# THIRD-PARTY PRIVATE SET INTERSECTION

FOO YEE YEO AND JASON H. M. YING

Abstract. Private set intersection (PSI) enables two parties, each holding a private set to compute their intersection without revealing other information in the process. We introduce a variant of conventional PSI termed as third-party PSI, whereby the intersection output of the two parties is only known to an inputless third party. In this setting, the two parties who participate in the protocol have no knowledge of the intersection result or any information of the set content of the other party. In general, third-party PSI settings arise where there is a need for an external party to obtain the intersection outcome without leakage of additional information to any other party. This setting is motivated by an increasing importance in several real-world applications. We describe protocols which achieve this functionality with minimal communication overhead. To the best of knowledge, our work is the first of its kind to explore this variant of PSI.

## 1. Introduction

Private set intersection (PSI) was first introduced in [9], incorporating ideas tracing back to [14, 23]. PSI enables the secure computation of the intersection of datasets held by two different parties. While this is a special case of secure multi-party computation, PSI protocols are in general much more efficient, thus allowing for practical applications on much larger datasets. An early PSI protocol proposed by Freedman *et al.* [6] cleverly applies a combination of oblivious polynomial evaluation and homomorphic encryption with the additive property.

Conventional two-party PSI has seen many diverse applications such as botnet detection [15], human genomes testing [1], private proximity testing [16], online advertising [18], privacy-preserving ride-sharing [7] and contact tracing [5]. Due to its utility, a vast amount of research has been devoted to improve the efficiency of PSI with respect to both communication and computational costs. Several state-of-the art PSI schemes include [12, 20, 21].

In recent times, there has been an increasing number of scenarios whereby an inputless third party requires the output of intersection between datasets held by two other parties. The privacy requirements are such that only the third party obtains the intersection output and receives no other information of the input sets of either party. Moreover, no information is revealed to either of the participating parties in the process. For example during a pandemic, various venues such as malls, supermarkets and places of worship are frequented by people. These premises keep an identity record of people who visit, along with their times of entry and exit for the purpose of contact tracing. In the event of a virus cluster at two or more of these locations, a regulatory organization is then able to obtain the database of people who are present on specific times at these respective premises to identify potential asymptomatic sources of the clusters in a privacy preserving manner. Another such scenario arises when a medical body such as the health ministry seeks to obtain genomic information of individuals contained in the list of two other medical

institutions to perform a research study. In this situation, the medical body being the third-party is able to obtain the required information without any leakage to the participating medical institutions.

While the above situations occur surprisingly often, there has not been a suitable privacy-preserving solution adopted to this problem in the literature to our knowledge thus far. The work here aims to bridge this gap. For the intended use cases in third-party PSI, there is no incentive for any party to deviate from the protocols and execution is regulated by policies in many instances. As such, the work in this paper focuses on the semi-honest security model. In this model, adversaries can attempt to obtain information from the execution of the protocol but they are unable to perform any deviations from the intended protocol steps.

Despite the enormous strides achieved in conventional two-party PSI, such schemes do not translate to a solution in the above setting. There has been research relating to PSI which involves an additional entity such as a server. For instance, Kamara *et al.* [11] described very efficient schemes for two parties to conduct PSI by utilizing an external server for assistance to perform an encrypted intersection before relaying back to the respective parties to decrypt and compute. In [13], the server is also used as an assistance tool to compute intersection cardinality. In essence, the above works are completely different to the existing problem in hand since the receiver of the intersection set are the participating parties in [11] instead of the intended third party. The complexity of the problem increases when the participating parties with inputs are unable to learn any information such as in the scenario we consider.

There are two major aspects when considering an efficiency of a scheme: computational cost and communication overhead. Due to advances in hardware, communication overhead has gradually been presented as a larger bottleneck than computational cost. Indeed, the work of Google [10] and Rosulek & Trieu [21] serve to highlight that communication costs far outweigh computation especially in the business settings as it is much less expensive to add CPUs than to expand network capacity. Hence, these protocols are constructed with communication cost as the primary consideration. As such, we aim to provide a practical solution with minimal communication overhead.

In this paper, we propose a variant of PSI, which we call third-party PSI. In third-party PSI, we privately compute the intersection of the datasets between two participating parties $P_1$, $P_2$ such that the result is revealed only to a third-party $Q$.

## 2. Preliminaries

**Definition 1** (Third-party PSI scheme)**.** *In a third-party PSI scheme,* 2 *parties* $P_1$ *and* $P_2$ *each holds a dataset with elements in* $\{0,1\}^*$*, while a third-party* $Q$ *has no input. At the end of the protocol,* $Q$ *outputs the set intersection functionality, and the other parties output* $\perp$*.*

We shall use ideal-world/real-world simulation-based definitions in order to define the security of a third-party PSI scheme. Specifically, we say that a third-party PSI scheme is secure in the semi-honest model if it achieves the ideal functionality shown in Figure 1 against semi-honest adversaries. In other words, $P_1$ and $P_2$ do not learn any information about the other parties' datasets, while $Q$ only learns the intersection, but nothing else.

(1) Get $P_1$'s input set $S_1$.
(2) Get $P_2$'s input set $S_2$.
(3) Send $S_1 \cap S_2$ to $Q$.

FIGURE 1. Third-party PSI ideal functionality

## 3. A Third-Party PSI Scheme

In this section, we will introduce the first third-party PSI scheme, which is secure against a semi-honest adversary corrupting a single party. Suppose $P_1$ and $P_2$ have sets $S_1 \subseteq \{0,1\}^\ell$ and $S_2 \subseteq \{0,1\}^\ell$ respectively, each of size $n$, and they wish to privately compute the intersection of their sets, while revealing the result only to a third party $Q$.

