March 2018

Last Revised: 24 June 2018

Accepted: 26 July 2018

Published Online: 30 July 2018

Keywords:
Private Set Intersection (PSI), Cloud Server, Eavesdropper Attack, O-PSI.

ABSTRACT

In recent years, determining the common information privately and efficiently between two mutually mistrusting parties have become an important issue in social networks. Many Private Set Intersection (PSI) protocols have been introduced to address this issue. By applying these protocols, two parties can compute the intersection between their sets without disclosing any information about components that are not in the intersection. Due to the broad range of computational resources that the cloud can provide for its users, determining the set intersection by cloud may decrease the computational cost of the users. The proposed protocols by Abadi *et al.* are two protocols in this context. In this paper, we show that their protocols are vulnerable to eavesdropping attack. Also, a solution is proposed to secure the protocol against mentioned attack. Moreover, we analyze the performance of both O-PSI and modified O-PSI protocols and show that our scheme is comparable with the O-PSI protocol. Actually, one trivial solution for the Abadi *et al.*'s proposed schemes is to use a secure channel like TLS. However, in the performance evaluation, we compare our applied modification with this trivial solution, and show that our proposed modification is more efficient as some extra encryptions imposed by TLS are no longer required.

1 Introduction

In a wide-range of real world applications such as homeland security and social networks, the involving parties want to find out the intersection between their set without revealing any other information. For example, assume a situation that a security agency wants to compare its list of suspects with a flight passenger list. In this case, the names of suspects must be kept hidden from the airline and the agency should not find out any information about the other passengers to protect their privacy.

Therefore, we need a protocol such that two parties could find the intersection set without revealing their private information. The protocols that meet this objective are called Private Set Intersections (PSI)

[☆] An earlier version of this paper has already been presented in ISCISC2017.

[*] Corresponding author.

Email addresses: mahdavioliaiy@ee.sharif.edu (M. Mahdavi Oliaee), m.delavar@sharif.ir (M. Delavar), ameri_mohammadhasan@ee.sharif.edu (M.H. Ameri), mohajer@sharif.ir (J. Mohajeri), aref@sharif.ir (M.R. Aref)

ISSN: 2008-2045 © 2018 ISC. All rights reserved.

protocols.

The first PSI protocol was introduced by Freedman *et al.* [1] in 2004. In this protocol, Paillier cryptosystem [2] has been applied. In some applications, it is only required that the cardinality of the intersection set be specified. To this end, Cristofaro *et al.* introduced the notion of Private Set Intersection Cardinality (PSI-CA) [3]. To prevent a party from modifying its information set and threatening the other party's privacy, the Authorized Private Set Intersection (APSI) notion was first introduced by Cristofaro *et al.* [4]. In the APSI protocols, the parties' information sets are signed by an authorized certification authority. The Authorized PSI-CA (APSI-CA) protocol was also presented by Cristofaro *et al.* in [3]. In addition, they demonstrated that how a PSI-CA protocol can be combined with a PSI protocol such that server decide on engaging in PSI or not based on the order of the intersection set which is privately obtained using PSI-CA [3]. If the order of the intersection set be higher than a threshold value, then the PSI protocol can be successfully executed. The threshold value is determined according to the server predefined security policies.

In some applications, the order of the party's information set is important and should be kept secret. For example, if the party's set be a list of people who have a particular disease, its size is important because it may effect on smuggling and selling drugs and the supply and demand of the drug market or it may inspire fear in the society. Since the communication cost of mentioned PSI protocols is linearly increased with the order of the party's information set, each dishonest party may find out the order of party's set. Ateniese *et al.* addressed this issue in their work and introduced the Size-Hiding Private Set Intersection (SHI-PSI) protocol which hides the order of party's set from the server [5].

The mentioned protocols cannot be applied for determining the intersection of information sets of more than two parties. In this regard, Kissner *et al.* in [6] introduced a PSI protocol named Multiparty PSI (MPSI) protocol which makes finding the intersections of more than two information sets possible.

In the mentioned protocols and some other protocols like [7–16], two (or more) parties jointly compute the intersection set which impose a high computational complexity to each of them. Since some parties may have limited computational resources, it will be more efficient to use a third party with high resources, such as a cloud server, to gather the information set of both parties and facilitate computing the intersection set between them. The PSI protocols with this property are named delegated PSI protocols [17]. However, the cloud server cannot be fully trusted [18], [19]. So, in the delegated PSI protocols, to preserve the privacy of involved parties, their information set should be outsourced to the cloud server in encrypted form. In this way, the cloud server will not be able to infer any sensitive information about the outsourced data.

There are two important challenges in these types of protocols. The first challenge is that how the cloud server can decrypt the outsourced data and compute the intersection set without knowing the decryption keys. This problem is more severe when the ciphertexcts are computed with different public keys. The second challenge is that how the parties can verify the correctness of the computed intersection set by the cloud server.

Zheng and Xu presented a verifiable delegated PSI protocol in [17] to address the mentioned issues. However, their protocol is not efficient due to the heavy computational load of bilinear pairings applied in it. The execution time of a bilinear pairing is significantly more than a modular exponentiation. Therefore, in [20], Mahdavi *et al.* introduced a protocol based on modular exponentiation which has much lower computational cost compared to [17].

The main issue of the aforementioned protocols and some other delegated PSI protocols such as [21–26] is the possibility of computing the size of the intersections by the cloud. Abadi *et al.* addressed this issue and solved it in [27] and [28] (which is the journal version of [27]). They named their protocols Outsourced-PSI (O-PSI) and Efficient Outsourced-PSI (EO-PSI) and claimed that they are secure against the corruption of one of the parties. Also this problem has been solved in [29–31].

Our Contribution. In this paper, we show that, in contrast to the claim of Abadi *et al.* in [27] and [28], if one of the parties is corrupted by an adversary, it can apply the eavesdropping attack to find out the polynomial representation of the other party's information set.

As another contribution, we propose a new delegated PSI protocol named *modified O-PSI protocol* which is secure against the mentioned attack. To prove the security of the proposed protocol, we first present an exact security definition. Then, we formally prove that its security is reducible to the semantic security of the applied homomorphic encryption scheme.

Moreover, we analyze the performance of both O-PSI and modified O-PSI protocols and show that our scheme is comparable with the O-PSI protocol. Actually, one trivial solution for the Abadi *et al.*'s proposed schemes is using a secure channel like TLS. However, in the performance evaluation, we compare our applied modification with this trivial solution, and show that our proposed modification is more efficient as some extra encryption imposed by TLS are no longer required.

Organization. The rest of the paper is organized as follows: In Section 2, we introduce the basic definitions and required notations. Section 3 is dedicated to describing the O-PSI protocol. In Section 4, we propose an eavesdropping attack on the O-PSI protocol. In Section 5, we propose a solution to make the O-PSI protocol secure against the mentioned attack. In Section 6, the security of our scheme is formally evaluated, and Section 7 presents the performance evaluation of our scheme. We conclude the paper in Section 8.

