Private and Efficient Set Intersection Protocol for Big Data Analytics

Zakaria Gheid, Yacine Challal

▶ **To cite this version:**

Zakaria Gheid, Yacine Challal. Private and Efficient Set Intersection Protocol for Big Data Analytics. 17th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP-2017), 2017, Helsinki, Finland. pp.149-164. hal-01590522

HAL Id: hal-01590522
https://hal.science/hal-01590522

Submitted on 19 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Private and Efficient Set Intersection Protocol For Big Data Analytics

Zakaria Gheid[1] and Yacaine Challal[1,2,3]

[1] Ecole nationale supérieure d'informatique,
Laboratoire des Méthodes de Conception des Systèmes, Algiers, Algeria
[2] Centre de Recherche sur l'Information Scientifique et Technique, Algiers, Algeria
[3] Université de Technologie de Compiègne,
Heudiasyc UMR CNRS 7253, Compiègne Cedex, France
{z_gheid,y_challal}@esi.dz

Abstract. Private Set Intersection (PSI) is a fundamental multi-party computation primitive used to secure many political, commercial, and social applications. PSI allows mistrustful parties to compute the intersection of their private sets without leaking additional information. PSI protocols has been largely proposed for both the semi-honest and the malicious settings. Nevertheless, the semi-honest setting does not suffice in many realistic scenarios and security in the malicious setting is built upon cryptographic schemes, which require hard assumptions and induce a high computational cost. In this work, we propose a novel two-party PSI protocol secure under the mixed model, where the server may be semi-honest and the client may be malicious. We build our protocol upon matrix algebra without using any cryptographic schemes or non-standard assumptions and we provide simulation-based security proof. Our protocol achieves a linear asymptotic complexity $O(k_v + k_c)$ for communications and server computations, where k_v and k_c are sizes of participating parties' sets. Besides, we compare empirical performance of our solution to the insecure hashing solution used in practice. Experimental results reveal efficiency and scalability of our new PSI protocol, which makes it adequate for Big Data analytics.

Keywords: Multi-Party Computation, Private Set Intersection, Big Data Analytics

1 Introduction

Private set intersection (PSI) protocol allows a client and a server to jointly compute the intersection of their private input sets without leaking any additional information. The client should learn the intersection and the server should learn nothing. PSI functionality is a core building-block for a variety of privacy-preserving Big Data applications such as relationship path discovery in social networks [1], online recommendation systems [2], medical studies of human genomes [3], suspects detection by government agencies [4] and other applications [5, 6].

As a very active research field, PSI problem has been largely studied and several protocols have been proposed for both the semi-honest setting (meaning that the adversary follows the protocol specifications) and the malicious setting (meaning that the adversary may deviate arbitrarily from the protocol) [7, 6]. However, there has been a poor adoption of PSI protocols in real applications due to several reasons [7]. The semi-honest assumption provides weak security guarantees and does not suffice for many realistic scenarios [8]. On the other hand, PSI protocols that are secure against malicious adversaries achieve an excessive overhead in communication and computation costs [7]. This is mainly due to the strong assumptions that require to be satisfied.

Recently, there has been a great interest in improving the efficiency of malicious-resistant PSI protocols [9, 6, 10]. Several works [11, 5] have achieved linear communication and computation complexities under non-standard security models as the Oracle [12] and the Common Reference String [13] models. Other works [6] proposed efficient PSI protocols with linear cost under the standard model, but, relying heavily on strong security assumptions and Homomorphic public key cryptosystems [14]. Despite an efficient asymptotic complexity, homomorphic encryption induces a high computational overhead and so, does not meet Big Data requirements [15].

Our Contribution. Facing the need for more efficient PSI schemes under realistic security models, we propose a novel PSI protocol that is secure under the mixed model of adversaries [16], where the client may be corrupted by a malicious adversary and the server may be corrupted by a semi-honest adversary. This model is more realistic than semi-honest one, since server entities are mostly governed by laws requiring data privacy and security. We give a simulation-based security proof for both cases where the client is corrupted and where the server is corrupted. We build our protocol only upon matrix algebra without any cryptographic scheme to cope with Big Data requirements in terms of computation efficiency. We achieve a linear complexity in communication and server computation costs and we confirm the efficiency of our protocol through experimental evaluations.

In Section 2, we provide a short survey on literature works in PSI field and we discuss them. Section 3 is devoted to preliminaries and standard notations used throughout this work. In Section 4, we present our methodology to build a novel PSI protocol and we describe its design. In Section 5, we give a simulation-based security proof using Real/Ideal paradigm [17]. In section 6, we analyse asymptotic complexities of communications and computations involved in our protocol. Then, we make experimental performance evaluations in Section 7 and we end up this work by a final conclusion.

2 Related Work

Private set intersection (PSI) problem is a fundamental functionality that has been largely studied due to its important applications. In this section, we review

important PSI protocols working in the standard (plain) model, where security is based only on complexity assumptions.

Assume a client (C) and a server (V) having private sets X and Y of sizes k_c and k_v respectively. Two main approaches were used to solve PSI(X,Y), namely Oblivious Polynomial Evaluation (OPE) [18] and Oblivious Pseudo-Random Functions (OPRF) evaluation [19].