Our scheme relies on the use of a deterministic commutative cipher. The basic idea is for $P_1$ and $P_2$ to first encrypt their elements using a common key $k$. $Q$ can then compute the intersection on the encrypted elements, and encrypt the intersection result again using a different key $k'$. The double encrypted intersection result is now decrypted by $P_1$, and then decrypted by $Q$ to reveal the intersection.

However, in order to hide the size of the intersection from $P_1$, it is necessary for $Q$ to pad the encrypted intersection result with enough elements to a total of $n$ elements (see step 5). To ensure that this can be done regardless of the actual intersection size, we shall identify $\{0,1\}^\ell$ with a subset of $\{0,1\}^{\ell+1}$. Let $E : \mathcal{K} \times \{0,1\}^{\ell+1} \to \{0,1\}^{\ell+1}$ be a deterministic commutative cipher with keyspace $\mathcal{K}$.

If $h$ is a positive integer, we let $[h] = \{1, 2, \ldots, h\}$. For any sequence $A = (a_1, a_2, \ldots, a_n)$ and any $L \subseteq [n]$, we shall denote the sets $\{a_i \in A : i \in L\}$ and $\{a_i \in A : i \notin L\}$ by $A_{\in L}$ and $A_{\notin L}$ respectively.

We will assume that secure communication channels exist between any two parties. Otherwise, the participating parties can carry out a simple key exchange protocol prior. Our protocol works as follows:

(1) $P_1$ and $P_2$ agree on a random key $k \leftarrow \mathcal{K}$.
(2) For each $i = 1, 2$, party $P_i$ computes the set
$$S'_i = E_k(S_i) = \{E_k(x) : x \in S_i\},$$
randomly rearranges its elements, and sends the result to $Q$.
(3) $P_1$ randomly chooses a set of $n$ elements $T \subseteq \{0,1\}^{\ell+1} \setminus \{0,1\}^\ell$ and sends $T' = E_k(T)$ to $Q$.
(4) $Q$ performs an intersection of the sets $S'_1$ and $S'_2$ to obtain $S' = S'_1 \cap S'_2$.
(5) Let $n' = |S'|$. $Q$ chooses $n - n'$ random elements from $T'$ and appends them to $S'$ to obtain a sequence $U$.
(6) $Q$ picks a random key $k' \leftarrow \mathcal{K}$ and encrypts $U$ to obtain $U' = E_{k'}(U)$.
(7) (a) $Q$ sends $U'$ to $P_1$.
    (b) $P_1$ decrypts $U'$ using the key $k$ to obtain $U'' = E_k^{-1}(U')$.
    (c) $P_1$ sends $U''$ to $Q$.
(8) $Q$ decrypts $U''$ using the key $k'$ to obtain $R = E_{k'}^{-1}(U'')$, and outputs $R_{\in [n']}$.

PROTOCOL 1. A semi-honest third-party PSI protocol

We will now prove the correctness and security properties of Protocol 1.

**Proposition 1.** *Assume $E$ is a deterministic commutative cipher. Then Protocol 1 always outputs the correct intersection.*

*Proof.* Let $I = S_1 \cap S_2$. First, observe that

$$S' = S'_1 \cap S'_2 = E_k(S_1) \cap E_k(S_2) = E_k(S_1 \cap S_2) = E_k(I).$$

Thus, by construction of $U$, we have $U_{\in[n']} = S' = E_k(I)$. It then follows from $R = E_{k'}^{-1}(E_k^{-1}(E_{k'}(U)))$ that

$$R_{\in[n']} = E_{k'}^{-1}(E_k^{-1}(E_{k'}(E_k(I)))) = I,$$

where the last equality uses the assumption that $E$ is a commutative cipher. □

To prove security against semi-honest adversaries, it suffices to consider parties $P_1$ and $Q$, since they are the only parties who receive messages when running the protocol.

**Proposition 2.** *Assume $E$ is a secure pseudorandom permutation (PRP). Then Protocol 1 is secure against a semi-honest $P_1$.*

*Proof.* The only message that $P_1$ receives is the sequence $U' = E_{k'}(U)$, where $U$ is independent of the key $k'$. Since $P_1$ does not know $k'$, the assumption that $E$ is a secure PRP means that changing $U'$ to $n$ uniformly random elements of $\{0,1\}^{\ell+1}$ is indistinguishable to $P_1$. □

**Proposition 3.** *Assume $E$ is a secure PRP. Then Protocol 1 is secure against a semi-honest $Q$.*

*Proof.* We shall change how $S'_1$ and $S'_2$ are computed. Specifically, the simulator chooses $2(n - n')$ distinct random elements $s'_1, s'_2, \ldots, s'_{n-n'}, t'_1, t'_2, \ldots, t'_{n-n'}$ from $\{0,1\}^\ell \setminus (S_1 \cap S_2)$, and defines

$$S'_1 = E_k(S_1 \cap S_2) \cup E_k(\{s'_1, s'_2, \ldots, s'_{n-n'}\}),$$
$$S'_2 = E_k(S_1 \cap S_2) \cup E_k(\{t'_1, t'_2, \ldots, t'_{n-n'}\}).$$

Since $E$ is a secure PRP, and since $T'$ and $U''_{\notin[n']}$ are both independent of $S_1$ and $S_2$, this interaction is indistinguishable from the real interaction. □

Note that the operations used in Protocol 1 are simply $O(n)$ encryptions and decryptions, and an intersection operation, hence, if we have fast encryptions and decryptions, Protocol 1 will be quite computationally efficient. However, in practice, commutative ciphers such as Pohlig-Hellman [19] and SRA [24] can be slow. The communication cost of Protocol 1 is $5n(\ell + 1)$ bits.