2 Preliminaries

In this section, we first express the commonly used notations. Then, we review Paillier cryptosystem, and the polynomial representation of sets which is widely applicable in the private set intersection protocols.

The notations that are applied in the rest of the paper has been shown in Table 1

Table 1. Frequently used notations in the proposed schemes

Notation	Description
$S_A = \{s_1, \dots, s_d\}$	Information set of the party A
d	The cardinality of the information set of each party
$\tau_A(x) = \prod_{s_i \in S} (x - s_i)$	The polynomial representation of the information set of the party A
n	An integer chosen by the cloud such that $d \leq \frac{n}{2}$
sk_A, pk_A	Secret and public keys of the party A in Paillier cryptosystem
$E_{pk_B}(.)$	Encryption with the public key of the party B
$D_{sk_B}(.)$	Dencryption with the secret key of the party B
M	Plaintext
lcm	Least Common Multiple

2.1 Paillier public Key Cryptosystem

In this part we review Paillier cryptosystem which is used in Abadi *et al.*'s O-PSI scheme [27]. Before introducing the Paillier scheme, we refer to its homomorphic features. Let $E_{pk}(.)$ be the encryption algorithm of Paillier cryptosystem.

- Given two ciphertexcts, $E_{pk}(a)$ and $E_{pk}(b)$, we will have $E_{pk}(a) . E_{pk}(b) = E_{pk}(a + b)$.
- Given a ciphertext, $E_{pk}(a)$ and a constant, b, we will have $E_{pk}(a)^{\ b} = E_{pk}(a.b)$.

Paillier cryptosystem contains three algorithms Key Generation, Encryption and Decryption. This cryptosystem works as follows [2].

Key Generation In this algorithm, two large prime numbers p and q are chosen based on the security parameter λ and the integer N is set equal to pq. Then, $t = lcm\,(p-1, q-1)$ is computed where lcm stands for the least common multiple. After that, a random number, $g \in Z^*_{N^2}$ is chosen such that $\mu = \left(L\left(g^t\ modN^2\right)\right)^{-1} \bmod N$ exists where $L(x) = \frac{x-1}{N}$. Finally, the public key and private key of this cryptosystem will respectively be set as (N, g) and (t, μ).

Encryption To encrypt a message, $M \in Z_N$, a random value, r $\in Z^*_N$ is selected and the ciphertext is computed such as follows:

$$C = E_{pk}(M) = g^M . r^N modN^2 \tag{1}$$

Decryption To decrypt the ciphertext, $C = E_{pk}(M)$, the following equation is computed:

$$M = D_{sk}(C) = L\left(C^t modN^2\right) .\mu \text{ mod N} \tag{2}$$

2.2 Polynomial Representaiton of Sets

In many protocols (e.g., [1], [7] and [32]), the polynomial representation of sets is widely used. Let F be a field. In this representation, each subset of F, $S = \{x_0, \dots, x_{n-1}\}$ $S = \{s_1,\ \dots,\ s_n\}$ can be represented via a polynomial, $\tau(x) = \prod_{s_i \in S} (x - s_i)$ in the polynomial ring $F[x]$. So, each element of S is a root of $\tau(x)$.

We should mention that to determine a unique n-degree polynomial by Lagrange Interpolation technique, at least $n+1$ points of its curve is required. So, given a set of n+1 points, $\{(x_0, y_0), \dots,\ (x_n, y_n)\}$, a polynomial with degree of n can be calculated.

3 An Overview On The O-PSI Protocol

In this section, the O-PSI protocol [27] is described. This protocol has been designed for finding the intersection of the information set of two parties, A and B, by help of a cloud sever. The protocol consists of three algorithms: Setup, Outsource and Set intersection.

Setup- Consider a big enough public finite field, F and a pseudorandom function, $f : \{0, 1\}^l \times Z_n \to F$ in which n is determined by the cloud server such that $n/2$ is an upper bound for the order of the parties' information sets. The function, f, pseudo-randomly maps an $(l+\log_2 n)$-bit string to an element in F. Also, without loss of generality, it has been assumed that the order of the information sets of both parties, A and B is equal to d. The vector, $X = \{x_0, \ldots, x_{n-1}\}$ is constructed by the cloud and published among all parties. Each party uses this vector during running the protocol.

Outsource- The party A with the information set $S_A = \left\{s_i^{(A)}, 1 \leq i \leq d\right\}$ runs this algorithm through the following steps:

(1) It generates a Paillier key pair (pk_a, sk_a) and publishes the public key, pk_a. Also, it chooses a random private key k_a.
(2) It constructs the polynomial τ_A as a representation of its information set S_A such that $\tau_A = \prod_{s_i^{(A)} \in S_A} (x - s_i^{(A)})$. Then, it computes the vector $\mathbf{y}^A$, which each of its elements $y_i^{(A)}$ is equal to $\tau_A(x_i)$ for $0 \leq i \leq n-1$.
(3) Then, it generates the vector $\mathbf{v}^{(A)}$, by computing $v_i^{(A)} = y_i^{(A)}.r_i^{(A)}$ where $r_i^{(A)} = f(k_a, i)$. In fact, $\mathbf{v}^{(A)}$ is a blinded version of $\mathbf{v}^{(A)}$. The vector $\mathbf{v}^{(A)}$, is outsourced to the cloud.

The party B follows the same procedure as the party A does.

Set Intersection In this algorithm, the party B finds out the intersection of its information set and the party A's through following steps:

(1) At first, the party B generates the vector $\mathbf{e}^{(B)}$, that its elements are $e_i^B = E_{pk_B}\left(r_i^{(B)}\right); 0 \leq i \leq n-1$. It sends the vector $\mathbf{e}^{(B)}$, to the party A.
(2) The party A also generates the vector $\mathbf{e}^{(A)}$, by computing $e_i^{(A)} = \left(e_i^{(B)}\right)^{\left(r_i^{(A)}\right)^{-1}} = E_{pk_B}\left(r_i^{(B)}.\left(r_i^{(A)}\right)^{-1}\right)$; $0 \leq i \leq n-1$ as its vector elements and sends it to the cloud.
(3) The cloud, after receiving the vector $\mathbf{e}^{(A)}$, randomly chooses two d-degree polynomials, w_A and w_B and generates the vectors, $\mathbf{w}^{(A)}$ and $\mathbf{w}^{(B)}$ such that $w_i^{(A)} = w_A(x_i)$ and $w_i^{(B)} = w_B(x_i)$ for $0 \leq i \leq n-1$.
(4) The cloud generates the vector t by using the stored and generated vectors, $\mathbf{v}^{(A)}$, $\mathbf{v}^{(B)}$, $\mathbf{w}^{(A)}$ and$\mathbf{w}^{(B)}$ for each $0 \leq i \leq n-1$, using homomorphic properties of Paillier cryptosystem such as follows:

$$\begin{aligned} t_i &= \left(e_i^{(A)}\right)^{v_i^{(A)}.w_i^{(A)}} .E_{pk_B}\left(v_i^{(B)}.w_i^{(B)}\right) \\ &= E_{pk_B}\left(r_i^{(B)}.\left(r_i^{(A)}\right)^{-1}.y_i^{(A)}.r_i^{(A)}.w_i^{(A)}\right) \\ &\quad .E_{pk_B}\left(y_i^{(A)}.r_i^{(A)}.w_i^{(B)}\right) \end{aligned}$$

$$\begin{aligned} t_i &= \left(e_i^{(A)}\right)^{v_i^{(A)}.w_i^{(A)}} .E_{pk_B}\left(v_i^{(B)}.w_i^{(B)}\right) \\ &= E_{pk_B}\left(r_i^{(B)}.\left(r_i^{(A)}\right)^{-1}.y_i^{(A)}.r_i^{(A)}.w_i^{(A)}\right) \\ &\quad .E_{pk_B}\left(y_i^{(A)}.r_i^{(A)}.w_i^{(B)}\right) \\ &= E_{pk_B}\left(r_i^{(B)}\left(y_i^{(A)}.w_i^{(A)} + y_i^{(B)}.w_i^{(B)}\right)\right). \end{aligned}$$

Then, it sends the vector, $\mathbf{t}$ to the party B.
(5) After receiving the vector $\mathbf{t}$, the party B computes the vector $\mathbf{z}$ such as follows:

$z_i = D_{sk_B}(t_i).(r_i^{(B)})^{-1} = y_i^{(A)}.w_i^{(A)} + y_i^{(B)}.w_i^{(B)}$; $0 \leq i \leq n-1$

Thus, by computing the interpolation of the points (x_i, z_i), the polynomial $Z(x) = \tau_A w_A + \tau_B w_B$ is obtained which its roots are the elements of the intersection of information sets of two parties, A and B.

The procedure of the O-PSI protocol is depicted in Figure 1.

4 Our Proposed Attack On The O-PSI Protocol

In [27], the authors by considering a semi-honest adversary model have proved that their protocol is secure. In this model, the adversary can corrupt any of involved parties or the cloud server. In their security model, they did not consider the situation in which the adversary can eavesdrop the communication between one of the parties and the cloud server. We show that the O-PSI protocol is vulnerable against the eavesdropping attack as follows.

In the second step of the Set Intersection algorithm, the party B may eavesdrop the communication between the party A and the cloud server and find the elements of the vector $\mathbf{e}^{(A)}$. Since these elements have been encrypted by the public key of the party B, it can easily decrypt them and obtain the values of $\left(r_i^{(A)}\right)^{-1}$. In this way, the party B can compute the vector $\mathbf{y}^{(A)}$ by multiplying the obtained values by the vector $\mathbf{v}^{(A)}$. Then, it can compute the polynomial τ_A using point-value pairs (x_i, y_i) through Lagrange interpolation technique. Since the roots of the polynomial τ_A are the elements of the party A's set, the party B will be able to find all the elements of information set of the party A [33].

Remark 1: It also worth mentioning that our proposed eavesdropping attack is also applicable on

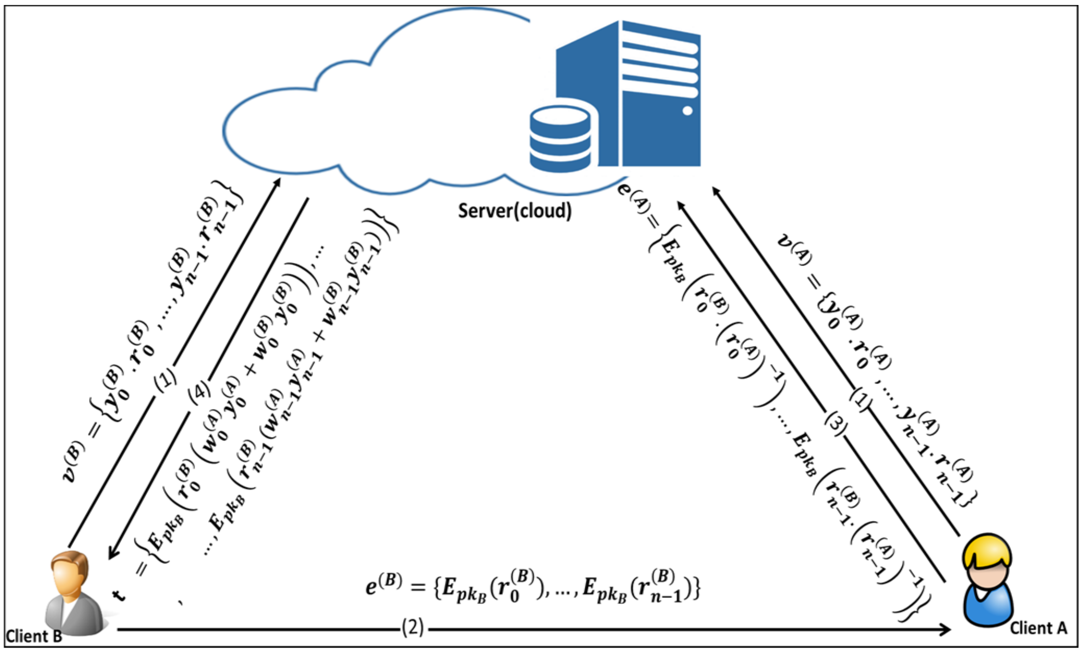
Server(cloud)
Client B
Client A
$v^{(B)} = \{y_0^{(B)}.r_0^{(B)}, \ldots, y_{n-1}^{(B)}.r_{n-1}^{(B)}\}$
(1)
$t = \{E_{pk_B}(r_0^{(B)}(w_0^{(A)}y_0^{(A)} + w_0^{(B)}y_0^{(B)})), \ldots, E_{pk_B}(r_{n-1}^{(B)}(w_{n-1}^{(A)}y_{n-1}^{(A)} + w_{n-1}^{(B)}y_{n-1}^{(B)}))\}$
(4)
$e^{(A)} = \{E_{pk_B}(r_0^{(B)}.(r_0^{(A)})^{-1}), \ldots, E_{pk_B}(r_{n-1}^{(B)}.(r_{n-1}^{(A)})^{-1})\}$
(3)
$v^{(A)} = \{y_0^{(A)}.r_0^{(A)}, \ldots, y_{n-1}^{(A)}.r_{n-1}^{(A)}\}$
(1)
$e^{(B)} = \{E_{pk_B}(r_0^{(B)}), \ldots, E_{pk_B}(r_{n-1}^{(B)})\}$
(2)

Figure 1. The Abadi's O-PSI protocol [27]

the proposed EO-PSI [28].