OPE based-PSI. In this approach, C defines a polynomial $P(.)$ such that $P(x) = 0$ for each $x \in X$, and sends to V homomorphic encryptions of the coefficients of $P(.)$. Then, V computes the encryption of $(r.P(y) + y)$ for each $y \in Y$, using homomorphic properties of the encryption system and a fresh randomness r. Finally, C decrypts the received cyphertexts and gets either elements of the intersection (if plaintexts match an element of X) or random values. Following this approach, Freedman et al. [2] proposed two PSI protocols for semi-honest and mixed models using balanced allocations. They incur linear communications and linear client computations besides quadratic server computations that can be reduced to $O(k_c + k_v \, log \, log \, k_c)$. Kissner and Song [20] proposed OPE-based protocols that can involve more than two parties in semi-honest and malicious settings. The former has a linear communication cost, but, incurs $O(k_v.k_c)$ computations. In the latter, authors rely on expensive generic zero knowledge proofs to achieve correctness [5, 10]. Later, Dachman-Soled et al. [21] proposed a PSI protocol based on [20]. They employ a secret sharing of the polynomial inputs and avoid the use of generic zero knowledge proofs. Their construction incurs communication of $O(k_v.k^2 \, log^2 k_c + k.k_c)$ and computation of $O(k_v.k_c.k \, log k_c + k_v.k^2 log^2 k_c)$, where k is a security parameter. Recently, [6] proposed a more efficient OPE-based PSI protocol for malicious settings. Their construction incurs $O(k_v + k_c)$ communications and implies the computation of $O(k_c + k_v \, log \, log \, k_c)$ under the strong Decisional Diffie-Hellman assumption (strong-DDH). They introduce hashing technique into their construction and implies the computation of $O(k_c + k_v \, log \, k_c)$ under the DDH assumption.

OPRF-based PSI. They rely on a secure computation of a pseudo random function (PRF) $f_k(x)$ on key (k) introduced by the server (V) and input (x) introduced by the client (C), such that C should only learn $f_k(x)$, whereas V should not learn x. PSI functionality was implemented using OPRF as follows: V defines a random key (k) for a PRF $f_k(.)$ and computes the set $f_{ky} = \{f_k(y) : y \in Y\}$. Then, V and C executes an OPRF protocol where V inputs $f_k(.)$ and C inputs the set X and gets the set $f_{kx} = \{f_k(x) : x \in X\}$. At the end, V sends the set f_{ky} to C that evaluates $f_{kx} \cap f_{ky}$. OPRF was used by Hazay and Lindell [22] to develop efficient PSI protocols secure against adversaries that are more realistic than semi-honest ones. They proposed a maliciously-secure protocol with simulation-proof for one corruption case (the client only). Besides, they proposed another PSI protocol secure under the covert model of adversaries, which is a non-standard model between semi-honest and malicious [21]. Their protocols incur $O(k_v.p(n) + k_c)$ communications and computations, where elements of the

sets are taken from $\{0,1\}^{p(n)}$. Jarecki and Liu [11] improved the protocol of [22] to propose a more efficient PSI protocol secure in the presence of both malicious parties in the Common Reference String Model [13], where it is assumed that all parties have access to a common string chosen from a predetermined distribution. Their protocol incurs $O(k_v + k_c)$ communication and computation costs, but, its security proof runs an exhaustive search on the input domain of the PRF, which implies that the complexity of the simulator will grow with the input domain of the PRF [6]. In addition, their construction requires a trusted third party for a safe RSA generation [5]. Later, Hazay and Nissim [9] proposed a PSI protocol secure in the setup of malicious parties and based on standard cryptographic assumptions (without requiring a trusted third party). Authors improved the work of [2] by introducing a zero-knowledge proof that C uses to prevent it from deviating from the protocol and a technique based on a hiding commitment scheme with an OPRF evaluation protocol to prevent deviation of V from the protocol. Their protocol incurs $O(k_v + k_c)$ communications and $O(k_c + k_v \, log \, log \, k_c)$ computations. Moreover, Their construction is fairly complicated and uses both OPE and OPRF approaches [6]. Recently, [6] introduced more efficient OPRF-based PSI protocols with $O(k_v + k_c)$ costs under the strong-DDH assumption and $O((k_v + k_c) \, log \, (k_v + k_c))$ communication and computation costs under the DDH assumption.

In this work, we propose a novel PSI protocol approach based on matrix representation of the private sets. We use efficient matrix algebra without any cryptographic operations and we provide security under the mixed model of adversaries. Our protocol incurs $O(k_v + k_c)$ communication and server computation costs while maintaining fairness. We provide a detailed discussion about efficiency of our work in Section 6.2 and Section 7.2.

3 Preliminaries

In this section, we present preliminaries and standard security notations used in this work. More specific notations will be described later.

3.1 Private Set Intersection

In what follows, we give a formal definition of the Private Set Intersection (PSI) computation. Let C and V denote a client and a server. A private set intersection (PSI) scheme is a two-party computation protocol between C and V, where C holds a set of private inputs of size k_c, drawn from some domain of size n, and V holds a set of private inputs of size k_v drawn from the same domain. At the end of the protocol, C should learn which specific inputs are shared by both C and V, whereas, V should learn nothing. Let $X = \{x_1, ..., x_{kc}\}$ and $Y = \{y_1, ..., y_{kv}\}$ denote respectively C's and V's sets of inputs, then, C learns $PSI(X, Y) = X \cap Y \longmapsto \{x_i \mid \exists j : x_i = y_j\}$. This is a branch of multi-party computation (MPC) problems [23].

3.2 Multi-Party Computation

Let us consider a set of participants that want to jointly compute the value of a public function f relying on their private data. Let P_1,...,P_n denote the participants and v_1,...,v_n their private data respectively. We call Multi-Party Computation (MPC) model the running process of $f(v_1, ..., v_n)$. Let Π denote a multi-party protocol executed by n participants (P_1,...,P_n) in order to evaluate the function f. Let v denote the set of inputs $(v_1, ..., v_n)$ and sec denote the set of security parameters.