Furthermore, it is important to note that both Pohlig-Hellman and SRA are not secure PRPs since they are homomorphic in their inputs. Hence, it is necessary to modify Protocol 1 when using either of these ciphers. By adapting an idea in [10], we now provide a suitably modified version of Protocol 1, that allows the use of a variant of Pohlig-Hellman as the deterministic commutative cipher, via the addition of an ideal permutation in step 2 of the protocol.

Let $G$ be a group of prime order $p$ such that the decisional Diffie-Hellman (DDH) assumption holds in $G$. For example, we can pick primes $p$ and $q$ such that $q = 2p+1$ ($p$ is known as a Sophie Germain prime), and let $G$ be the subgroup of order $p$ of $\mathbb{F}_q^*$. Assume $p > 2^{\ell+1}$. We will identify $\{0,1\}^{\ell+1}$ with some subset of $G$. Fix an ideal permutation $\Pi : G \to G$. The modified protocol is as follows:

(1) $P_1$ and $P_2$ agree on a random key $k \leftarrow \mathbb{Z}_p^*$.
(2) For each $i = 1, 2$, party $P_i$ computes the set
$$S_i' = (\Pi(S_i))^k = \{\Pi(x)^k : x \in S_i\},$$
randomly rearranges its elements, and sends the result to $Q$.
(3) $P_1$ randomly chooses a set of $n$ elements $T \subseteq \{0,1\}^{\ell+1} \setminus \{0,1\}^\ell$ and sends $T' = (\Pi(T))^k$ to $Q$.
(4) $Q$ performs an intersection of the sets $S_1'$ and $S_2'$ to obtain $S' = S_1' \cap S_2'$.
(5) Let $n' = |S'|$. $Q$ chooses $n - n'$ random elements from $T'$ and appends them to $S'$ to obtain a sequence $U$.
(6) $Q$ picks a random key $k' \leftarrow \mathbb{Z}_p^*$ and computes $U' = U^{k'}$.
(7) (a) $Q$ sends $U'$ to $P_1$.
(b) $P_1$ decrypts $U'$ using $k^{-1}$ to obtain $U'' = (U')^{k^{-1}}$.
(c) $P_1$ sends $U''$ to $Q$.
(8) $Q$ decrypts $U''$ using $(k')^{-1}$ to obtain $R = (U'')^{(k')^{-1}}$, and outputs $\Pi^{-1}(R_{\in[n']})$.

PROTOCOL 1′. A semi-honest third-party PSI protocol

We now reduce the security of Protocol 1′ to the DDH assumption in the group $G$.

**Proposition 4.** *Suppose the DDH assumption holds in $G$. Then Protocol 1′ is secure against a semi-honest $P_1$.*

*Proof.* We prove this using a sequence of hybrids.

*Hybrid 0*: The real interaction.

*Hybrid 1*: We change how the ideal permutation $\Pi$ is simulated. Fix a generator $g$ of the cyclic group $G$. For each query $x$ to the ideal permutation $\Pi$, we choose a random $\alpha_x \leftarrow \mathbb{Z}_p$ and set $\Pi(x) = g^{\alpha_x}$, aborting if $\alpha_x = \alpha_y$ for some previously queried $y$. For sufficiently large $p$, the probability of abort is negligible. Since $g^{\alpha_x}$ is a uniformly random element of $G$, this hybrid is indistinguishable from Hybrid 0.

*Hybrid $(2, h)$ for $h \in [n+1]$*: We change how the sequence $U' = (u_1', \ldots, u_n')$ is computed. Let $m = |S_1 \cap S_2|$ and $S_1 \cap S_2 = \{x_1, \ldots, x_m\}$. Pick distinct random elements $x_{m+1}, \ldots, x_n \leftarrow T$. We set
$$u_i' = \begin{cases} \Pi(x_i)^{kk'} = \left(g^{k'\alpha_{x_i}}\right)^k & \text{if } i \geq h, \\ \left(g^{\beta_i}\right)^k \text{ where } \beta_i \leftarrow \mathbb{Z}_p & \text{otherwise,} \end{cases}$$
aborting if $u_i' = u_j'$ for some $i \neq j$. The probability of abort is, again, negligible for sufficiently large $p$. Note that Hybrid $(2, 1)$ is identical to Hybrid 1. By the DDH assumption, $P_1$ cannot distinguish between the triples $(g^{k'}, g^{\alpha_{x_h}}, g^{k'\alpha_{x_h}})$ and $(g^{k'}, g^{\alpha_{x_h}}, g^{\beta_h})$, hence, for each $h \in [n]$, Hybrids $(2, h)$ and $(2, h+1)$ are indistinguishable. Therefore, Hybrid $(2, n+1)$ is indistinguishable from Hybrid 1.

*Simulator*: We simulate $U' = (g_1^k, \ldots, g_n^k)$ where $g_1 \ldots, g_n \leftarrow G$ are distinct random elements of $G$. This interaction is identically distributed to Hybrid $(2, n+1)$. □

**Proposition 5.** *Suppose the DDH assumption holds in $G$. Then Protocol 1′ is secure against a semi-honest $Q$.*

*Proof. Hybrid 0*: The real interaction.

*Hybrid 1*: Fix a generator $g$ of the cyclic group $G$. For each query $x$ to the ideal permutation $\Pi$, choose a random $\alpha_x \leftarrow \mathbb{Z}_p$ and set $\Pi(x) = g^{\alpha_x}$, aborting if $\alpha_x = \alpha_y$ for some previously queried $y$. This hybrid is indistinguishable from Hybrid 0.

