Mutual Private Set Intersection with Linear Complexity[⋆]

Myungsun Kim, Hyung Tae Lee, and Jung Hee Cheon

ISaC & Dept. of Mathematical Sciences, Seoul National University,
599 Gwanangno, Gwangak-gu, Seoul 151-747, Korea
{msunkim,htsm1138,jhcheon}@snu.ac.kr

Abstract. A private set intersection (PSI) protocol allows players to obtain the intersection of their inputs. While in its unilateral version only the client can obtain the intersection, the mutual PSI protocol enables all players to get the desired result. In this work, we construct a mutual PSI protocol that is significantly more efficient than the state-of-the-art in the computation overhead. To the best of our knowledge, our construction is the *first* result with linear computational complexity in the semi-honest model. For that, we come up with an efficient data representation technique, called *prime representation.*

Keywords: Mutual Private Set Intersection, Prime Representation

1 Introduction

The *mutual* Private Set Intersection (PSI) problem is the following: both of two players with private sets learn the intersection of their sets without releasing any other information to each other. Roughly speaking, a mutual PSI (mPSI) protocol is a secure computation protocol for the ideal functionality $\mathcal{F}_{\mathsf{mPSI}} : (\mathtt{X}_A, \mathtt{X}_B) \rightarrow (\mathtt{X}_A \cap \mathtt{X}_B, \mathtt{X}_A \cap \mathtt{X}_B)$ where $\mathtt{X}_A$ (resp., $\mathtt{X}_B$) is a private set of the player A (resp., B). This paper's main goal is to construct a secure mPSI protocol that is more efficient than existing work.

There has been much research on the PSI problem. Examples include [2, 14, 19, 16, 5, 18, 17, 10, 9, 11, 7]. In contrast to our work, however, most of prior work except for [19, 5, 11, 7] is focused on solving the *unilateral* PSI problem.[1] In this problem, two players – a server and a client – are allowed to interact on their private sets such that the client only learns the intersection of their input sets, while the server learns nothing.

According to De Cristofaro and Tsudik [10], a mutual PSI protocol can be easily obtained by two instantiations of a unilateral PSI protocol. This argument works fairly well under the semi-honest model. Given a secure mPSI protocol designed by the above approach, consider a way to transform it to one secure in the malicious model. In general, we enforce a malicious player to behavior

[⋆] This paper is to be (was) presented at WISA 2011.

[1] In [10], the authors call this problem a one-way version of the PSI problem.

as a semi-honest player using zero-knowledge proof techniques. In this model, however, since there is no way to prevent a player from prematurely suspending the execution one of users always can abort when he obtains the intersection and before his counterpart obtains the intersection [15]. Therefore, this approach has a principal limitation to get an mPSI protocol with better security. This is the main reason why we choose not to pursue the direction.

This work may find applications in real-life business requiring enhanced privacy. For example, our work is useful in the relationship-graph example in social networks of Mezzour et al. [21]. A social relationship can correspond to a private personal real-world relationship. Further, the relationship paths are often used for access control mechanisms: nearby people deserve a higher level of trust. Thus, the discovery of relationship paths may be maliciously used in the large-scale targeting and monitoring of multiple individuals in real life based on exposed relationship paths. Mezzour et al. [21] present techniques to protect the privacy of relationship paths in a social network by means of private set intersection.

1.1 Our Contributions

We begin with the novel work by Kissner and Song [19]. Their set-intersection protocol incurs $\mathcal{O}(k^2)$ computation but linear communication overhead where k is the cardinality of each private set. Recall that our goal is to construct a secure and efficient mPSI scheme, more specifically, an mPSI scheme with *linear* computational complexity in the semi-honest model.

Contributions of our work include:

- We present a new representation technique. We call it *Prime Representation.* In contrast to prior work, we represent each element in a private set as a prime number in $\mathbb{Z}^+$. This technique enables to significantly improve the computation complexity.
- We construct an mPSI protocol more efficient than prior work in the computation overhead. To the best of our knowledge, our mPSI protocol is the *first* result with linear computational complexity in the semi-honest model.

1.2 Related Work

Mutual PSI Protocols Kissner and Song [19] propose an mPSI protocol using oblivious polynomial evaluation (OPE) technique [22]. This protocol is secure in the semi-honest and also malicious model with quadratic computation complexity in the cardinality of set. As mentioned before, they use zero-knowledge proofs (ZKP) to prevent players from deviating the protocol. Later, Dachman-Soled et al. [11] propose an improved construction using Shamir's secret sharing instead of ZKPs. Complexity of their work amounts to $\mathcal{O}(k^2 \log k + k \log^2 k)$ in computation. Camenisch and Zaverucha [5] propose an mPSI protocol for certified sets. Their protocol also builds on OPE and achieves quadratic computation overhead. Finally, Cheon et al. [7] improve efficiency of Kissner and Song's protocol using the fast Fourier transformation. Complexity of this protocol amounts to sub-quadratic ($\mathcal{O}(k \log^2 k)$) in computation, which is not still linear.