Notation 1 *Let $view_E^{\Pi}(w,sec)_i$ denote the set of messages received by the party $P_{i\in\{1,...,n\}}$ along with its inputs and outputs during the execution E of Π on the set of inputs w and security parameters sec.*

Notation 2 *Let $out_E^{\Pi}(w,sec)_i$ denote the output of the party $P_{i\in\{1,...,n\}}$ by the execution E of the protocol Π on the set of inputs w and security parameters sec. Let $out_E^{\Pi}(v,sec)$ denote the global output of all collaborating parties from the same execution of Π, where*

$$out_E^{\Pi}(w, sec) = \cup_{i=1}^{n} out_E^{\Pi}(w, sec)_i \tag{1}$$

In next section, we introduce a novel MPC protocol for private set intersection. Later, we will use these MPC notations to prove the security of our proposal.

3.3 Privacy Threat Model

In MPC protocols, the possible security threat raising from a corrupted party that participates to the execution of the protocol can be classified according to the corrupting adversary's model

Passive model (semi-honest) In this model, the corrupted parties are supposed following the protocol's specifications, yet they are allowed to analyse all information gathered during the execution of the protocol.

Active model (malicious) In this model, the corrupted parties may randomly deviate from the protocol specifications. The two common behaviors in such a model are a) aborting the protocol untimely or b) injecting fake inputs.

Mixed model This is an extension of the above assumed behavioral models, in which the adversary can either corrupt some parties actively, and other parties passively. Thus, allowing each party to behave according to its corruption model (active or passive).

4 A Novel Private And Efficient Set Intersection Protocol

In this section, we present our novel private set intersection protocol and we describe its design.

4.1 Overview and motivation

The hash-based PSI solutions use a commutative one-way hash function to encrypt all items [24]. Each party encrypts its items with its own key, then, each set is passed to the other party to be encrypted. Since, encryption is commutative, encrypted values will be equal if and only if the original values were the same. This hash-based scheme is very efficient, but, provides weak security guarantees if the input domain is not large or does not have a high entropy [7]. One party could run a brute force attack by applying all items that can be in the input domain to the hash function and compare the results to the received hashes.

In this work, we introduce a novel matrix model that is secure against such brute force attacks. To do this, we represent the private input sets as row-matrices (each matrix corresponds to a private set and each row within it corresponds to an element in the set). Then, each party obfuscates its matrix by performing a product with a random matrix chosen independently from the input domain. Next, each party sends its resultant matrix to the other party to be multiplied by the other random matrix. Since, matrix product is not commutative, which is required for the correctness of the scheme, the two parties will interchange the side of the matrix product (left multiplication and right multiplication). At the end, the two resultant matrices will be checked for rows equality as each row corresponds to an original element in the set.

4.2 Notational Conventions

In this work, we use a special typographical style to denote matrix tools that we use to build our proposal. The used notational conventions are as follows.

- We represent matrices by capital letters in bold with one-underline, e.g. $\underline{\mathbf{M}}$.
- The set of all m-by-n matrices is denoted $\mathbb{M}(m, n)$.
- Elements of matrix are indexed between brackets, e.g. $\underline{\mathbf{M}}[2, 5]$. An asterisk is used to refer to a whole row or column, e.g. $\underline{\mathbf{M}}[1, *]$.
- The multiplication operator between two matrices is denoted $\otimes$.

4.3 Protocol Design

To introduce our novel private set intersection protocol (Π-SI), we consider a client denoted C and a server denoted V having respectively $X = \{x_1, ..., x_k\}$ and $Y = \{y_1, ..., y_k\}$ sets of private data and want to securely get the intersection between their sets (Section 3.1). Assume for $1 \leq i \leq k$ and $1 \leq j \leq k$: x_i and $y_j \in \mathbb{R}^n$. Let $\underline{\mathbf{M1}}$ and $\underline{\mathbf{M2}}$ denote random invertible matrices used by C and V respectively to obfuscate their sets, where $\underline{\mathbf{M1}} \in \mathbb{M}(k, k)$ and $\underline{\mathbf{M2}} \in \mathbb{M}(n, n)$. Let $\underline{\mathbf{MX}}$ and $\underline{\mathbf{MY}}$ denote the private sets X and Y respectively, represented as row matrices, where $\underline{\mathbf{MX}} \in \mathbb{M}(k, n)$ and $\underline{\mathbf{MY}} \in \mathbb{M}(k, n)$. Without loss of generality, we consider the case where the sets X and Y have the same size (k) and we present the detail of Π-SI protocol in Algorithm1. Later, we describe the more general case.

ALGORITHM 1 (Π-SI): A Private and Efficient Set Intersection Protocol

INPUT. *C*'s input is a set X = {x1, . . ., xk}, *V*'s input is a set Y = {y1, . . ., yk }. The elements in the input sets are taken from a domain of size n.

REQUIRE. 0 < k < n (where k is the size of the sets)

PRE-PROCESSING.

a. "*C*" Creates $\underline{MX} \in \mathbb{M}$ (k,n) with *X*'s elements put as rows
b. "*V*" Creates $\underline{MY} \in \mathbb{M}$ (k,n) with *Y*'s elements put as rows

STEP1. "*C*" performs the following

1. Generates a random invertible $\underline{M1} \in \mathbb{M}$ (k,k)
2. Computes $\underline{M1X} = \underline{M1} \otimes \underline{MX}$
3. Sends $\underline{M1X}$ to "*V*"

$\underline{M1X}$ ⟶

STEP2. "*V*" performs the following

4. Generates a random invertible $\underline{M2} \in \mathbb{M}$ (n,n)
5. Computes $\underline{M1X2} = \underline{M1X} \otimes \underline{M2}$
6. Computes $\underline{MY2} = \underline{MY} \otimes \underline{M2}$
7. Sends $\underline{M1X2}$ and $\underline{MY2}$ to "*C*"

⟵ $\underline{M1X2}$, $\underline{MY2}$

STEP3. "*C*" performs the following

8. Computes $\underline{M1Y2} = \underline{M1} \otimes \underline{MY2}$
9. Compares $\underline{M1X2}$[i,*] and $\underline{M1Y2}$[j,*] such that for any (1≤i≤k) and (1≤j≤k), if $\underline{M1X2}$[i,*]= $\underline{M1Y2}$[j,*], then, $\underline{MX}$[i,*]= $\underline{MY}$[j,*]. Therefore, adds the row $\underline{MX}$[i,*] to the set Ψ (defined as an empty set).