*Hybrid* $(2, h)$ *for* $h \in [n+1]$: We modify how $S_1'$ is computed. Let $S_1 = \{s_1, \ldots, s_n\}$ and $m = |S_1 \cap S_2|$. We set $S_1' = \{s_1', \ldots, s_n'\}$ where

$$s_i' = \begin{cases} \Pi(s_i)^k = g^{k\alpha_{s_i}} & \text{if } s_i \in S_2 \text{ or } i \geq h, \\ g^{\beta_i} \text{ where } \beta_i \leftarrow \mathbb{Z}_p & \text{otherwise,} \end{cases}$$

aborting if $s_i' = s_j'$ for some $i \neq j$ or $|S_1' \cap S_2'| > m$. The probability of abort is negligible for sufficiently large $p$. Since $(g^k, g^{\alpha_{s_h}}, g^{k\alpha_{s_h}})$ and $(g^k, g^{\alpha_{s_h}}, g^{\beta_h})$ are indistinguishable by the DDH assumption, Hybrid $(2, h)$ is indistinguishable to Hybrid $(2, h+1)$ for $h \in [n]$. Thus, Hybrid $(2, n+1)$ is indistinguishable from Hybrid 1.

*Hybrid* $(3, h)$ *for* $h \in [n+1]$: We modify how $S_2'$ is computed. Write $S_2 = \{t_1, \ldots, t_n\}$. We set $S_2' = \{t_1', \ldots, t_n'\}$ where

$$t_i' = \begin{cases} \Pi(t_i)^k = g^{k\alpha_{t_i}} & \text{if } t_i \in S_1 \text{ or } i \geq h, \\ g^{\gamma_i} \text{ where } \gamma_i \leftarrow \mathbb{Z}_p & \text{otherwise,} \end{cases}$$

aborting if $t_i' = t_j'$ for some $i \neq j$ or $|S_1' \cap S_2'| > m$. Hybrid $(3, n+1)$ is indistinguishable from Hybrid $(2, n+1)$ by the DDH assumption.

*Simulator*: Let $S_{1,2}' = \{\Pi(x)^k : x \in S_1 \cap S_2\}$. We simulate

$$S_1' = S_{1,2}' \cup \{g_1, \ldots, g_{n-m}\}, \quad S_2' = S_{1,2}' \cup \{h_1, \ldots, h_{n-m}\},$$

for distinct random elements $g_1, \ldots, g_{n-m}, h_1, \ldots, h_{n-m} \leftarrow G \setminus S_{1,2}'$. This interaction is identically distributed to Hybrid $(3, n+1)$. □

Finally, we note that while Protocol 1 will be quantum-safe assuming the cipher $E$ is such, the security of Pohlig-Hellman and SRA both rest on the hardness of the discrete logarithm problem, hence, Protocol 1 will not be quantum-safe in practice. In the next section, we shall introduce a different third-party PSI protocol that achieves security against quantum adversaries.

## 4. A Quantum-Safe Third-Party PSI Scheme

In this section, we present a different third-party PSI scheme, which relies internally on the use of a key agreement protocol. Our work builds upon that of Rosulek and Trieu [21] which had incorporated techniques due to Cho, Dachman-Soled, and Jarecki [4]. Our scheme is quantum-safe with the use of a post-quantum key agreement protocol.

Suppose $P_1$ and $P_2$ have sets $S_1, S_2 \subseteq \{0,1\}^\ell$, each of size $n$. Let $\lambda > 0$ be the security parameter. We identify $\{0,1\}^\ell$ with a subset $S$ of a finite field $\mathbb{F}$ with $|\mathbb{F}| \geq 2^{\ell+\lambda+2\log n}$. Let $S_1 = \{s_1, \ldots, s_n\}$ and $S_2 = \{t_1, \ldots, t_n\}$. We fix:

- a 2-round key agreement protocol $\mathsf{KA}$ (see Figure 2) with space of randomness $\mathsf{KA}.\mathcal{R}$, message space $\mathsf{KA}.\mathcal{M} = \mathbb{F}$ and key space $\mathsf{KA}.\mathcal{K} = \mathbb{F}$,
- an ideal permutation $\Pi : \mathbb{F} \to \mathbb{F}$.

(1) $P_1$ picks $a \leftarrow \mathsf{KA}.\mathcal{R}$, and sends $m_1 = \mathsf{KA}.\mathsf{msg}_1(a)$ to $P_2$.
(2) $P_2$ picks $b \leftarrow \mathsf{KA}.\mathcal{R}$, and sends $m_2 = \mathsf{KA}.\mathsf{msg}_2(b, m_1)$ to $P_1$.
(3) $P_1$ and $P_2$ output $\mathsf{KA}.\mathsf{key}_1(a, m_2)$ and $\mathsf{KA}.\mathsf{key}_2(b, m_1)$ respectively.

Figure 2. A 2-round key agreement protocol between $P_1$ and $P_2$

For two probability distributions $X$ and $Y$ (each indexed by a security parameter), we write $X \approx Y$ to denote that $X$ and $Y$ are computationally indistinguishable. The key agreement protocol $\mathsf{KA}$ should satisfy the following three properties:

**Property 1.** *A 2-round key agreement protocol* $\mathsf{KA}$ *is correct if*

$$\mathsf{KA.key}_1(a, \mathsf{KA.msg}_2(b, \mathsf{KA.msg}_1(a))) = \mathsf{KA.key}_2(b, \mathsf{KA.msg}_1(a))$$

*for all* $a, b \in \mathsf{KA}.\mathcal{R}$.