5 Our Proposed Protocol: The Modified O-PSI Protocol

In this section, we propose the modified O-PSI protocol which is secure against eavesdropping and collusion attacks.

Like the O-PSI protocol [27], the modified O-PSI protocol also contains three main algorithms which are Setup, Outsource and Set intersection. The Setup and Outsource algorithms are the same as what is presented in the O-PSI protocol except that in the Outsource algorithm of our protocol, the vectors, $\mathbf{v}^{(A)}$ and $\mathbf{v}^{(B)}$, are encrypted by cloud server's public key. Therefore, in the following, we just describe the Set Intersection algorithm in details. By assuming that the party B wants to find out the intersection set of its information set and the information set of the party A, this algorithm runs such as follows:

(1) At first, the party B generates the vector $\mathbf{e}^{(B)}$, where its elements are $e_i^B = E_{pk_B}\left(\left(r_i^{(B)}\right)^{-1}\right); 0 \le i \le n-1$. It requests the party A to run the protocol and sends the vector, $\mathbf{e}^{(B)}$ to the cloud. After receiving request from the party B similarly, the party A generates the vector,$\mathbf{e}^{(A)}$ and sends it to the cloud where its elements are $e_i^A = E_{pk_B}\left(\left(r_i^{(A)}\right)^{-1}\right)$.

(2) After receiving the vectors $\mathbf{e}^{(A)}$ and $\mathbf{e}^{(B)}$, the cloud randomly chooses two d-degree polynomials, w_A and w_B and generates the vectors, $\mathbf{w}^{(A)}$ such that its elements are defined as $w_i^{(A)} = w_A(x_i)$ for $0 \le i \le n-1$. Then, using homomorphic property of the Paillier cryptosystem and the vectors $\mathbf{v}^{(A)}$ and $\mathbf{v}^{(B)}$, the cloud generates the vector $\mathbf{t}$, where its elements are computed such as follows:

$$\begin{aligned} t_i &= E_{pk_B}\left(\left(r_i^{(A)}\right)^{-1}\right)^{w_i^{(A)}.D_{sk_c}\left(v_i^{(A)}\right)} \\ &.E_{pk_B}\left(\left(r_i^{(B)}\right)^{-1}\right)^{w_i^{(B)}.D_{sk_c}\left(v_i^{(B)}\right)} \\ &= E_{pk_B}\left(w_i^{(A)}.\left(r_i^{(A)}\right)^{-1}.y_i^{(A)}.r_i^{(A)}\right) \\ &.E_{pk_B}\left(w_i^{(B)}.\left(r_i^{(B)}\right)^{-1}.y_i^{(B)}.r_i^{(B)}\right) \\ &= E_{pk_B}\left(y_i^{(A)}.w_i^{(A)} + y_i^{(B)}.w_i^{(B)}\right) \end{aligned}$$

Then, the cloud sends the vector, $t = (t_0, \ldots, t_{n-1})$ to the party B.

(3) After receiving the vector $\mathbf{t}$, the party B computes each element of the vector $\mathbf{z}$, i.e., z_i, as follows:
$z_i = D_{sk_B}(t_i) = y_i^{(A)}.w_i^{(A)} + y_i^{(B)}.w_i^{(B)}$
Such that $z_i = Z(x_i)$.

Finally, the party B generates the polynomial $Z(x)$ by applying the Lagrange interpolation technique. The roots of this polynomial are the common elements of the information sets of the parties A and B.

The procedure of the modified O-PSI protocol has been depicted in Figure 2.

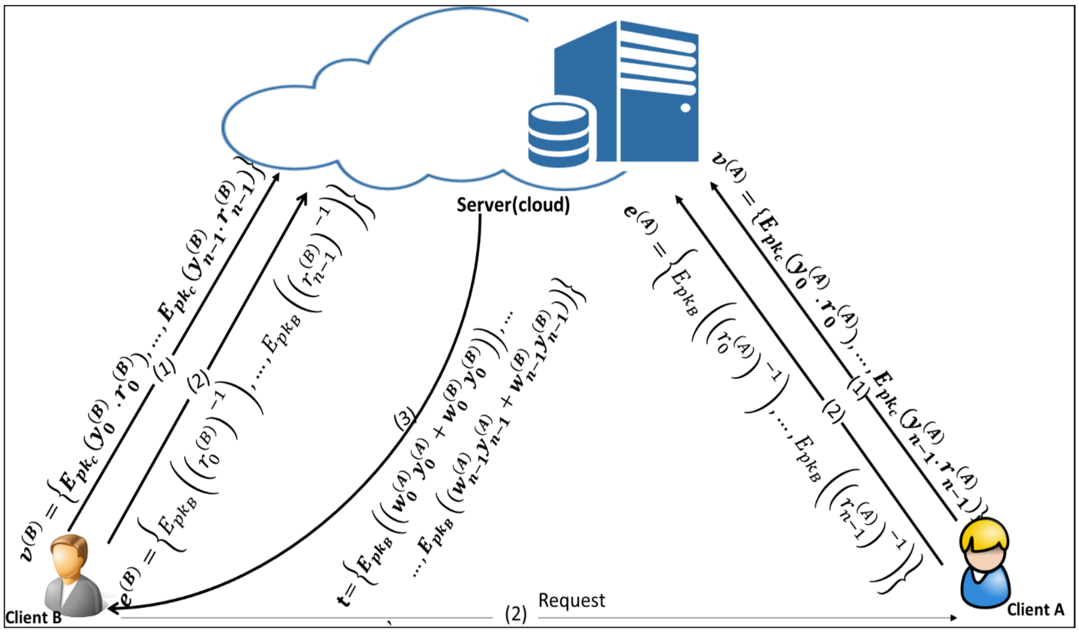
Server(cloud)
$v^{(B)} = \{E_{pk_c}(y_0^{(B)}.r_0^{(B)}), \dots, E_{pk_c}(y_{n-1}^{(B)}.r_{n-1}^{(B)})\}$
(1)
$e^{(B)} = \{E_{pk_B}((r_0^{(B)})^{-1}), \dots, E_{pk_B}((r_{n-1}^{(B)})^{-1})\}$
(2)
$t = \{E_{pk_B}((w_0^{(A)}y_0^{(A)} + w_0^{(B)}y_0^{(B)})), \dots, E_{pk_B}((w_{n-1}^{(A)}y_{n-1}^{(A)} + w_{n-1}^{(B)}y_{n-1}^{(B)}))\}$
(3)
$v^{(A)} = \{E_{pk_c}(y_0^{(A)}.r_0^{(A)}), \dots, E_{pk_c}(y_{n-1}^{(A)}.r_{n-1}^{(A)})\}$
(1)
$e^{(A)} = \{E_{pk_B}((r_0^{(A)})^{-1}), \dots, E_{pk_B}((r_{n-1}^{(A)})^{-1})\}$
(2)
(2) Request
Client B
Client A