OUTPUT. "*C*" returns Ψ that contains all items of the intersection

4.4 Generalization

In a more general case, we consider the client (C) having k_c elements and the server (V) having k_v elements. Then, V can simply creates several matrices $\underline{\mathbf{MY}_i} \in \mathbb{M}(k, n)$, where $i > 1$ (Instruction b. Algorithm1) and distributes its k_v elements on them. Thus, V will repeat instruction 6 for each $\underline{\mathbf{MY}_i}$ and will send $\bigcup_{i>1}\underline{\mathbf{MY}_i}$ instead of $\underline{\mathbf{MY}}$ during instruction 7. At the reception, C will perform instruction 8 and instruction 9 for each received $\underline{\mathbf{MY}_i}$.

5 Security Analysis

In this section, we present a simulation-based security proof for our protocol using the Real/Ideal model [17].

5.1 Real/Ideal Model

Let Π denote a multi-party protocol executed by m participants (P_1,...,P_m) in order to evaluate a function f. Let B denote the class of adversary that may corrupt participants in Π. Let R and D denote respectively the real and the ideal executions of Π on the set of inputs w and the set of security parameters sec.

During a **real execution** (R) we consider the presence of an adversary denoted A that behaves according to the class B while corrupting a set of participants $P_{i(1\leq i\leq m)}$. At the end of R, uncorrupted parties output whatever was specified in Π and the corrupted P_i outputs any random functions of their $view_R^{\Pi}(w,sec)_i$.

During an **ideal execution** (D) we consider the presence of a trusted incorruptible party denoted T, which receives the set of inputs w from all participants in order to evaluate the function f in the presence of an adversary denoted S. We assume S corrupts the same P_i as the correspondent adversary A of real execution, and behaves according to the same class B before sending inputs to T. By the end of D, uncorrupted participants output what was received from T and the corrupted P_i output any random functions of their $view_D^{\Pi}(w,sec)_i$.

Definition 1. *Let Π and f be as above. We consider Π a secure multi-party protocol if for any real adversary A having a class B and attacks the protocol Π during its execution on the set of inputs w and the set of security parameters sec, there exists an adversary S in the ideal execution having the same class B and that can emulate any effect achieved by A. Let $\stackrel{d}{\equiv}$ denote the distribution equality. We formalize the definition of a secure multi-party protocol Π as follows*

$$\{out_R^{\Pi}(w, sec)\} \stackrel{d}{\equiv} \{out_D^{\Pi}(w, sec)\} \tag{2}$$

5.2 Security Proof

In what follows, we give security simulations of Π-SI protocol using Real/Ideal paradigm. The allowed behavioural class of adversary is the mixed one (Section 3.3), where the client (C) is actively corrupted and the server (V) is passively corrupted.

Let A, S and T denote respectively a real adversary, an ideal adversary and a trusted third party, where A and S have the same class. Let Π denote the Π-SI protocol (Algorithm1), w denote the set of inputs $\{\underline{\mathbf{MX}}, \underline{\mathbf{MY}}\}$, sec denote security parameters that will be presented below and $PSI(X, Y)$ denote the private set intersection between X and Y, which are the private sets of C and

V respectively. For simplicity, we give a simulation for the specific case where the sets X and Y have the same size (k). Next, we show how to generalize the proof.

Theorem 1. *Given a set of security parameters (sec) defined as* $sec = \{(n,k) \in \mathbb{N}^2 : 0 < k < n\}$. *Under these conditions, the protocol* Π*-SI defined in Algorithm1 is a secure multi-party protocol against an active corruption of* C.

Proof. Assume C is actively corrupted by A. Then, it can only inject fake inputs ($\underline{\mathbf{MX}}$) since aborting the protocol untimely will have no meaning. Assume C sends a fake $\underline{\mathbf{MX}}$. In this case, S can emulate A by just handling the fake $\underline{\mathbf{MX}}$ and sends it to T, which performs the required computation and sends back $PSI(X,Y)$ to C. Thereby, completing the simulation. At the end, the views of C in Ideal and Real executions will be as follows

$$view_D^{\Pi}(w, sec)_C = \{\underline{\mathbf{MX}}, PSI(X,Y)\} \tag{3}$$

$$view_R^{\Pi}(w, sec)_C = \{\underline{\mathbf{MX}}, \underline{\mathbf{M1X2}}, \underline{\mathbf{MY2}}, PSI(X,Y)\} \tag{4}$$