Unilateral PSI Protocols Freedman et al. [14] introduce the PSI problem and first present protocols based on OPE. The construction in the semi-honest model incurs quadratic computational complexity. But, the number of modular exponentiations can be reduced to $\mathcal{O}(k \log \log k)$ exponentiations for server and $\mathcal{O}(k)$ exponentiations for client. Later, Hazay and Lindell [16] propose one solution using oblivious pseudorandom functions (OPRF). This protocol has been later improved by Jarecki and Liu [18]. The latter incurs the linear computational complexity for each server and client. More recently, Hazay and Nissm [17] improve Freedman et al.'s construction by combining with OPRF. Another family of unilateral PSI protocols utilize blind-RSA signatures [6]. De Cristofaro and Tsudik [10] present unilateral PSI protocols with linear complexity. De Cristofaro et al. [9] provide a unilateral PSI protocol secure in malicious setting with the same complexity.

Organization The remainder of this paper is organized as follows. In the next section, we briefly introduce the security model and the cryptographic tool. Section 3 provides a full explanation of our representation technique. We present our construction in Section 4 along with analysis. Finally, in Section 5 we discuss how to convert our semi-honest protocol to a malicious one.

2 Preliminaries

In this section, we present our cryptographic tools and security model.

2.1 Additive Homomorphic Encryption

Our construction requires a semantically-secure public-key encryption scheme that holds the group homomorphism of addition and multiplication by a constant. Let $\mathcal{E}_{pk}(\cdot)$ denote the encryption with a key pair (pk, sk). Precisely speaking, an additive homomorphic encryption scheme $\mathcal{E}_{pk}$ supports the following operations that can be performed without knowledge of the private key: (1) Given two encryptions $\mathcal{E}_{pk}(m_1)$ and $\mathcal{E}_{pk}(m_2)$, we can efficiently compute the encryption of $(m_1 + m_2)$, denoted by $\mathcal{E}_{pk}(m_1 + m_2) = \mathcal{E}_{pk}(m_1) +_{\mathsf{h}} \mathcal{E}_{pk}(m_2)$, (2) Given some constant c and an encryption $\mathcal{E}_{pk}(m)$, we can also efficiently obtain $\mathcal{E}_{pk}(cm) = c *_{\mathsf{h}} \mathcal{E}_{pk}(m)$. This property is satisfied by the Paillier encryption [23] or the ElGamal encryption [13], but our protocol utilizes the Paillier encryption.

Remark 1 *One can say the ElGamal encryption [13] can be applied to get the same property. In fact, when an encoding to an individual element a in a set is defined as $a \mapsto g^a$ where g is a generator of a cyclic group, one can use the ElGamal encryption and can enjoy the homomorphic properties under addition as well. However, one can easily see that the ElGamal encryption does not provide the efficient decryption.*

Threshold Decryption Working from Shoup's threshold version of RSA in [24], Damgård and Jurik propose in [12] a threshold version of Paillier's encryption scheme. Threshold encryption requires a pre-determined number of players to collaborate on fully decrypting a message. Any collaboration between fewer than the specified number of contributors does not result in a complete decryption.

2.2 Security Model

We mainly consider the semi-honest model rather than the malicious model. Of course, our final goal is to construct an mPSI secure against malicious adversaries; but sometimes it is not easy to directly obtain the desired result and so we first construct an mPSI protocol secure in the semi-honest model. We then convert this to an construction secure in the malicious model using ZKPs.

In the semi-honest model, all players behavior according to the protocol specification. Security in this model is straightforward: (1) Correctness. an mPSI protocol is *correct* if at the conclusion of execution all of two players output the exact intersection (possible empty) of their respective sets. (2) Privacy. an mPSI protocol is *private* if no players learn information about the subset elements on each player that are not in the intersection of their respective sets.

3 Prime Representation

In this section, we describe the basic intuition of our data representation technique, which is called *prime representation.*

We begin with explaining the rationale behind prime representation. Roughly speaking, our idea is to represent the individual elements of a set as prime numbers. As mentioned above, in Freedman et al.'s protocol [14] a set is represented as a polynomial and the elements of the set is as its roots. That is, a player represents elements in his private set, $\mathtt{X} = \{a_1, \ldots, a_k\}$, as the roots of a k-degree polynomial on a ring $\mathbf{R}$, $f(x) = \prod_{i=1}^{k}(x - a_i) \in \mathbf{R}[x]$. Most of mPSI protocols [19, 5, 7] follow this idea.