Figure 2. The Modified O-PSI protocol

Note that similar to the O-PSI protocol, our proposed protocol can also be used in multi-party setting. In this case, the party B is interested in finding the intersection set of its information set and some other parties' $A_j (1 \leq j \leq m)$. To this end, it sends a request to all of them and the cloud server. The cloud computes the vector $\mathbf{t} = (t_0, , t_(n-1))$ for each $0 \leq i \leq n-1$ such as follows:

$$
\begin{aligned}
t_i &= E_{pk_B}\left(\left(r_i^{(B)}\right)^{-1}\right)^{w_i^{(B)}.D_{sk_c}\left(v_i^{(B)}\right)} \\
&\cdot \prod_{1\leq j\leq m} \left(E_{pk_B}\left(\left(r_i^{(A_j)}\right)^{-1}\right)\right)^{w_i^{(A_j)}.D_{sk_c}\left(v_i^{(A_j)}\right)} \\
&= E_{pk_B}\left(y_i^{(B)}.w_i^{(B)} + \sum_{1\leq j\leq m} w_i^{(A_j)} \cdot y_i^{(A_j)}\right)
\end{aligned}
$$

Then, the cloud sends the vector t to the party, B. In this way, the party B is able to construct the polynomial $Z(x)$ which its roots are the common elements of the information sets of the parties B and the other parties A_js.

6 Security Evaluation

In this section, we first introduce our security model and definition required to prove the security of the modified O-PSI protocol. Then, we present our security proof.

6.1 Security Model and Definition

Like the adversarial model presented in [27] and [28], we also assume that the adversary is semi-honest, and its main goal is to achieve the secret information of the other parties.

In each delegated PSI protocol Π, usually, three parties including a Server C and two parties A and B are involved, and an algorithm $F : \Lambda \times 2^U \times 2^U \rightarrow \Lambda \times \Lambda \times f_\cap$ is defined where the symbols Λ, 2^U and $f_\cap$ respectively denote the null symbol, the universe space of all possible information sets and the intersection set extractor function. For every tuple of inputs(Λ, S_A, S_B), the algorithm F outputs the tuple $(\Lambda, \Lambda, f_\cap(S_A, S_B) = S_A \cap S_B)$ where two first elements are sent to the cloud server C and the party A and the last one is received by the party B.

In the semi-honest model, the protocol Π is secure if in the end of protocol, each party only obtains its dedicated information and not achieve any more information about the other parties. This security definition is formalized by means of the simulation paradigm [34]. Simulation-based security proof is a typical technique in proving the security of cryptographic protocols, and in this kind of approaches the security definitions is presented according to the view of the parties involved in the protocol. Actually, the view of each party is referred to all the messages which can be observed by the mentioned party. For this aim, the notation $view_u^\pi$ can be used which implies to all the messages which are observable by the party u when the protocol Π is executed. Beside the parties' views, the notation $Sim_u^{\Pi}(In, Out)$ is also used to imply the output of a probabilistic polynomial-time (PPT) simulator who simulates the real view of the party u by means the pair (In, Out). In the pair (In, Out), In, implies the party u's dedicated information and Out stands for the output of algorithm F which is sent to this party.

Based on the notations $view_u^{\Pi}$ and $Sim_u^{\Pi}(In, Out)$, the security of the O-PSI protocol is defined. Following, we present the formal security definition of a delegated private set intersection protocol.

Definition 1 [27, 28]. Let F be a deterministic algorithm as defined before. We say that the protocol Π securely runs the algorithm F in the presence of semi-honest adversaries if there exists probabilistic polynomial-time simulators $Sim_C^{\Pi}(\Lambda, \Lambda)$, $Sim_A^{\Pi}(S_A, \Lambda)$ and $Sim_B^{\Pi}(S_B, f_{\cap}(S_A, S_B))$ to simulate computationally distinguishable view form the real views $view_C^{\Pi}$, $view_A^{\Pi}$ and $view_B^{\Pi}$, correspondingly. Therefore, in a secure O-PSI protocol the following relations should be preserved simultaneously:

$$Sim_C^{\Pi}(\Lambda, \Lambda) \approx^{ind} view_C^{\Pi}$$

$$Sim_A^{\Pi}(S_A, \Lambda) \approx^{ind} view_A^{\Pi}$$

$$Sim_B^{\Pi}(S_B, f_{\cap}(S_A, S_B)) \approx^{ind} view_B^{\Pi}$$

In the above relations, the notation $\approx^{ind}$ implies that both sides of the relation are computationally indistinguishable.

In Definition 1, the relation $Sim_C^{\Pi}(\Lambda, \Lambda) \approx^{ind} view_C^{\pi}$ implies that the cloud server C, cannot infer any information more than the null symbol Λ, because its view is indistinguishable with the case when the protocol Π is executed based on the null information, Λ.

6.2 The Weakness of Security Proof of Abadi *et al.*

Abadi *et al.* [27] have formally proved that the O-PSI protocol is secure without considering that the party B may eavesdrop the communication between the cloud server and the party A. In Section IV, we showed that the O-PSI protocol is vulnerable against eavesdropping attack. In what follows, we will show that if Abadi *et al.* considered the communication between the cloud server and the party A in the view of B (in the case that the party B is malicious), then, the simulated view and the real view would be distinguishable from each other, and this means that running this protocol leaks some information for the party B more than its dedicated information.

If Abadi *et al.* considered the real view of the party B in the proof section of [27] such as follows:

$$Sim_B^{\Pi}(S_B, f_{\cap}(S_A, S_B)) \approx^{ind} view_B^{\Pi}$$

Then, according to their proof procedure, they should simulate the values, $e_i^{(A)} = E_{pk_B}\left(r_i^B\left(r_i^A\right)^{-1}\right)$ and $v^{(A)} = y_i^A . r_i^A$ respectively with $e'^{(A)}_i = E_{pk_B}\left(r'^B_i . \left(r'^A_i\right)^{-1}\right)$ and $v'^{(A)}_i = y'^A_i . r'^A_i$ where y'^A_i, r'^B_i and r'^A_i are chosen randomly by the simulator. In this way, since the party B possesses the secret key corresponding to pk_B and knows the value of r_i^B, it can distinguish between the value of $v_i^{(A)}$ and $v'^{(A)}_i$ through following procedure.