**Property 2.** *A 2-round key agreement protocol* $\mathsf{KA}$ *has pseudorandom second messages if*

$$\{(a, \mathsf{KA.msg}_2(b, m_1))\}_{b \leftarrow \mathsf{KA}.\mathcal{R}} \approx \{(a, m_2)\}_{m_2 \leftarrow \mathsf{KA}.\mathcal{M}}$$

*for all* $a \in \mathsf{KA}.\mathcal{R}$, $m_1 = \mathsf{KA.msg}_1(a)$.

**Property 3.** *A 2-round key agreement protocol* $\mathsf{KA}$ *has pseudorandom keys if*

$$\{\mathsf{KA.key}_2(b, \mathsf{KA.msg}_1(a))\}_{b \leftarrow \mathsf{KA}.\mathcal{R}} \approx \{k\}_{k \leftarrow \mathsf{KA}.\mathcal{K}}$$

*for all* $a \in \mathsf{KA}.\mathcal{R}$.

Our protocol works as follows:

(1) $P_1$ picks a random $a \leftarrow \mathsf{KA}.\mathcal{R}$.
(2) $P_1$ sends $m = \mathsf{KA.msg}_1(a)$ to $P_2$.
(3) For each $i \in [n]$, $P_2$ picks a random $b_i \leftarrow \mathsf{KA}.\mathcal{R}$ and let $m'_i = \mathsf{KA.msg}_2(b_i, m)$ and $f_i = \Pi^{-1}(m'_i)$.
(4) $P_2$ computes the unique polynomial $p$ of degree $\leq n-1$ such that $p(t_i) = f_i$ for all $i \in [n]$, and sends $p$ to $P_1$.
(5) For each $i \in [n]$, $P_1$ computes $k_i = \mathsf{KA.key}_1(a, \Pi(p(s_i)))$.
(6) $P_1$ shuffles $K = \{k_1, \ldots, k_n\}$ and sends $K$ to $Q$.
(7) $P_2$ computes the unique polynomial $q$ of degree $\leq n-1$ such that $q(t_i) = \mathsf{KA.key}_2(b_i, m)$ for all $i \in [n]$, and sends $q$ to $Q$.
(8) For each $i \in [n]$, $Q$ computes all solutions $t$ to the equation $q(T) = k_i$ with $t \in S$, and outputs $\{t \in S : q(t) = k_i \text{ for some } i\}$.

PROTOCOL 2. A semi-honest third-party PSI protocol

From the above description, it is easily seen that the total amount of communication required by the protocol is $(3n+1)(\ell + \lambda + 2\log n)$ bits. When $\log n$ is small compared to $\ell$, this gives us a significant improvement over Protocol 1. Asymptotically, Protocol 2 incurs less communication cost for $\ell = \omega(\log n)$.

Figure 3 illustrate concrete values for which the improvements in communication overhead are attained as represented by the upper shaded portion of the curve and beyond, where $\lambda = 40$ is a standard statistical security parameter.

As a consequence, Protocol 2 has a lower communication cost compared to Protocol 1 for all practical values of $n$ for element length of at least 128 bits. The reduction in communication becomes greater as the element length increases. This is especially relevant for elements comprising of moderate to long length strings such as DNA sequences.

### 4.1. **Correctness and security.**

**Proposition 6.** *Assume* $\mathsf{KA}$ *satisfies Properties 1 and 3, and* $\Pi$ *is an ideal permutation. Then Protocol 2 is correct except with negligible probability.*

*Proof.* Protocol 2 outputs $S_1 \cap S_2$ unless

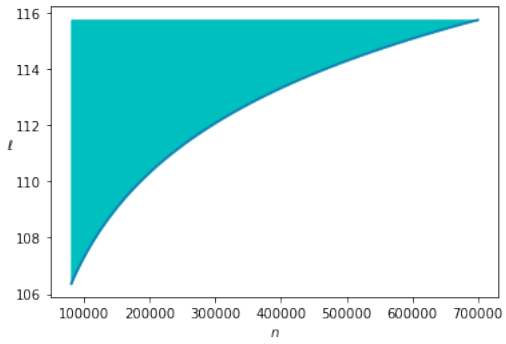


FIGURE 3. Region representing the communication cost improvement of Protocol 2

(i) for some $i \in [n]$ and $t_j \in S_2$ such that $t_j \neq s_i$, $k_i = \mathsf{KA.key}_2(b_j, m)$ where $b_j \in \mathsf{KA}.\mathcal{R}$ is the randomness corresponding to $t_j$, or
(ii) for some $i \in [n]$ such that $s_i \neq t_j$ for all $j \in [n]$, $q(T) = k_i$ has a solution $t \in S$, or
(iii) for some $i \in [n]$ such that $s_i = t_j$ for some $j \in [n]$, $q(T) = k_i$ has a solution $t \in S$ with $t \neq t_j$.

By Property 3 of KA, for fixed $i, j \in [n]$ such that $t_j \neq s_i$, the probability that $k_i = \mathsf{KA.key}_2(t_j, m)$ is negligibly close to $1/|\mathsf{KA}.\mathcal{K}|$. Taking the union bound over $i, j \in [n]$, we see that the probability that (i) holds is $\leq n^2/|\mathsf{KA}.\mathcal{K}| + \eta(\lambda) = 2^{-\ell-\lambda} + \eta(\lambda)$, where $\eta(\lambda)$ is a negligible function of $\lambda$.

Now assume (i) does not occur. Since outputs of KA are indistinguishable from uniformly random, $q$ is indistinguishable from a random polynomial of degree $\leq n-1$ in $\mathbb{F}[X]$.