Our basic observations are:

- Each linear term of the polynomial $f(x)$ is irreducible in the ring of polynomials.
- The fundamental reason that existing mPSI schemes are not practically efficient is that polynomial multiplication and evaluation over encrypted data are too expensive.

Accelerating these operations needs a well-known method called the fast Fourier transformation, or simply FFT. Then polynomial multiplication incurs linear complexity in multiplications on $\mathbf{R}$. However, this technique requires a polynomial to be written by point-value pairs instead of its coefficients. Thus when there are required to evaluate at the product of polynomials, we again have to rewrite the polynomial by its coefficients. In fact, all prior work based on OPE should do polynomial evaluation in the last step. This last step for polynomial evaluation over encrypted data requires at least $\mathcal{O}(k \log^2 k)$ exponentiations. This is the direction that Cheon et al. pursue in [7].

The essence of our idea is simple. While OPE-based mPSIs view each element $a_i \in \mathtt{X}$ as an irreducible element $(x - a_i) \in \mathbf{R}[x]$, we view each element a_i as an *irreducible* element in $\mathbf{R}$ where $i \in [1, k]$. Here we should use $\mathbf{R}$ that has non-trivial irreducible elements, for example, the integers.

3.1 Map-To-Prime

The trivial algorithm for prime representation is as follows. Let $\mathbf{R} = \mathbb{Z}$. Given as input an element $a \in \mathtt{X}$, using a hash function $H : \{0,1\}^* \to \mathbb{Z}$ it first computes $\alpha = H(a)$. The algorithm then determines whether α is prime or composite. If α is prime, the algorithm outputs α and terminates; otherwise it increments α and checks if $\alpha + 1$ is prime or not. The algorithm repeats this process until obtaining prime.

The above algorithm appears to be suitable for our purpose. However, this solution may be still problematic: if a probabilistic algorithm (e.g., Miller-Rabin algorithm) is employed to determine whether a given value is prime, some composite numbers could be declared "probably prime" with some probability. This may make our protocol work incorrectly. Therefore, we have to use a deterministic algorithm for primality test such as the Agrawal-Kayal-Saxana (AKS) algorithm [1]. In turn, we face to the problem that the AKS algorithm is not efficient enough to be used in practice.

In order to address both non-determinism and inefficiency, we utilize the prime number table $\mathcal{P}_\eta$ that contains η-bit primes. Then we only have to define a random hash function to an index of the table. More specifically, denote by $\wp$ a function to a prime table $\wp : \{0,1\}^* \to \mathcal{P}_\eta$ and denote by H a hash function $H : \{0,1\}^* \to \{1, \ldots, \ell\}$ where ℓ is a constant. However, we can see that the function $\wp$ is not collision-free and must be accommodated in some way. For that, we define a process that throws prime numbers into ℓ buckets, such that each bucket contains at most m elements. We will briefly analyze the collision probability later in this section. Our simple algorithm is as follows:

1. Access to a prime table $\mathcal{P}_\eta$ consisting of η-bit primes.
2. For each $a_i \in \mathtt{X}$
   - $\alpha_i = \wp(a_i)$
   - Add α_i to a bucket $\mathcal{B}_j$ where $j = H(\alpha_i)$ for some $j \in \{1, \ldots, \ell\}$.
3. Return $\{\mathcal{B}_j\}_{j=1}^{\ell}$.

Brief Analysis. When it is assumed that the function $\wp$ is uniformly random, the probability that collision does not occur in m results of $\wp$ from distinct elements is

$$\left(1 - \frac{1}{|\mathcal{P}_\eta|}\right) \times \cdots \times \left(1 - \frac{m}{|\mathcal{P}_\eta|}\right) \geq \left(1 - \frac{m}{|\mathcal{P}_\eta|}\right)^m .$$

In our protocol, because we do not need to take care of collision between data elements in different buckets and the average number of elements in each bucket is small (e.g., $m \approx 10$), the probability that collision by $\wp$ occurs in a given bucket is negligible if the size of $\mathcal{P}_\eta$ is sufficiently large (e.g., $|\mathcal{P}_\eta| = 2^{20}$). Moreover, the size of $\mathcal{P}_\eta$ does not depend on the cardinality of datasets since the problem of large datasets can be addressed by adding to the number of buckets. The set of all 20-bit primes is a good example of $\mathcal{P}_\eta$.

4 Our mPSI Protocol

In this section, we provide our mPSI protocol secure against a semi-honest adversaries and analyze the security of the proposed mPSI protocol. Then we compare the complexities with previous mPSI protocols.