Otherwise, $\underline{\mathbf{M1X2}} = \underline{\mathbf{M1X}} \otimes \underline{\mathbf{M2}}$, where $\underline{\mathbf{M1X}} \in \mathbb{M}(k,n)$ and $\underline{\mathbf{M2}} \in \mathbb{M}(n,n)$. According to security parameters (sec), we have $k < n$. This, preserves well the privacy of $\underline{\mathbf{M2}}$. Thereby, $\underline{\mathbf{M1X2}}$ that contains $(k \times n)$ equations opposite to $(n \times n)$ unknowns for C, will not involve meaningful information for it and can be reduced from its view. Likewise, $\underline{\mathbf{MY2}} = \underline{\mathbf{MY}} \otimes \underline{\mathbf{M2}}$, where $\underline{\mathbf{MY}} \in \mathbb{M}(k,n)$ and $\underline{\mathbf{M2}} \in \mathbb{M}(n,n)$. Then, $\underline{\mathbf{MY2}}$ will contain $(k \times n)$ equations opposite to $((k \times n)+(n \times n))$ unknowns for C, which does not involve meaningful information for it and can be so, reduced from its view. After these reductions, the view of C in real execution could be described as follows

$$view_R^{\Pi}(w, sec)_C = \{\underline{\mathbf{MX}}, PSI(X,Y)\} \tag{5}$$

Thus, relying on (3) and (5) we get

$$\{out_R^{\Pi}(w, sec)_C\} \stackrel{d}{\equiv} \{out_D^{\Pi}(w, sec)_C\} \tag{6}$$

On the other hand, the uncorrupted V can not be affected by the corruption of C since V does not require any output in real execution. Thus, T will simply not send it any output during ideal execution. This, means that

$$\{out_R^{\Pi}(w, sec)_V\} \stackrel{d}{\equiv} \{out_D^{\Pi}(w, sec)_V\} \tag{7}$$

Through (6) and (7), we proved by simulation that all effects achieved by a real active adversary corrupting C can also be achieved in an ideal execution. Then, Π-SI is a secure multi-party protocol against active corruption of C (Definition1).

Theorem 2. *Given a set of security conditions (sec) defined as* $sec = \{(n,k) \in \mathbb{N}^2 : 0 < k < n\}$. *Under these conditions, the protocol* Π*-SI defined in Algorithm1 is a secure multi-party protocol against a passive corruption of* V.

Proof. Assume V is passively corrupted. In this case, V should follow the specification of the protocol Π-SI, yet, it is allowed to analyse all data gathered during the execution. Then, S will just handle V's input and sends it to T, which performs the required computation and sends $PSI(X,Y)$ to C while sending nothing to V. Thereby, completing the simulation. At the end, the views of V in Ideal and Real executions will be as follows

$$view_D^{\Pi}(w, sec)_V = \{\underline{\mathbf{MY}}\} \tag{8}$$

$$view_R^{\Pi}(w, sec)_V = \{\underline{\mathbf{MY}}, \underline{\mathbf{M1X}}\} \tag{9}$$

Moreover, $\underline{\mathbf{M1X}} = \underline{\mathbf{M1}} \otimes \underline{\mathbf{MX}}$, where, $\underline{\mathbf{M1}} \in \mathbb{M}(k,k)$ and $\underline{\mathbf{MX}} \in \mathbb{M}(k,n)$. Then, since we defined $0 < k$ as security parameter (sec), we get $(k \times n) < ((k \times n) + (k \times k))$. Thus, $\underline{\mathbf{M1X}}$ that contains $(k \times n)$ equations opposite to $((k \times n) + (k \times k))$ unknowns for V, will not involve meaningful information for it and can be so, reduced from its view. After reduction, we obtain

$$view_R^{\Pi}(w, sec)_V = \{\underline{\mathbf{MY}}\} \tag{10}$$

Thus, relying on (8) and (10) we get

$$\{out_R^{\Pi}(w, sec)_V\} \stackrel{d}{\equiv} \{out_D^{\Pi}(w, sec)_V\} \tag{11}$$

On the other hand, the uncorrupted C outputs what was received from T in ideal execution, which is $PSI(X,Y)$ according to the simulation given above and outputs what was specified int the protocol Π-SI in real execution, which is $PSI(X,Y)$ (Algorithm1, Output section) . Then, we have

$$\{out_R^{\Pi}(w, sec)_C\} \stackrel{d}{\equiv} \{out_D^{\Pi}(w, sec)_C\} \tag{12}$$

Through (11) and (12) we proved by simulation that all effects achieved by a real passive adversary corrupting V can also be achieved in an ideal execution. Then, Π-SI is a secure multi-party protocol against passive corruption of V (Definition1).

Corollary 1. *Given a set of security conditions (sec) defined as* $sec = \{(n,k) \in \mathbb{N}^2 : 0 < k < n\}$. *Under these conditions, the protocol Π-SI defined in Algorithm1 is a secure multi-party protocol in the mixed model of adversary, where C is actively corrupted and V is passively corrupted.*

Proof. Corollary1 relies heavily on the Theorem1 and Theorem2 proved above, while considering separately the case when the client (C) is corrupted and the case when the server (V) is corrupted. We assume that if both parties are corrupted we are not required to provide security guarantees.

Note 1. To generalize the proof (Section 4.4), assume the client (C) has k_c elements and the server (V) has k_v elements. This, will affect the view of C in real execution when it is corrupted (Equation (4)), which will be defined as follows

$$view_R^{\Pi}(w, sec)_C = \{\underline{\mathbf{MX}}, \underline{\mathbf{M1X2}}, \bigcup_{i>1} \underline{\mathbf{MY}_i\mathbf{2}}, PSI(X,Y)\} \tag{13}$$

Where $\underline{\mathbf{MY}_i\mathbf{2}} = \underline{\mathbf{MY}_i} \otimes \underline{\mathbf{M2}}$ for $i > 1$. According to security parameters (sec), $\underline{\mathbf{M2}}$ is unknown for C (Proof Theorem1), then, each $\underline{\mathbf{MY}_i}$ remains private and does not involve meaningful information for C and can be so, reduced from its view. Likewise, $\underline{\mathbf{M1X2}}$ can be reduced from the view of C in Real execution (Proof Theorem1). Thus, Theorem1 remains valid in the general case. On the other hand, the the views of V when it is corrupted will be augmented with $\bigcup_{i>1}\underline{\mathbf{MY}_i}$ instead of $\underline{\mathbf{MY}}$. However, this will affect the views of V in both ideal and real executions. Thus, Theorem2 remains valid in the general case.