For (ii), we note that if $s_i \in S_1$ does not correspond to any $t_j \in S_2$, then the roots of $q(T) = k_i = \mathsf{KA.key}_1(a, \Pi(p(s_i))$ are uniformly random in $\mathbb{F}$. The probability that a root lies in $S$ is $|S|/|\mathbb{F}| \leq 2^{-\lambda-2\log n}$, thus the probability that one or more of the roots of $q(T) = k_i$ lie in $S$ is $\leq (n-1)2^{-\lambda-2\log n}$.

For (iii), since $s_i = t_j$ for some $j$, $q(T) = k_i$ has the solution $t_j$. Now, $(q(T) - k_i)/(T - t_j)$ is indistinguishable from a uniformly random polynomial of degree $\leq n-2$, so the probability that one or more roots of $(q(T) - k_i)/(T - t_j)$ lie in $S$ is $\leq (n-2)2^{-\lambda-2\log n}$.

By the union bound again, Protocol 2 gives the correct output except with probability $\leq 2^{-\ell-\lambda} + \eta(\lambda) + n(n-1)2^{-\lambda-2\log n} < 2^{-\lambda+1} + \eta(\lambda)$, i.e. the protocol is correct except with negligible probability. □

We adapt the proofs of Lemmas 11 and 12 in [21] to prove that Protocol 2 is secure against semi-honest adversaries.

**Proposition 7.** *Assume* KA *satisfies Property* 2 *and* $\Pi$ *is an ideal permutation. Then Protocol 2 is secure against a semi-honest* $P_1$.

*Proof.* It suffices to show how to simulate $p$. Since KA satisfies Property 2, changing $m'_i = \mathsf{KA.msg}_2(b_i, m)$ to $m'_i \leftarrow \mathsf{KA}.\mathcal{M}$ cannot be distinguished by $P_1$. Then the $f_i$'s become independently and uniformly distributed, thus $p$ is simply a uniformly chosen polynomial of degree $\leq n-1$. □

**Proposition 8.** *Protocol 2 is secure against a semi-honest* $P_2$.

*Proof.* This is clear since the only protocol message received by $P_2$ is $m$, which does not depend on $S_1$. □

**Proposition 9.** *Assume* KA *satisfies Properties 1, 2 and 3, and that* $\Pi$ *is an ideal permutation. Then Protocol 2 is secure against a semi-honest* $Q$*.*

*Proof. Hybrid 0*: The real interaction.

*Hybrid 1*: We abort if there exists $s^* \in S_1 \setminus S_2$ and $t^* \in S_2$ such that $p(s^*) = p(t^*)$. Since $p$ is indistinguishable from a uniformly chosen polynomial of degree $\leq n-1$, the probability of abort is $\leq n^2/|\mathbb{F}| < 2^{-\lambda}$ by the union bound. Since the probability of abort is negligible, this hybrid is indistinguishable from Hybrid 0.

*Hybrid 2*: We modify how the ideal permutation $\Pi$ is simulated. For each $s_i \in S_1 \setminus S_2$, we know there has been no query to $\Pi$ at $p(s_i)$ in steps 1 to 4. This hybrid chooses $r_i \leftarrow \mathsf{KA}.\mathcal{R}$, and sets $\Pi(p(s_i)) = \mathsf{KA}.\mathsf{msg}_2(r_i, m)$. This hybrid is indistinguishable from Hybrid 1 since $\mathsf{KA}.\mathsf{msg}_2(r, m)$ is indistinguishable from uniformly random by Property 2.

*Hybrid 3*: We change how the $k_i$ values are computed. We set $k_i = \mathsf{KA}.\mathsf{key}_2(b_j, m)$ if $s_i = t_j$ for some $t_j \in S_2$, else $k_i = \mathsf{KA}.\mathsf{key}_2(r_i, m)$. Hybrids 2 and 3 are identical by Property 1 of KA.

*Hybrid* $(4, h)$ *for* $h \in [n+1]$: We again change how the $k_i$ values are computed. We set:

$$k_i = \begin{cases} \mathsf{KA}.\mathsf{key}_2(b_j, m) & \text{if } s_i = t_j \text{ for some } t_j \in S_2, \\ k_i' \text{ where } k_i' \leftarrow \mathsf{KA}.\mathcal{K} & \text{if } s_i \neq t_j \text{ for all } t_j \in S_2 \text{ and } i < h, \\ \mathsf{KA}.\mathsf{key}_2(r_i, m) & \text{otherwise.} \end{cases}$$

Hybrid $(4, 1)$ is identical to Hybrid 3. By Property 3 of KA, Hybrid $(4, h)$ is indistinguishable from Hybrid $(4, h+1)$ for each $h \in [n]$. Hence, Hybrid $(4, n+1)$ is indistinguishable from Hybrid 3.

*Hybrid* $(5, h)$ *for* $h \in [n+1]$: We let $q$ be the unique polynomial of degree $\leq n-1$ such that

$$q(t_i) = \begin{cases} \mathsf{KA}.\mathsf{key}_2(b_i, m) & \text{if } t_i = s_j \text{ for some } j \in [n] \text{ or } i \geq h, \\ q_i \text{ where } q_i \leftarrow \mathsf{KA}.\mathcal{K} & \text{otherwise.} \end{cases}$$

Hybrid $(5, 1)$ is identical to Hybrid $(4, n+1)$, and Hybrid $(5, h)$ is indistinguishable from Hybrid $(5, h+1)$ for each $h \in [n]$, again, by Property 3 of KA. This shows that Hybrid $(5, n+1)$ is indistinguishable from Hybrid $(4, n+1)$.