Notation. We denote the map-to-prime function by $\wp : \{0,1\}^* \to \mathcal{P}_\eta$ and a uniform random hash function by $H : \{0,1\}^* \to \{1,\ldots,\ell\}$ where ℓ is the number of buckets. $\mathcal{E}_{pk}(\cdot)$ denotes a threshold additive homomorphic encryption scheme. In particular, in this paper we use a threshold version [12] of Paillier's cryptosystem [23] with 2048-bit Paillier modulus N^2, whose message space is $\mathbb{Z}_N^*$.

4.1 Protocol Description

Now, we are ready to describe our mPSI. Each player A, B participates in the protocol with own private input $\mathtt{X}_A = \{a_1,\ldots,a_k\}$ and $\mathtt{X}_B = \{b_1,\ldots,b_k\}$, respectively. For each private element, each player first calculates its bucket index and then maps to a prime using $\wp$. Then, for each bucket index j, the player A (resp. the player B) computes $A_j = \prod_{j=H(a_i)} \wp(a_i)$ (resp., $B_j = \prod_{j=H(b_i)} \wp(b_i)$). In addition, the player A (resp., the player B) chooses random elements r_1, r_2 (resp., s_1, s_2) in $\mathbb{Z}_{\lfloor \sqrt{N}/4 \rfloor}$ for each bucket. Then for each bucket, players A and B do the following:

1. The player A (resp., the player B) computes $\mathcal{E}_{pk}(r_2), \mathcal{E}_{pk}(A_j^2), \mathcal{E}_{pk}(r_1 A_j^2)$ (resp., $\mathcal{E}_{pk}(s_1)$, $\mathcal{E}_{pk}(B_j^2)$, $\mathcal{E}_{pk}(s_2 B_j^2)$) and sends them to his counterpart.
2. Each player computes $\mathcal{E}_{pk}((r_1+s_1)A_j^2 + (r_2+s_2)B_j^2)$ using additive homomorphic property.
3. Players A and B perform a threshold decryption to obtain $(r_1+s_1)A_j^2 + (r_2+s_2)B_j^2$.
4. Each player checks whether $\wp(a)^2 \mid (r_1+s_1)A_j^2 + (r_2+s_2)B_j^2$ or not for all own private input a whose bucket index is j. If $\wp(a)^2$ divides $(r_1+s_1)A_j^2 + (r_2+s_2)B_j^2$, then a is included in the intersection.

Common Input: $(\mathcal{E}_{pk}(\cdot),\ H,\ \wp)$

Player A's private input:
$\mathtt{X}_A = \{a_1, \ldots, a_k\}$

Player B's private input:
$\mathtt{X}_B = \{b_1, \ldots, b_k\}$

For all $i = 1, \ldots, k$,
$A_{H(a_i)} \leftarrow A_{H(a_i)} \cdot \wp(a_i)$

For all $i = 1, \ldots, k$,
$B_{H(b_i)} \leftarrow B_{H(b_i)} \cdot \wp(b_i)$

For each $j = 1, \ldots, \ell$

$r_1 \leftarrow_R \mathbb{Z}_{\lfloor \sqrt{N}/4 \rfloor}$,
$r_2 \leftarrow_R \mathbb{Z}_{\lfloor \sqrt{N}/4 \rfloor}$,
computes
$\mathcal{E}_{pk}(r_2), \mathcal{E}_{pk}(A_j^2), \mathcal{E}_{pk}(r_1 A_j^2)$

$s_1 \leftarrow_R \mathbb{Z}_{\lfloor \sqrt{N}/4 \rfloor}$,
$s_2 \leftarrow_R \mathbb{Z}_{\lfloor \sqrt{N}/4 \rfloor}$,
computes
$\mathcal{E}_{pk}(s_1), \mathcal{E}_{pk}(B_j^2), \mathcal{E}_{pk}(s_2 B_j^2)$

$$\xrightarrow{\mathcal{E}_{pk}(r_2),\mathcal{E}_{pk}(A_j^2),\mathcal{E}_{pk}(r_1 A_j^2)}$$

$$\xleftarrow{\mathcal{E}_{pk}(s_1),\mathcal{E}_{pk}(B_j^2),\mathcal{E}_{pk}(s_2 B_j^2)}$$

computes
$$\begin{aligned}&\mathcal{E}_{pk}((r_1+s_1)A_j^2 + (r_2+s_2)B_j^2)\\ &= (A_j^2 *_{\mathsf{h}} \mathcal{E}_{pk}(s_1))\\ &\quad +_{\mathsf{h}} (r_2 *_{\mathsf{h}} \mathcal{E}_{pk}(B_j^2)) +_{\mathsf{h}} \mathcal{E}_{pk}(r_1 A_j^2)\\ &\quad +_{\mathsf{h}} \mathcal{E}_{pk}(s_2 B_j^2)\end{aligned}$$