6 Complexity Analysis

In this section, we analyse asymptotic complexities of communications and computations involved in our protocol (Π-SI: Algorithm1, Section 4.3). We make comparison with recent efficient protocols and we highlight our improvements.

6.1 Analysis

Let C and V denote a client and a server and k_c and k_v denote respectively the number of elements in their sets (General case, Section 4.4), where each element is assumed to be in $\mathbb{R}^n$. As off-line operations do not affect significantly the running time, we do not consider complexities of random matrices generation (Algorithm1: Pre-processing).

In step 1, getting $\underline{\mathbf{M1X}}$ requires $O(k_c^2 n)$ computations and sending $\underline{\mathbf{M1X}}$ costs $O(k_c n)$. In step 2, the server (V) performs $O(k_c n^2)$ operations to get $\underline{\mathbf{M1X2}}$, then, it performs at most $((k_v/k_c)+1)O(k_c n^2)$ to get the set $\bigcup_{i>1}\underline{\mathbf{MY}_i\mathbf{2}}$. This, results in $O((k_v + k_c)n^2)$. Moreover, sending $\underline{\mathbf{M1X2}}$ costs $O(k_c n)$ and sending $\bigcup_{i>1}\underline{\mathbf{MY}_i\mathbf{2}}$ is bounded by $O((k_v + k_c)n)$. In step 3, C should compute $\underline{\mathbf{M1Y}_i\mathbf{2}}$ for each $\underline{\mathbf{MY}_i\mathbf{2}}$ received ($i > 1$), which requires at most $O((k_v.k_c)n)$.

As a the length of elements (n) is assumed to be fixed for each application, we can reduce the complexities formulas to get a communication cost of $O(k_v+k_c)$, a server computation cost of $O(k_v+k_c)$ and a client computation cost of $O(k_v.k_c)$.

6.2 Discussion

In communication cost, we have achieved a linear complexity of $O(k_v + k_c)$. As far as we know, this is the most efficient complexity achieved by protocols working under standard assumptions and secure against malicious clients. The client computation cost is quadratic, bounded by $O(k_v.k_c)$, which is the minimum required for the native set intersection verification. Moreover, our protocol brings a significant improvement on the server side computations, costing a linear complexity of $O(k_v + k_c)$ without requiring any hard or non-standard assumption. This efficient cost will ensure the scalability of our protocol for multi-client contexts. To the best of our knowledge, the only standard set intersection protocol, which is secure against malicious client and that reached $O(k_v + k_c)$ computations on the server side without requiring non-standard assumption is [6]. The

latter protocol is proven to be secure under the strong Decisional Diffie-Hellman assumption. In contrast, our protocol does not require any cryptographic assumption, which makes it more practical. In Table1, we summarize complexities of some representative state-of-the art PSI protocols under the standard model.

Table 1: Complexity analysis of state-of-the art PSI protocols

Reference	**Adversary Model**	**Assumption**	**Communication**	**Server Computation**	**Client Computation**
[2]	Semi-honest	Homomorphic Encryption	$O(k_v+k_c)$	$O(k_c + k_v \log\log k_c)$	$O(k_v+k_c)$
[20]	Malicious	Homomorphic Encryption	$O(k_v+ k_c)$	$O(k_v.k_c)$	$O(k_v.k_c)$
[9]	Malicious	Decisional Diffie-Hellman	$O(k_v+ k_c)$	$O(k_c + k_v \log\log k_c)$	$O(k_v+ k_c)$
[6]	Malicious	Decisional Diffie-Hellman	$O(k_v+ k_c)$	$O(k_c + k_v \log k_c)$	-
[6]	Malicious	d-strong Decisional Diffie-Hellman	$O(k_v+ k_c)$	$O(k_v+ k_c)$	-
[10]	Semi-honest	Homomorphic Encryption	$O(k_v+ k_c)$	$O(k_v)$	$O(k_c.k_c)$
[10]	Malicious	Homomorphic Encryption	$O(k_v + k_c \log k_v)$	$O(k_v) + O(k_v.k_c)$ Mult.	$O(k_v.k_c)$
Our Work	**Mixed**	**No Crypto-Assumption**	$\mathbf{O(k_v+ k_c)}$	$\mathbf{O(k_v+ k_c)}$	$\mathbf{O(k_v.k_c)}$

7 Empirical Evaluation

In this section, we evaluate the computation performances of our proposed Π-SI protocol and we make comparison with the efficient and insecure hashing scheme used in practice.