*Simulator*: We simulate

$$K = \{\mathsf{KA}.\mathsf{key}_2(b_j, m) : t_j \in S_1 \cap S_2\} \cup \{k_1'', \ldots, k_{|S_1 \setminus S_2|}''\}$$

where each $k_i'' \leftarrow \mathsf{KA}.\mathcal{K}$, and

$$q \leftarrow \{\theta \in \mathbb{F}[X] : \deg(\theta) \leq n-1,\ \theta(t_j) = \mathsf{KA}.\mathsf{key}_2(b_j, m) \text{ for } t_j \in S_1 \cap S_2\}.$$

This interaction is identically distributed to Hybrid $(5, n+1)$. □

4.2. **Implementation details.** In the above protocol, steps 4 and 7 involve polynomial interpolation, while step 8 involves finding the roots of polynomials over a finite field. Let us briefly discuss how these steps can be implemented efficiently.

One straightforward way to implement step 8 is to apply the Cantor-Zassenhaus algorithm [3] a total of $n$ times, but that gives us a total complexity of $O(n^4 + n^3 \log |\mathbb{F}|)$.

Instead, we can evaluate the polynomial $q$ at all points of $S \subset \mathbb{F}$. Using a divide-and-conquer strategy with the Schönhage-Strassen algorithm [22], a multipoint evaluation of a degree $n$ polynomial at $n$ distinct points has complexity $O(n(\log n)^2 \log\log n)$. Hence, evaluating $q$ at all points of $S$ has a total complexity of $O(|S|(\log n)^2(\log\log n))$.

Roughly speaking, the second method is superior if $n > |S|^{\frac{1}{4}}$, hence the appropriate method should be chosen depending on the values of $n$ and $|S|$.

For an actual implementation, we note that both of the above methods are easily parallelizable. Furthermore, for small to moderate values of $n$, Berlekamp's algorithm [2] will likely be faster than the Cantor-Zassenhaus algorithm in practice [25].

For steps 4 and 7, a naive implementation of Lagrange interpolation has a complexity of $O(n^2)$. However, polynomial interpolation can be reduced to the problems of polynomial multiplication and multipoint polynomial evaluation [8]. Hence, by using fast multiplication and multipoint evaluation algorithms, we can perform steps 4 and 7 using $O(n(\log n)^2 \log\log n)$ operations.

### 4.3. **Extensions to the main protocol.**

4.3.1. *Offloading computations to an untrusted server.* From the above discussion, we note that step 8 is the most computationally intensive part of the entire protocol. Hence, it might be beneficial to allow an untrusted party $R$ (such as a cloud service) to perform step 8. This can be achieved as follows.

Let $E : \mathcal{K} \times \{0,1\}^\ell \to \{0,1\}^\ell$ be a pseudorandom permutation (PRP). At the start of the protocol, $P_1$, $P_2$ and $Q$ first agree on a random key $k \leftarrow \mathcal{K}$. Then, $P_1$ and $P_2$ apply $E$ to the sets $S_1$ and $S_2$ to obtain $E_k(S_1)$ and $E_k(S_2)$ respectively. The parties now follow steps 1 to 5 of Protocol 2 with $S_1$ replaced by $E_k(S_1)$ and with $S_2$ replaced by $E_k(S_2)$. In steps 6 and 7, $P_1$ and $P_2$ send $K$ and $q$ to $R$ instead of $Q$. The untrusted party $R$ can then perform step 8, before forwarding the result, which is equal to $E_k(S_1 \cap S_2)$, to $Q$. Finally, $Q$ applies $E^{-1}$ to obtain the desired intersection.

Note that this protocol does leak the size of the intersection to $R$, but is easily seen to be secure otherwise.

4.3.2. *Third-party PSI with intersection-sum.* Another possible extension to our main protocol is to augment it so that the sum of auxiliary data over the intersection set can be computed privately. Assume that this auxiliary data is held by one of the parties, say $P_1$. To be precise, we suppose that $P_1$ possesses both a set $S_1$ and, for each element $s \in S_1$, a corresponding value $v_s$. By scaling the values of $v_s$, we may assume that all the $v_s$ values are integral. Further, assume that the $v_s$ values are bounded (in absolute value) by $B$ for some $B > 0$.

We will apply the additively homomorphic Paillier cryptosystem [17] with primes $p$ and $q$ such that $N = pq > 2nB$. Let $E : \mathbb{Z}_N \to \mathbb{Z}_{N^2}$ and $D : \mathbb{Z}_{N^2} \to \mathbb{Z}_N$ be the encryption and decryption functions respectively.

Our augmented protocol is as follows:

(1) $P_1$ picks a random $a \leftarrow \mathsf{KA}.\mathcal{R}$.
(2) $P_1$ sends $m = \mathsf{KA}.\mathsf{msg}_1(a)$ to $P_2$.
(3) For each $i \in [n]$, $P_2$ picks a random $b_i \leftarrow \mathsf{KA}.\mathcal{R}$ and let $m'_i = \mathsf{KA}.\mathsf{msg}_2(b_i, m)$ and $f_i = \Pi^{-1}(m'_i)$.

(4) $P_2$ computes the unique polynomial $p$ of degree $\leq n-1$ such that $p(t_i) = f_i$ for all $i \in [n]$, and sends $p$ to $P_1$.
(5) For each $i \in [n]$, $P_1$ computes $k_i = \mathsf{KA.key}_1(a, \Pi(p(s_i)))$ and $w_i = E(v_i)$.
(6) $P_1$ shuffles $K = \{(k_1, w_1), \ldots, (k_n, w_n)\}$ and sends $K$ to $Q$.
(7) $P_2$ computes the unique polynomial $q$ of degree $\leq n-1$ such that $q(t_i) = \mathsf{KA.key}_2(b_i, m)$ for all $i \in [n]$, and sends $q$ to $Q$.
(8) For each $i \in [n]$, $Q$ computes all solutions $t$ to the equation $q(T) = k_i$ with $t \in S$.
(9) $Q$ computes the product $w$ of all $w_i$ for which $q(T) = k_i$ has a solution with $t \in S$, chooses a random $e \in \mathbb{Z}_N^*$ and sends $\sigma = w^e$ to $P_1$.
(10) $P_1$ decrypts $\sigma$ and sends $D(\sigma)$ to $Q$.
(11) $Q$ outputs $\{t \in S : q(t) = k_i \text{ for some } i\}$ and the sum $e^{-1}D(\sigma)$ (interpreted as an integer between $-nB$ and $nB$).