computes
$$\begin{aligned}&\mathcal{E}_{pk}((r_1+s_1)A_j^2 + (r_2+s_2)B_j^2)\\ &= (B_j^2 *_{\mathsf{h}} \mathcal{E}_{pk}(r_2))\\ &\quad +_{\mathsf{h}} (s_1 *_{\mathsf{h}} \mathcal{E}_{pk}(A_j^2)) +_{\mathsf{h}} \mathcal{E}_{pk}(s_2 B_j^2)\\ &\quad +_{\mathsf{h}} \mathcal{E}_{pk}(r_1 A_j^2)\end{aligned}$$

Players A and B perform a threshold decryption to obtain $(r_1+s_1)A_j^2 + (r_2+s_2)B_j^2$. Then each player checks whether $\wp(a)^2$ divides $(r_1+s_1)A_j^2+(r_2+s_2)B_j^2$ or not for all a in his private set, satisfying $j = H(a)$. If $\wp(a)^2 \mid (r_1+s_1)A_j^2 + (r_2+s_2)B_j^2$, a belongs to $\mathtt{X}_A \cap \mathtt{X}_B$.

Fig. 1. Our mPSI for Semi-Honest Model (mPSI-SH)

4.2 Security Analysis

Correctness Players participating in the protocol correctly obtain the intersection of participating players' private inputs. The following lemma shows that our protocol gives the correctness with overwhelming property.

Lemma 1 (Correctness) *Protocol mPSI-SH correctly computes for the function $\mathcal{F}_{\mathsf{mPSI}}$ with overwhelming property.*

Proof. When a is an element in the intersection $\mathtt{X}_A \cap \mathtt{X}_B$, $\wp(a)$ divides A_j and B_j for the bucket $j = H(a)$. Hence $\wp(a)^2$ divides A_j^2, B_j^2, and $(r_1+s_1)A_j^2+(r_2+s_2)B_j^2$. Therefore, each player learns that a is an element in the intersection.

Assume that a is not an element in the intersection $\mathtt{X}_A \cap \mathtt{X}_B$. We do not consider a is not in $\mathtt{X}_A$ and not in $\mathtt{X}_B$, since no players try to check the divisibility of $\wp(a)^2$. Without loss of generality, suppose a is in $\mathtt{X}_A$, but not in $\mathtt{X}_B$. Then, $\wp(a)$ divides A_j, but does not divide B_j. Hence $\wp(a)^2$ divides A_j^2, but does not divide B_j^2.

In order that $\wp(a)^2$ does not divide $(r_1+s_1)A_j^2+(r_2+s_2)B_j^2$, $\wp(a)^2$ should not divide r_2+s_2. Since r_2 and s_2 are chosen randomly in $\mathbb{Z}_{\lfloor\sqrt{N}/4\rfloor}$, the probability that $\wp(a)^2$ divides r_2+s_2 is $\frac{1}{\wp(a)^2}$. It is the probability that the player A misunderstands that a belongs to the intersection. When the bit size of primes in $\mathcal{P}_\eta$ is 20-bit, the probability becomes about $\frac{1}{2^{40}}$. $\square$

Remark 2 *One may object that the message $(r_1+s_1)A_j^2+(r_2+s_2)B_j^2$ may be wrap-rounded by the modular exponentiation of the encryption scheme. Although it is assumed that H is a uniform random hash function, it occurs that some buckets have more than m elements where m is a pre-fixed value such as 10. When players are faced with this situation, they select another uniform random hash function and increase the number of buckets.*

In general, players set m to 10 and use $\mathcal{P}_{20}$ in the protocol. Then since r_i's and s_i's ($i=\{1,2\}$) are chosen at random in $\mathbb{Z}_{\lfloor\sqrt{N}/4\rfloor}$, A_j and B_j are about 20m-bit, $(r_1+s_1)A_j^2+(r_2+s_2)B_j^2$ does not exceed N where N is an 1024-bit integer.