7.1 Experimental environment and scenarios

In order to prove the efficiency of Π-SI protocol in practical scenarios, we evaluate the computational time required by a server (V) and a client (C) while executing Π-SI protocol in a real environment. We make two experiments denoted E_1 and E_2 to simulate respectively the case of equal and unequal dataset sizes. Let k_v and k_c denote the sizes of the set of V and the set of C respectively, where each element within a set is assumed to be in $\mathbb{R}^n$. Let *mult* and *add* denote one multiplication and one addition respectively. We evaluate the computational costs involved in Π-SI protocol (Algorithm1) as follows

$$Cost_V^{(\Pi-SI)} = n^2(k_v + k_c)\ mult\ + n(n-1)(k_v + kc)\ add$$
$$Cost_C^{(\Pi-SI)} = nk_c(k_v + 2k_c)\ mult\ + n(k_c - 1)(k_v + 2k_c)\ add$$

For more realistic results, we compare the performance of our solution to the hashing PSI scheme used in practice. We chose a simple and efficient commutative hash function H, such as $H_k(x) = x^k\ mod\ p$, where k is a 32-bit security parameter and p is a 32-bit random prime. Let exp and mod denote respectively one exponentiation and one modulo. We evaluate the hashing scheme as follows

$$Cost_V^{(hashing)} = Cost_C^{(hashing)} = n(k_v + k_c)\ exp + n(k_v + k_c)\ mod$$

We make evaluations on the same elements using a custom simulator built in Python and an Intel i5-2557M CPU running at 1.70 GHz and having a 4 GB of RAM.

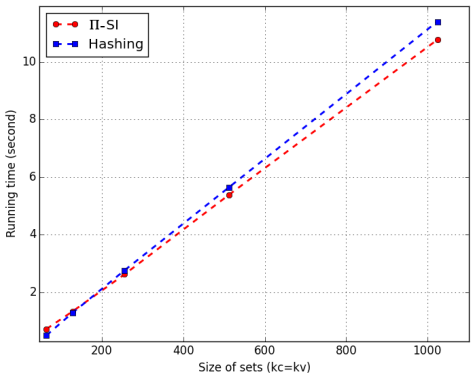
7.2 Results and discussion

Table 2: E_1. Running time of Π-SI and the insecure hash solution over equal set sizes

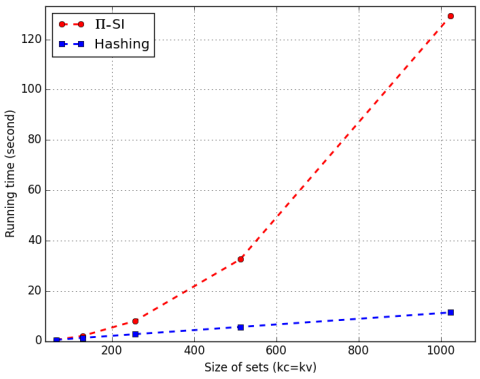
k_v (k_c=k_v)		2^6	2^7	2^8	2^9	2^{10}
Server Computation Cost (s)	**Π-SI**	0.72	1.33	2.62	5.38	**10.77**
	Hashing	0.50	1.28	2.76	5.64	11.40
Client Computation Cost (s)	**Π-SI**	0.50	2.06	8.04	32.53	129.35
	Hashing	0.50	1.28	2.76	5.64	11.40

Table 3: E_2. Running time of Π-SI and the insecure hash solution over unequal set sizes

k_v ($k_c = 2^6$)		2^6	2^7	2^8	2^9	2^{10}
Server Computation Cost (s)	**Π-SI**	0.72	1.01	1.67	3.03	**5.70**
	Hashing	0.50	0.97	1.72	3.18	6.08
Client Computation Cost (s)	**Π-SI**	0.50	0.69	1.01	1.69	**3.04**
	Hashing	0.50	0.97	1.72	3.18	6.08

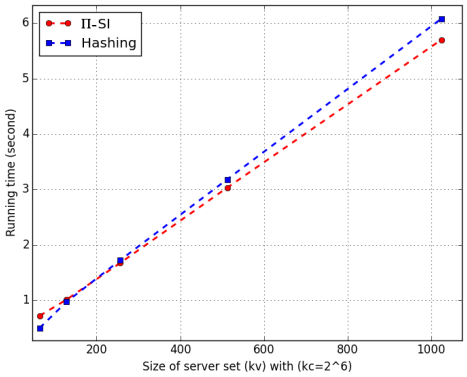


(a) Server computation cost

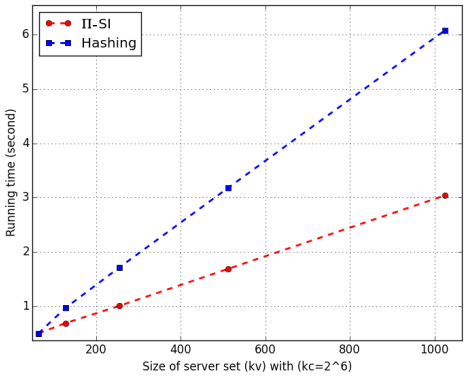


(b) Client computation cost

Fig. 1: Running time of Π-SI and the insecure hash solution over equal set sizes of $\mathbb{R}^7$ elements



(a) Server computation cost



(b) Client computation cost

Fig. 2: Running time of Π-SI and the insecure hash solution over unequal set sizes of $\mathbb{R}^7$ elements with ($k_c = 2^6$)

E_1 expriment. In E_1, we evaluated the running time of Π-SI and hashing protocols over server (V) and client (C) sets having equal sizes (k_v=k_c). We varied the size of the sets in the range $\{2^6, 2^7, 2^8, 2^9, 2^{10}\}$ of elements belonging to $\mathbb{R}^n$ (n=2^7) and we sketch results in Table2 and Figure1. Regarding server computation cost, Π-SI protocol has a short efficiency distance ($0.x$ s) lower than the hashing scheme for small sets (2^6,2^7). Then, Π-SI execution revealed a slower increasing rate than the hashing scheme, which makes its more efficient

for big sets (2^8,2^9). This efficient increasing rate presented by Π-SI is due to the use of efficient arithmetic operations (addition, multiplication) compared to the expensive operations involved in the hashing solution (modulo, exponentiation). Regarding the client computation cost, the hashing solution outperforms Π-SI with at most one order of magnitude ($\times 10$), which is very efficient compared to existing solutions [7].