For $1 \leq i \leq n-1$, it computes $\left(e_i^{(A)}\right)^{v_i^{(A)}} = E_{pk_B}\left(r_i^B\left(r_i^A\right)^{-1} . v_i^{(A)}\right) = E_{pk_B}\left(r_i^B\left(r_i^A\right)^{-1} . y_i^A . r_i^A\right)$, and then decrypts it to obtain $r_i^B . y_i^A$. By dividing the result to r_i^B which is known by the party B, the value of y_i^A will be emerged, for $1 \leq i \leq n-1$. In a similar pattern, the party B by considering the values of $e_i^{(A)}$ and $v'^{(A)}_i$, will compute the values of $o_i = y'^A_i . r'^A_i . \left(r_i^A\right)^{-1}$.

As the values of y_i^A $(0 \leq i \leq n-1)$ are n distinct points of a d-degree polynomial, by selecting each subset of $(d+1)$-points from these n points, and running the Lagrange interpolation, we can elicit a unit polynomial, while by choosing different subsets of $(d+1)-$ points from the values of o_i $(0 \leq i \leq n-1)$, the party B will extract different polynomials with a high probability negligibly lower than one.

In this way, the party B can distinguish between the values of $v_i^{(A)}$ and $v_{i'}^{(A)}$, and consequently between the real view and the simulated view.

With regard to this weakness, we have modified their scheme to enhance its security. In what follows, in the proposed security proof, we consider the complete view of the parties, and we will show that even though the party B eavesdrops the communication flaw, our scheme remains secure.

6.3 Security Proof

This part is presented with the aim of security proof of the proposed O-PSI in the semi-honest model.

Theorem 1. *The O-PSI protocol is secure in the presence of semi-honest adversary, if the using homomorphic encryption scheme is semantically secure.*

Proof. To prove the security of the proposed scheme, three cases will be considered such that in each case only one of the parties has been corrupted. For more clarification, it should be implied that in each case, the simulator with the corresponding party's input and output is invoked.

Case 1 (Corrupted cloud server): In the case of corrupted server, we will show that we can construct the simulator $Sim_C^{\Pi}(\Lambda,\ \Lambda)$ which can produce a computationally indistinguishable view from the view of the server from the real execution of the protocol. It should be noted that the server's view in the real execution is as follows:

$$view_c^{\Pi} = \left\{\Lambda,\ v^{(A)},\ v^{(B)},\ Compute, e^{(B)},\ e^{(A)}, \Lambda\right\}$$

In the above relation, $\mathbf{v}^{(A)}$ and $\mathbf{v}^{(B)}$ are the blinded form of set representations corresponding to the A's and B's information set which are encrypted by the cloud's public key, Compute is the command which is queried by B to receive the intersection of her information with A's, and $\mathbf{e}^{(A)}$ and $\mathbf{e}^{(B)}$ are the encrypted vectors.

Now, we simulate the view according to the $Sim_C^{\Pi}(\Lambda,\ \Lambda)$ which is conducted through the following procedure. First of all, $Sim_C^{\Pi}(\Lambda,\ \Lambda)$ will append the null symbol, Λ to the view. After that, $Sim_C^{\Pi}(\Lambda,\ \Lambda)$ picks two d-element sets S'_A and S'_B which are generated uniformly at random. Moreover, for the pseudorandom function f, it opts two random keys k'_A and k'_B, and represents S'_A in terms of its corresponding polynomial. Finally, with the help of the public values, it evaluates the polynomial, blinds and encrypts the evaluation results by means of $r_i^{(A)'} = f\left(k'_A, i\right)$ for $0 \le i \le n1$ and the cloud's public key such that the results is $\mathbf{v}^{(A')}$. In the same way and similar to this scenario, the value of $\mathbf{v}^{(B')}$ can be extracted. After the computation of $\mathbf{v}^{(A')}$ and $\mathbf{v}^{(B')}$, they will be appended to the view of the simulation. Also, the simulator computes $e_i^{(A)'} = E_{pk_B}\left(\left(r_i^{(A)'}\right)^{-1}\right)$ and $e_i^{(B)'} = E_{pk_B}\left(\left(r_i^{(B)'}\right)^{-1}\right)$ by using the party B's public key. In addition, the simulator appends the Compute command to the view.

Case 2 (Corrupted party A): In the case of corrupted party A, we will show that it is possible to construct the simulator $Sim_A^{\Pi}(S_A,\ \Lambda)$ which can produce a computationally indistinguishable view from the view of the party A from the real execution the protocol. It should be noted that the party A's view in the real execution is as follows:

$$\mathrm{view}_{\mathrm{A}}^{\Pi} = \left\{\mathrm{S_A},\ r^{(A)}, e^{(B)}, v^{(B)}, \mathbf{t},\ \Lambda\right\}$$

The simulator $Sim_A^{\Pi}(S_A,\ \Lambda)$ computes the view of the party A according to the following procedure.

The simulator $Sim_A^{\Pi}(S_A,\ \Lambda)$ first creates an empty view, and then appends Λ and $\mathbf{r}^{(A)'}$ which is chosen uniformly at random.

After that, $Sim_A^{\Pi}(S_A,\ \Lambda)$ picks two d-element sets S_A and S'_B which are generated uniformly at random. Moreover, for the pseudorandom function f, it opts a random key k'_B, and represent S'_B in terms of its corresponding polynomial.

Finally, the values of the resulting polynomial for each elements of the vector $\mathbf{x} = (x_0, \ldots x_{n-1})$ are evaluated and the results are blinded by means of $r_i^{(B)'} = f\left(k'_B, i\right)$ for $0 \le i \le n1$ in which the key k'_B is randomly chosen. The result of blinding and encrypting process is $\mathbf{v}^{(B)'}$. Also, it computes the $\mathbf{e}_i^{(B)'} = \mathrm{E}_{pk_B}\left(r_i^{(B)'}\right)^{-1}$. After that the values of $\mathbf{v}^{(B)'}$ and $\mathbf{e}^{(B)'}$ will be appended to the view. In the real view, $\mathbf{v}^{(B)}$ is blinded with the outputs of a pseudorandom function and encrypted by the party B's public key. Similarly, in the simulated view, the same scenario is considered for the value of $\mathbf{v}^{(B)'}$. Because of the semantic security of the applied encryption scheme, we can conclude that the distributions of $\mathbf{v}^{(B)}$, $\mathbf{v}^{(B)'}$ are computationally indistinguishable from each other.

It also randomly chooses two d-degree polynomials $\mathbf{w}'_A$ and $\mathbf{w}'_B$. Finally, for each value of i the simulator computes the value of $t'_i = E_{pk_B}\left(w_i^{(A)'} \cdot y_i^{(A)'} + w_i^{(B)'} \cdot y_i^{(B)'}\right)$ and appends it to the view. It can be easily seen that the distributions of $\mathbf{t}$ and $\mathbf{t}'$ are computationally indistinguishable from each other.