Privacy During participating in our mPSI-SH protocol, an adversary can only obtain inputs of a player who is manipulated by himself, encrypted values, and the last value $(r_1+s_1)A_j+(r_2+s_2)B_j$. Suppose that a utilized additive homomorphic threshold encryption $\mathcal{E}_{pk}(\cdot)$ is semantically secure. Without loss of generality, it is assumed that the player B is manipulated by the adversary. In order that the adversary learns any information of the player A, he has to find the factor of A_j in the equation

$$(r_1+s_1)A_j+(r_2+s_2)B_j=d.$$

Since s_1, s_2, d and B_j are known values to the adversary, it is equivalent to find the factor of an appropriate value of x in the equation

$$xy+c_1x+c_2z=c_3, \tag{1}$$

for variables x, y and z and constant c_1, c_2 and c_3, where x can be a product of m primes in table $\mathcal{P}_\eta$. Equation (1) can be substituted by the equation

$$xy+c_1z=c_2. \tag{2}$$

As far as we know, Equation (2) has finitely many positive integer solutions but there is no efficient algorithm to find solutions. Hence we believe the following conjecture is true.

Conjecture 1 *For variables x, y, z and given constant c_1, c_2, there is no efficient algorithm to find all solutions for Equation (2).*

Moreover, since z is chosen at random in $Z_{\lfloor\sqrt{N}/4\rfloor}$, the number of possible values of z is about 2^{510}. Hence one has to factor about 2^{510} (c_2-c_1z)'s to solve Equation (2).

The following lemma guarantees the security of our mPSI-SH assuming Conjecture 1 is true and an additive homomorphic threshold encryption is semantically secure.

Lemma 2 (Privacy) *Assume that an additive homomorphic threshold encryption $\mathcal{E}_{pk}(\cdot)$ is semantically secure and Conjecture 1 is true, with overwhelming probability, any adversary learns no more information than would be obtained by using the same private inputs in the ideal model with a trusted third party.*

Proof. Since we know that an instance of additively homomorphic encryption is semantically secure, a corrupted player (say B) obtains no information from ciphertexts received from his counterpart.

After engaging in a threshold decryption, the corrupted player learns

$$I = (r_1 + s_1)A_j^2 + (r_2 + s_2)B_j^2.$$

Since only one of players can be controlled over by the adversary, r_i's ($i = \{1, 2\}$) look to be random and are unknown to the adversary.

Hence, by Conjecture 1, $I = (r_1 + s_1)A_j^2 + (r_2 + s_2)B_j^2 = xy + c_1z + c_2$ for some variables x, y and z and constants c_1 and c_2, reveals no information about the private inputs of the honest player (say A), with overwhelming probability, except for that given by computing the intersection of their private sets. $\square$

4.3 Efficiency Analysis

In this subsection, we analyze the computational and communicational complexity of our mPSI protocol. Also we compare the complexities with those of previous mPSI protocols.

In our protocol, we utilize integer multiplications, modular exponentiations (ME) and integer divisions. Among these operations, ME is the most expensive operations. Hence, we analyze and compare the computational complexity based on the number of MEs.

Complexity of Our mPSI-SH In our protocol, each player sends three ciphertexts per each bucket. Also each player sends one element to perform a threshold decryption per each bucket. Hence the total communication complexity of mPSI-SH is $8\ell \approx 8k/m$ ciphertexts when ℓ is the number of buckets, k is the cardinality of private input sets, and m is a pre-fixed number which is the bound of the number of elements in a bucket. Therefore, the communication complexity of mPSI-SH is $\mathcal{O}(k)$.

In case of the computational complexity, it is assumed that the threshold Paillier encryption [12] is utilized, which requires 2 MEs for one encryption and 3 MEs (1 ME for share decryption and 2 MEs for share combining) for a threshold decryption per each player. Hence, per each bucket, each player requires 6 MEs for encryptions, 2 MEs for $\mathcal{E}_{pk}((r_1 + s_1)A_j^2 + (r_2 + s_2)B_j^2)$ computation using additive homomorphic property and 3 MEs for a threshold decryption. Therefore, the total computational complexity is $22\ell \approx 22k/m$ MEs and hence it is $\mathcal{O}(k)$.

Table 1. Complexity Comparison

Protocol	Computation	Communication
[19]	$\mathcal{O}(k^2)$	$\mathcal{O}(k)$
[11]	$\mathcal{O}(k^2 \log k + k \log^2 k)$	$\mathcal{O}(k \log^2 k)$
[5]	$\mathcal{O}(k^2)$	$\mathcal{O}(k)$
[7]	$\mathcal{O}(k \log^2 k)$	$\mathcal{O}(k)$
OURS	$\mathcal{O}(k)$	$\mathcal{O}(k)$

Comparison with Previous Works As mentioned before, Kissner and Song [19], Dachman-Soled et al. [11], Camenisch and Zaverucha [5], and Cheon et al. [7] proposed mPSI protocols. Referred to previous analyses, their work has linear communication complexity and more than quasi-linear computational complexity. Table 1 compares our mPSI protocol with the complexity of previous mPSI protocols when two players participate in the protocol.