Protocol 3. A semi-honest third-party PSI protocol with intersection-sum

To see the correctness of this modified protocol, recall that the Paillier cryptosystem satisfies the property that $D(E(v)E(v')) = v + v'$. Hence, $w$ is an encryption of $\sum_{s \in S_1 \cap S_2} w_s$ and $\sigma$ is an encryption of $e\big(\sum_{s \in S_1 \cap S_2} w_s\big)$.

It is also possible to hide the actual intersection elements from $Q$ by having $P_1$ and $P_2$ apply a PRP to the elements of $S_1$ and $S_2$ before running Protocol 3, thus achieving the functionality of third-party private intersection-sum with cardinality.

## 5. Conclusion

In this paper, we introduce the concept of a third-party PSI scheme, which is motivated by many existing use cases, and present two protocols to achieve this functionality with low communication overhead. The first protocol provides a baseline for such a scheme, while the second protocol enjoys several advantages over the first, namely, it incurs a lower communication overhead for element strings of moderate to large lengths, achieves quantum-safe security with the use of a post-quantum key agreement protocol, and is likely to be easily adaptable to be secure against malicious adversaries without incurring additional communication overhead.

## References

[1] Pierre Baldi, Roberta Baronio, Emiliano De Cristofaro, Paolo Gasti, and Gene Tsudik. 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)*, pages 691–702. ACM, 2011.
[2] Elwyn R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, New York, 1968.
[3] David G. Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587–592, 1981.
[4] Chongwon Cho, Dana Dachman-Soled, and Stanisław Jarecki. Efficient concurrent covert computation of string equality and set intersection. In *Topics in Cryptology-CT-RSA 2016: The Cryptographers' Track at the RSA Conference 2016*, pages 164–179. Springer International Publishing, 2016.
[5] Thai Duong, Duong Hieu Phan, and Ni Trieu. Catalic: delegated PSI cardinality with applications to contact tracing. In *International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2020)*, pages 870–899. Springer, 2020.
[6] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology - EUROCRYPT 2004*, pages 1–19. Springer, Berlin, Heidelberg, 2004.

[7] Per Hallgren, Claudio Orlandi, and Andrei Sabelfeld. Privatepool: Privacy-preserving ridesharing. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 276–291. IEEE, 2017.

[8] Ellis Horowitz. A fast method for interpolation using preconditioning. *Information Processing Letters*, pages 157–163, 1972.

[9] Bernardo A. Huberman, Matt Franklin, and Tad Hogg. Enhancing privacy and trust in electronic communities. In *Proceedings of the 1999 ACM CONFERENCE ON ELECTRONIC COMMERCE*. ACM, 1999.

[10] Mihaela Ion, Ben Kreuter, Ahmet Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, Mariana Raykova, David Shanahan, and Moti Yung. On deploying secure computing: Private intersection-sum-with-cardinality. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 370–389. IEEE, 2020.

[11] 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, Berlin, Heidelberg, 2014.

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

[13] Phi Hung Le, Samuel Ranellucci, and S. Dov Gordon. Two-party private set intersection with an untrusted third party. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS'19)*, pages 2403–2420. ACM, 2019.

[14] Catherine Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *Proceedings of the 1986 IEEE Symposium on Security and Privacy*, pages 134—134. IEEE, 1986.

[15] Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov. BotGrep: Finding P2P bots with structured graph analysis. In *19th USENIX Security Symposium (USENIX Security 10)*, pages 95—110, 2010.

[16] Arvind Narayanan, Narendran Thiagarajan, Mugdha Lakhani, Michael Hamburg, and Dan Boneh. Location privacy via private proximity testing. In *Network and Distributed Security Symposium (NDSS'11)*. The Internet Society, 2011.

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

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

[19] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, 24:106–110, 1978.

[20] Srinivasan Raghuraman and Peter Rindal. Blazing fast PSI from improved OKVS and subfield VOLE. In *ACM CCS'21*, 2021.

[21] Mike Rosulek and Ni Trieu. Compact and malicious private set intersection for small sets. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS'21)*, pages 1166—1181. ACM, 2021.

[22] A. Schönhage and V. Strassen. Schnelle multiplikation großer zahlen. *Computing*, pages 1436–5057, 1971.

[23] Adi Shamir. On the power of commutativity in cryptography. In *Proceedings of the 1980 International Colloquium on Automata, Languages, and Programming*, pages 582—595. Springer, Heidelberg, 1980.

[24] Adi Shamir, Ronald L. Rivest, and Leonard M. Adleman. Mental poker. In *The Mathematical Gardner*, pages 37–43. Springer US, 1981.

[25] Victor Shoup. Factoring polynomials over finite fields: Asymptotic complexity vs. reality. In *Proceedings of the IMACS Symposium*, 1993.

Foo Yee Yeo, Seagate Technology, Singapore
*Email address*: fooyee.yeo@seagate.com

Jason H. M. Ying, Seagate Technology, Singapore
*Email address*: jasonhweiming.ying@seagate.com