Case 3 (Corrupted party B): In this case we construct the simulator $Sim_B^{\Pi}(S_B,\ f(S_A \cap S_B))$ which can produce a computationally indistinguishable view from the real view of the party B. It should be noted that the party B's view in the real execution is as follows:

$$view_B^{\pi} = \left\{S_B,\ r^{(B)},\ Permit,\ t,\ e^{(A)}, v^{(A)},\ f_{\cap}(S_A,\ S_B)\right\}$$

The simulator $Sim_B^{\Pi}(S_B,\ f(S_A \cap S_B))$ computes the view of the party B as follows:

It first creates an empty view, and then appends the null symbol Λ and the random vector $\mathbf{r}^{(B)'}$ which is chosen uniformly at random to the view. For the pseudorandom function f, it selects two random keys k'_A and k'_B. It should be noted that the information set S'_A is chosen in such a way that $S'_A \cap S_B = f_{\cap}(S_A,\ S_B)$ and the remaining elements of S'_A are chosen uniformly at randomly. After that, the simulator represents S'_A in terms of its corresponding polynomial. Finally, the values of the resulting polynomial for each elements of the vector $\mathbf{x} = (x_0, \ldots x_{n-1})$ are evaluated, the results are blinded by means of $r_i^{(A)'} = f\left(k'_A, i\right)$ for $0 \le i \le n-1$, and encrypted by B's public key such that the results is the vector $\mathbf{v}^{(A)'}$. In the same way and similar to this scenario, the vector $\mathbf{v}^{(B)'}$ can be extracted. After the computation of $\mathbf{v}^{(A)'}$ and $\mathbf{v}^{(B)'}$, they will be appended to the simulation view. In addition, the simulator generates the

Permit command string and appends it to the view. For each ,$0 \leq i \leq n-1$, the simulator also computes the values $e_i^{(A)'} = E_{pk_B}\left(\left(r_i^{(A)'}\right)^{-1}\right)$ and appends the resulted vector $\mathbf{e}^{(A)'}$ and Λ to the view. Similar to the previous case, it can be shown that the values of $\mathbf{v}^{(A)'}$ and $\mathbf{v}^{(A)}$ are computationally indistinguishable.

To complete the simulation, the simulator $Sim_B^{\Pi}(S_B, f_{\cap}, (S_A, S_B))$ picks the command string "Permit" which is selected according to a valid format, and appends it to view. Then, similar to the previous case, based on S'_A and S_B the vectors $y^{(A)'}$ and $y^{(B)'}$ are computed. Moreover, the simulator Sim_B randomly selects the two d-degree polynomials $\mathbf{w}'_A$ and $\mathbf{w}'_B$ and after evaluation of them by using the public value $\mathbf{x}$, it extracts $\mathbf{w}^{(A)'}$ and $\mathbf{w}^{(B)'}$.

Finally, for each value of $0 \leq i \leq n-1$ the simulator computes $t'_i = E_{pk_B}(w_i^{(A)'} \cdot y_i^{(A)'} + w_i^{(B)'} \cdot y_i^{(B)'})$ and sets the vector $\mathbf{t}' = (t'_0, \ldots, t'_{n-1})$. Then, it appends $\mathbf{t}'$ along with the intersection set, $f_{\cap}(S_A, S_B)$ to the view. It can be seen that the distributions of $\mathbf{t}$ and $\mathbf{t}'$ are computationally indistinguishable from each other as the using encryption scheme is semantically secure.

According to the three discussed cases, it can be seen that all the conditions of Definition 1 are hold simultaneously, and consequently we can draw the conclusion that the resulting and improved O-PSI scheme is provably secure.

7 Performance Evaluation

In this section, we compare our proposed O-PSI protocol with the O-PSI protocol [27]. To compare these protocols, we considered the computational overhead of the outsourcing and set intersection algorithms in the both protocols. According to Table 2, it can be seen that in the set intersection algorithm, the computational overheads for the party A and the cloud server are similar for the both protocols, while, this overhead for the party B, in our protocol is significantly less than the same value in the O-PSI protocol.

Although, in the outsource algorithm, the computational overhead of our protocol is more than [27], but it should be noted that the outsource algorithm can be considered as an off-line phase which is run for just one time in the beginning of the protocol. Therefore, it does not affect the efficiency of the proposed scheme.

The comparison results of the computational overhead of our protocol and O-PSI protocol are presented in Table 2 in details.

As we have checked, the enhanced version of [27], i.e., [28], is also vulnerable to such attack. One trivial solution for making both [27] and [28] secure is using TLS protocol to providing a secure channel against eavesdropping attack. In this regard, we have shown our modification on the proposed protocol of [27] and [28] results in a better performance in the set intersection phase of the protocol. In Table 2, it is illustrated that our proposal is more efficient as there is no need for the extra encryption and decryption exposed by the TLS protocol.

In Table 2, E_T and $E_{T_{uu}}$ are the TLS encryption algorithm between the cloud and user and between the users, respectively. Similarly, D_T and $D_{T_{uu}}$ are the decryption algorithms of TLS scheme. Moreover, h is the length of hash function used in [28].

8 Conclusion

Finding the common information between two or more parties in a private and efficient manner is an important issue in many applications such as social networks. Private Set Intersection (PSI) protocols have been introduced to determine the intersection of two or more information sets such that no information about the components that are not belong to the intersection set is revealed. Using benefits of a cloud server in design of these protocols can reduce the computational cost of the parties. In this paper, we investigated the security of two delegated PSI protocols named O-PSI and EO-SPI, and showed that they are vulnerable against eavesdropping attack. We improved the security by proposing the modified O-PSI protocol. Through the simulation-based technique, we formally proved that the security of our scheme is reduced to the semantically security of the applied homomorphic encryption scheme. Also, we demonstrated the performance of the modified O-PSI protocol is comparable with the O-PSI protocol.

References

[1] Michael J Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *International conference on the theory and applications of cryptographic techniques*, pages 1–19. Springer, 2004.

[2] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

[3] Emiliano De Cristofaro, Paolo Gasti, and Gene Tsudik. Fast and private computation of cardinality of set intersection and union. In *International Conference on Cryptology and Network Security*, pages 218–231. Springer, 2012.

[4] Emiliano De Cristofaro and Gene Tsudik. Prac-

Table 2. The comparison results

	Computational Complexity					
Algorithm	Outsource			Set Intersection		
	[Ours]	Using TLS in [27]	Using TLS in [28]	[Ours]	Using TLS in [27]	Using TLS in [28]
The party A' side	$n.mult + n.E_{pk}$	$n.mult + n.E_T$	$h.PRF + h.E_T$	$n.E_{pk}$	$n.E_{pk} + n.E_T$	$3.E_T + h.E_{T_{uu}} + 2.D_{T_{uu}}$
The party B's side	$n.mult + n.E_{pk}$	$n.mult + n.E_T$	$n.PRF + n.E_T$	$n.E_{pk} + n.D_{sk}$	$n.mul+n.E_{pk}+ n.D_T + n.D_{sk}$	$2.E_{T_{uu}} + h.D_{T_{uu}} + h.D_T$
The cloud server side	$2n.D_{sk}$	$2n.D_T$	$2n.D_T$	$3n.mult + 2n.E_{pk}$	$3n.mult + n.D_T + n.E_T + 2n.E_{pk}$	$3.D_T + h.E_T$

tical private set intersection protocols with linear complexity. In *International Conference on Financial Cryptography and Data Security*, pages 143–159. Springer, 2010.