Remark 3 (ElGamal Encryption vs. Paillier Encryption) *While our* mPSI-SH *can not utilize the threshold ElGamal encryption scheme, previous mPSI's [19, 5, 7] can utilize the ElGamal encryption or a threshold ElGamal encryption. In case of the ElGamal or a threshold ElGamal encryption over elliptic curves, the ciphertext size is 320-bit, but that of the threshold Paillier encryption is 2048-bit. We would like to note that since our protocol requires $7k/m$ ciphertexts, the total transmitted bits are similar to those of other protocols. When $m = 10$, our protocol transmits $1433.6k$ $(= 2048 \cdot 7k/10)$ bits. However, protocols in [19, 5, 7] transmit $3480k$, $960k$ and $3480k$ bits, respectively.*[2]

In case of ME, 160-bit ME over 1024-bit modulus is about 30 times faster than 1024-bit ME over 2048-bit modulus. Hence, ME in the ElGamal encryption is 30 times faster than that in the Paillier encryption. However the factor m can cancel out the effect of the use of the ElGamal encryption. Since the constant term of the computational complexity of other mPSI protocols are similar with that of ours, the relation between the computational complexities still holds.

5 Transformation to a Malicious Protocol

In this section, we discuss modifications of the protocol mPSI-SH so as to be secure in the malicious model. We add zero-knowledge proofs to mPSI-SH in order to ensure the correctness of all computation. Note that we just provide a sketch for a way to construct a malicious protocol instead of detailed descriptions. Moreover, it should be pointed out that in general generic zero-knowledge proofs are not efficient – especially range proof used in our protocol, and so although our malicious protocol still has the asymptotically linear complexity it may work inefficiently.

We take a look at deviating activities by a malicious player (let say B). The following malicious activities should be taken care:

[2] The communication complexity of protocols in [19, 5, 7] are $12k$, $3k$ and $12k$, respectively.

1. A malicious player uses a random s_i such that $s_i \geq \lfloor \frac{\sqrt{N}}{4} \rfloor$ for $i = 1, 2$.
2. A malicious player makes a product B_j for $j \in [1, \ell]$ that is multiplied by more than m primes.

Recall that any other correctness can be detected at the beginning of a threshold decryption. For example, one may think that as a verifier the player A should check the player B has correctly multiplied $\mathcal{E}_{pk}(A_j^2)$ by s_1. However, if the corrupted player B participates in the threshold decryption with different values committed to in encryptions, the honest player can detect his counterpart has a different value and so he can abort the protocol.

Zero-Knowledge Proofs We use $PK\{(a)|\phi(a)\}$ to denote a zero-knowledge proof of knowledge of the value a that satisfies a publicly computable relation ϕ. For the Paillier encryption, we can efficiently construct zero-knowledge proofs using well-known constructions [3, 8, 4]. Let $\mathcal{C} = (\mathsf{G}, \mathsf{C}, \mathsf{O})$ be the generation, the commit and the open algorithm of a trapdoor commitment scheme [20].

– $PK\{(\alpha_1, \alpha_2)|u = \mathsf{C}(\alpha_1) \wedge v = \mathcal{E}_{pk}(\alpha_2) \wedge w = \alpha_1 *_\mathsf{h} v\}$: a zero-knowledge proof of knowledge that C encrypts $\alpha_1\alpha_2 \pmod N$ [8, Sec. 8.1.2].
– $PK\left\{(r)|C = \mathcal{E}_{pk}(r) \wedge r \in \left[0, \lfloor \frac{\sqrt{N}}{4} \rfloor\right]\right\}$: a zero-knowledge proof of knowledge that r lies in $r \in \left[0, \lfloor \frac{\sqrt{N}}{4} \rfloor\right]$ [4].
– $PK\left\{(\alpha_1, \ldots, \alpha_m)| \wedge_{i=1}^{m} (u_i = \mathsf{C}(\alpha_i)) \wedge_{i=1}^{m-1} (v_i = \mathcal{E}_{pk}(\alpha_{i+1})) \wedge_{i=1}^{m-1} (w_i = w_{i-1} *_\mathsf{h} v_i)\right\}$ where $w_0 = \alpha_1$: the generalized proof of $PK\{(\alpha_1, \alpha_2)|u = \mathsf{C}(\alpha_1) \wedge v = \mathcal{E}_{pk}(\alpha_2) \wedge w = \alpha_1 *_\mathsf{h} v\}$ for m tuples $\{\alpha_1, \ldots, \alpha_m\}$ [3, 8].