E_2 expriment. In E_2, we simulated the case of unequal set sizes, which is more realistic. For this, we fixed the size of the client set to $k_c = 2^6$ elements of $\mathbb{R}^n$ (n=2^7) and we varied the size of the server sets in the range $\{2^6, 2^7, 2^8, 2^9, 2^{10}\}$. Results presented in Table3 and Figure2 reveal a very efficient level of Π-SI that outperforms the hashing solution on the server side for big sets (2^8,2^9). Regarding the client computational cost, Π-SI presented a slow increasing rate that makes it more efficient than the hashing solution. This efficiency presented by Π-SI on the client side contrary to E_1 is due to the linear dependability of the client computational cost on the server set size. These results, confirm the adequacy of Π-SI protocol to be implemented on servers having Big Data sets.

8 Conclusion

In this paper, we have proposed a novel two-party Private Set Intersection protocol named (Π-SI). We have built this protocol upon efficient matrix algebra without any cryptographic scheme to cope with Big Data sets. Through security analysis conducted with the standard real/ideal paradigm, we have proved the privacy protection ensured by (Π-SI) against a semi-honest server and a malicious client. (Π-SI) involves linear asymptotic complexities for communication and server computations, which makes it scalable for multi-client settings. Besides, across empirical evaluations performed on Big Data sets (2^{10} elements of $\mathbb{R}^7$), we have confirmed the efficiency level provided by (Π-SI) protocol compared to the insecure hashing solution used in real applications.

References

[1] Mezzour, G., Perrig, A., Gligor, V., Papadimitratos, P. In: Privacy-Preserving Relationship Path Discovery in Social Networks. Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 189–208

[2] Freedman, M.J., Nissim, K., Pinkas, B. In: Efficient Private Matching and Set Intersection. Springer Berlin Heidelberg, Berlin, Heidelberg (2004) 1–19

[3] Baldi, P., Baronio, R., De Cristofaro, E., Gasti, P., Tsudik, G.: Countering gattaca: Efficient and secure testing of fully-sequenced human genomes. In: Proceedings of the 18th ACM Conference on Computer and Communications Security. CCS '11, New York, NY, USA, ACM (2011) 691–702

[4] Fischlin, M., Pinkas, B., Sadeghi, A.R., Schneider, T., Visconti, I. In: Secure Set Intersection with Untrusted Hardware Tokens. Springer Berlin Heidelberg, Berlin, Heidelberg (2011) 1–16

[5] De Cristofaro, E., Kim, J., Tsudik, G. In: Linear-Complexity Private Set Intersection Protocols Secure in Malicious Model. Springer Berlin Heidelberg, Berlin, Heidelberg (2010) 213–231
[6] Hazay, C. In: Oblivious Polynomial Evaluation and Secure Set-Intersection from Algebraic PRFs. Springer Berlin Heidelberg, Berlin, Heidelberg (2015) 90–120
[7] Pinkas, B., Schneider, T., Zohner, M.: Scalable private set intersection based on ot extension. (2016)
[8] Aumann, Y., Lindell, Y.: Security against covert adversaries: Efficient protocols for realistic adversaries. Journal of Cryptology **23**(2) (2010) 281–343
[9] Hazay, C., Nissim, K.: Efficient set operations in the presence of malicious adversaries. Journal of Cryptology **25**(3) (2012) 383–433
[10] Hazay, C., Venkitasubramaniam, M. In: Scalable Multi-party Private Set-Intersection. Springer Berlin Heidelberg, Berlin, Heidelberg (2017) 175–203
[11] Jarecki, S., Liu, X. In: Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 577–594
[12] Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security. CCS '93, New York, NY, USA, ACM (1993) 62–73
[13] Canetti, R., Fischlin, M. In: Universally Composable Commitments. Springer Berlin Heidelberg, Berlin, Heidelberg (2001) 19–40
[14] Paillier, P. In: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. Springer Berlin Heidelberg, Berlin, Heidelberg (1999) 223–238
[15] Lu, R., Zhu, H., Liu, X., Liu, J.K., Shao, J.: Toward efficient and privacy-preserving computing in big data era. IEEE Network **28**(4) (July 2014) 46–50
[16] Ishai, Y., Kushilevitz, E., Lindell, Y., Petrank, E. In: On Combining Privacy with Guaranteed Output Delivery in Secure Multiparty Computation. Springer Berlin Heidelberg, Berlin, Heidelberg (2006) 483–500
[17] Canetti, R.: Security and composition of multiparty cryptographic protocols. Journal of CRYPTOLOGY **13**(1) (2000) 143–202
[18] Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing. STOC '99, New York, NY, USA, ACM (1999) 245–254
[19] Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O. In: Keyword Search and Oblivious Pseudorandom Functions. Springer Berlin Heidelberg, Berlin, Heidelberg (2005) 303–324
[20] Kissner, L., Song, D.: Privacy-preserving set operations. In: Proceedings of the 25th Annual International Conference on Advances in Cryptology. CRYPTO'05, Berlin, Heidelberg, Springer-Verlag (2005) 241–257
[21] Dachman-Soled, D., Malkin, T., Raykova, M., Yung, M. In: Efficient Robust Private Set Intersection. Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 125–142
[22] Hazay, C., Lindell, Y.: Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. Journal of Cryptology **23**(3) (2010) 422–456
[23] Lindell, Y., Pinkas, B.: Secure multiparty computation for privacy-preserving data mining. Journal of Privacy and Confidentiality **1**(1) (2009) 5
[24] Vaidya, J., Clifton, C.: Secure set intersection cardinality with application to association rule mining. J. Comput. Secur. **13**(4) (July 2005) 593–622