[5] Giuseppe Ateniese, Emiliano De Cristofaro, and Gene Tsudik. (if) size matters: size-hiding private set intersection. In *International Workshop on Public Key Cryptography*, pages 156–173. Springer, 2011.

[6] Lea Kissner and Dawn Song. Privacy-preserving set operations. In *Annual International Cryptology Conference*, pages 241–257. Springer, 2005.

[7] Atsuko Miyaji and Shohei Nishida. A scalable multiparty private set intersection. In *International Conference on Network and System Security*, pages 376–385. Springer, 2015.

[8] Dilip Many and Martin Burkhart. Fast private set operations with sepia. 2012.

[9] Peter Rindal and Mike Rosulek. Improved private set intersection against malicious adversaries. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 235–259. Springer, 2017.

[10] Changhee Hahn and Junbeom Hur. Scalable and secure private set intersection for big data. In *Big Data and Smart Computing (BigComp), 2016 International Conference on*, pages 285–288. IEEE, 2016.

[11] 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.

[12] Arun Thapa, Ming Li, Sergio Salinas, and Pan Li. Asymmetric social proximity based private matching protocols for online social networks. *IEEE Transactions on parallel and distributed systems*, 26(6):1547–1559, 2015.

[13] Zhiyi Shao, Bo Yang, and Yong Yu. Private set intersection via public key encryption with multiple keywords search. In *Intelligent Networking and Collaborative Systems (INCoS), 2013 5th International Conference on*, pages 323–328. IEEE, 2013.

[14] Keita Emura, Atsuko Miyaji, and Mohammad Shahriar Rahman. Private multiparty set intersection protocol in rational model. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pages 431–438. IEEE, 2013.

[15] Dongdong Zhao and Wenjian Luo. A study of the private set intersection protocol based on negative databases. In *Dependable, Autonomic and Secure Computing (DASC), 2013 IEEE 11th International Conference on*, pages 58–64. IEEE, 2013.

[16] Benny Pinkas, Thomas Schneider, and Michael Zohner. Scalable private set intersection based on ot extension. *ACM Transactions on Privacy and Security (TOPS)*, 21(2):7, 2018.

[17] Qingji Zheng and Shouhuai Xu. Verifiable delegated set intersection operations on outsourced encrypted data. In *Cloud Engineering (IC2E), 2015 IEEE International Conference on*, pages 175–184. IEEE, 2015.

[18] Mohammadhassan Ameri Ekhtiarabadi, Habib Allah Yajam, Javad Mohajeri, and Mahmoud Salmasizadeh. Verifiable identity-based mix network. In *Electrical Engineering (ICEE), 2015 23rd Iranian Conference on*, pages 406–409. IEEE, 2015.

[19] Vahid Yousefipoor, Mohammad Hassan Ameri, Javad Mohajeri, and Taraneh Eghlidos. A secure attribute based keyword search scheme against keyword guessing attack. In *Telecommunications (IST), 2016 8th International Symposium on*, pages 124–128. IEEE, 2016.

[20] Mahdi Mahdavi Oliaiy, Mohammad Hassan Ameri, Javad Mohajeri, and Mohammad Reza

Aref. A verifiable delegated set intersection without pairing. In *Electrical Engineering (ICEE), 2017 Iranian Conference on*, pages 2047–2051. IEEE, 2017.

[21] Shuo Qiu, Jiqiang Liu, Yanfeng Shi, Ming Li, and Wei Wang. Identity-based private matching over outsourced encrypted datasets. *IEEE Transactions on Cloud Computing*, 2015.

[22] Ran Canetti, Omer Paneth, Dimitrios Papadopoulos, and Nikos Triandopoulos. Verifiable set operations over outsourced databases. In *International Workshop on Public Key Cryptography*, pages 113–130. Springer, 2014.

[23] En Zhang, Fenghua Li, Ben Niu, and Yanchao Wang. Server-aided private set intersection based on reputation. *Information Sciences*, 387:180–194, 2017.

[24] Florian Kerschbaum. Outsourced private set intersection using homomorphic encryption. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pages 85–86. ACM, 2012.

[25] Florian Kerschbaum. Collusion-resistant outsourcing of private set intersection. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 1451–1456. ACM, 2012.

[26] Fang Liu, Wee Keong Ng, Wei Zhang, Shuguo Han, et al. Encrypted set intersection protocol for outsourced datasets. In *Cloud Engineering (IC2E), 2014 IEEE International Conference on*, pages 135–140. IEEE, 2014.

[27] Aydin Abadi, Sotirios Terzis, and Changyu Dong. O-psi: delegated private set intersection on outsourced datasets. In *IFIP International Information Security Conference*, pages 3–17. Springer, 2015.

[28] Aydin Abadi, Sotirios Terzis, Roberto Metere, and Changyu Dong. Efficient delegated private set intersection on outsourced private datasets. *IEEE Transactions on Dependable and Secure Computing*, 2017.

[29] Aydin Abadi, Sotirios Terzis, and Changyu Dong. Vd-psi: Verifiable delegated private set intersection on outsourced private datasets. In *International Conference on Financial Cryptography and Data Security*, pages 149–168. Springer, 2016.

[30] Xiaoyuan Yang, Xiaoshuang Luo, Xu An Wang, and Shuaiwei Zhang. Improved outsourced private set intersection protocol based on polynomial interpolation. *Concurrency and Computation: Practice and Experience*, 30(1):e4329, 2018.

[31] Seny Kamara, Payman Mohassel, Mariana Raykova, and Saeed Sadeghian. Scaling private set intersection to billion-element sets. In *International Conference on Financial Cryptography and Data Security*, pages 195–215. Springer, 2014.

[32] Mohammad Hassan Ameri, Mahshid Delavar, Javad Mohajeri, and Mahmoud Salmasizadeh. A key-policy attribute-based temporary keyword search scheme for secure cloud storage. *IEEE Transactions on Cloud Computing*, 2018.

[33] Sergei V Fedorenko and Peter V Trifonov. Finding roots of polynomials over finite fields. *IEEE Transactions on communications*, 50(11):1709–1711, 2002.

[34] S Dov Gordon, Jonathan Katz, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Multi-client verifiable computation with stronger security guarantees. In *Theory of Cryptography Conference*, pages 144–168. Springer, 2015.