Transformation Recall that the primary purpose of this section is to show that using generic zero-knowledge techniques we can convert our mPSI-SH to one that is secure in the malicious model. To do so, when the player A sends $\mathcal{E}_{pk}(r_2)$ to the player B, he sends it along with a zero-knowledge proof of range proof (the second PK in the above list). The player A sends $\mathcal{E}_{pk}(r_1 A_j^2)$ along with a zero-knowledge proof of the correct multiplication, i.e., the first PK. Further, he sends $\mathcal{E}_{pk}(A_j^2)$ along with the third PK, which is the generalization of a zero-knowledge proof of the correct multiplication.

6 Conclusion

In this work, primarily we present a mutual private set intersection protocol with linear complexity. Further we compare our construction with existing work and show it is secure in the semi-honest model. However, there is still remaining work as follows: (1) present a detailed description for the malicious mPSI protocol, (2) show that this construction is secure in the malicious model in the simulation paradigm, and (3) finally extend it to the multiparty setting.

References

1. M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Annals of Mathematics (2)*, 160(2):781–793, 2004.
2. R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private database. In A. Halevy, Z. Ives, and A. Doan, editors, *SIGMOD*, pages 86–97, 2003.
3. J. Camenisch. Proof systems for general statements about discrete logarithms. Technical Report TR 260, Dept. of Computer Science, ETH Zurich, 1997.
4. J. Camenisch, R. Chaabouni, and abhi shelat. Efficient protocols for set membership and range proofs. In J. Pieprzyk, editor, *Advances in Cryptology-AsiaCrypt*, LNCS 5350, pages 234–252, 2008.
5. J. Camenisch and G. Zaverucha. Private intersection of certified sets. In R. Dingledine and P. Golle, editors, *Financial Cryptography*, LNCS 5628, pages 108–127, 2009.
6. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. Rivest, and A. Sherman, editors, *Advances in Cryptology-Crypto*, pages 199–203, 1982.
7. J. H. Cheon, S. Jarecki, and J. H. Seo. Multi-party privacy-preserving set intersection with quasi-linear complexity. In *Cryptology ePrint Archive*, 2010/512, 2010.
8. R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In B. Pfitzmann, editor, *Advances in Cryptology-EuroCrypt*, LNCS 2045, pages 280–299, 2001.
9. E. D. Cristofaro, J. Kim, and G. Tsudik. Linear-complexity private set intersection protocols secure in malicious model. In M. Abe, editor, *Advances in Cryptology-AsiaCrypt*, LNCS 6477, pages 213–231, 2010.
10. E. D. Cristofaro and G. Tsudik. Practical private set intersection protocols with linear computational and bandwidth complexity. In R. Sion, editor, *Financial Cryptography*, LNCS 6052, pages 143–159, 2010.
11. D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung. Efficient robust private set intersection. In M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, editors, *ACNS*, LNCS 5536, pages 125–142, 2009.
12. I. Damgård and M. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In K. Kim, editor, *Public Key Cryptography*, LNCS 1992, pages 119–136, 2001.
13. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakely and D. Chaum, editors, *Advances in Cryptology-Crypto*, LNCS 196, pages 10–18, 1984.
14. M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set-intersection. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology-EuroCrypt*, LNCS 3027, pages 1–19, 2004.
15. O. Goldreich. *The foundations of cryptography*, volume 2. Cambridge University Press, 2004.
16. C. Hazay and Y. Lindell. Efficient protocols for set intersection and pattern matching with security against mailicious and covert adversaries. In R. Canetti, editor, *TCC*, LNCS 4948, pages 155–175, 2008.
17. C. Hazay and K. Nissim. Efficient set operations in the presence of malicious adversaries. In P. Q. Nguyen and D. Pointcheval, editors, *Public Key Cryptography*, LNCS 6056, pages 312–331, 2010.

18. S. Jarecki and X. Liu. Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In O. Reingold, editor, *TCC*, LNCS 5444, pages 577–594, 2009.
19. L. Kissner and D. Song. Privacy-preserving set operations. In V. Shoup, editor, *Advances in Cryptology-Crypto*, LNCS 3621, pages 241–257, 2005.
20. P. MacKenzie and K. Yang. On simulation-sound trapdoor commitments. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology-EuroCrypt*, LNCS 3027, pages 382–400, 2004.
21. G. Mezzour, A. Perrig, V. Gligor, and P. Papadimitratos. Privacy-preserving relationship path discovery in social networks. In J. Garay, A. Miyaji, and A. Otsuka, editors, *CANS*, LNCS 5888, pages 189–208, 2009.
22. M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *STOC*, pages 245–254, 1999.
23. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Advances in Cryptology-EuroCrypt*, LNCS 1592, pages 223–238, 1999.
24. V. Shoup. Practical threshold signatures. In B. Preneel, editor, *Advances in Cryptology-EuroCrypt*, LNCS 1807, pages 207–220, 2000.