Differentially Private Set Intersection: Refined Definition and Concretely Efficient Constructions

Anonymous Author

No Institute Given

Abstract. Private Set Intersection (PSI) enables a sender and a receiver to jointly compute the intersection of their sets without disclosing non-trivial information about other items. However, depending on the context of joint data analysis, information derived from the items in the intersection may also be considered sensitive. To protect such sensitive information, prior work proposed Differentially private PSI (DPSI), which can be instantiated with circuit-PSI using Fully Homomorphic Encryption. Although asymptotically efficient, its concrete performance is sub-optimal compared with the practical state-of-the-art (SOTA) circuit-PSI.
In this paper, we propose two generic DPSI constructions with provable security and privacy. We revisit the DPSI definition and identify the critical criteria for selecting essential PSI-related tools. Then, we present two generic DPSI constructions. The first construction allows us to achieve provable privacy and efficiency by integrating any circuit-PSI with the randomized response mechanism. By plugging the SOTA circuit-PSI protocol, we obtain a DPSI protocol with concrete performance enhancement. The second construction offers a more efficient DPSI alternative by using multi-query Reverse Private Membership Test (mqRPMT) at the price of intersection size leakage, but such leakages can be bounded with differential privacy by padding random dummy items in input sets. We conduct comprehensive experiments with various instantiations. The experiments show that our instantiations significantly outperform the existing DPSI construction: 2.5-22.6$\times$ more communication-efficient and up to 110.5-151.8$\times$ faster. Our work also shows a new application for mqRPMT besides obtaining Private Set Operation (PSO).

Keywords: private set intersection, differential privacy.

1 Introduction

Private Set Intersection (PSI) allows a sender $\mathcal{P}_1$ and a receiver $\mathcal{P}_2$, holding sets X and Y respectively, to identify the intersection $X \cap Y$ without revealing any information about items not in the intersection. PSI helps to achieve data minimization as no data is shared beyond what each party has in common [27]. As one of the most well-studied secure multi-party computation (MPC) protocols, PSI has been widely recognized as a valuable tool in numerous applications.

Although PSI preserves privacy for items outside the intersection, depending on the context, the intersection itself is also sensitive. Consider the following Example. A card-payment company (acted as $\mathcal{P}_1$) wants to present advertisements to some individuals on an advertising platform (acted as $\mathcal{P}_2$). To this end, $\mathcal{P}_1$ and $\mathcal{P}_2$ run PSI to ensure that $\mathcal{P}_2$ can only identify individuals who are common targets for advertisements known by both parties. In this way, individuals outside the intersection are kept unknown for $\mathcal{P}_2$, preventing information leakage. However, all users in the intersection have an implicit feature that they are $\mathcal{P}_1$'s users, and this information is exposed to $\mathcal{P}_2$ directly. This result can be seen as a data privacy problem when some users might be reluctant to disclose such information to the advertisement company. It can also be seen as a data secrecy problem because it allows the advertising platform to help another rival card-payment company, e.g., $\mathcal{P}_1'$, win customers and gain an advantage for $\mathcal{P}_1'$ over $\mathcal{P}_1$ directly from the intersection. Furthermore, since the ideal functionality of PSI [33,38,43] requires outputting the correct intersection, this inherent privacy problem cannot be handled by conventional PSI alone.

Through the lens of privacy, one can view PSI as a procedure where the sender $\mathcal{P}_1$ (holding the input set X) tells the receiver $\mathcal{P}_2$ (holding the input set Y) a Boolean vector of size $|Y|$, each bit of which indicates whether the corresponding item in Y is in the intersection, leaking one-bit information for each item in Y. In this way, the problem can be formalized in the Differential Privacy (DP) paradigm. As a measurement of privacy leakage and a set of tools to mitigate such leakages, DP has been increasingly regarded as a *de facto* standard for protecting individual privacy that has been studied extensively in both academia [15, 23, 24, 34, 35, 46, 53, 54] and industry [4, 13, 16, 22, 36]. A DP mechanism guarantees that the presence or absence of any particular individual's record has a quantifiable impact on the likelihood that a particular response is returned to a query. Consequently, an adversary's probability of success in inferring any individual's record value or whether the record is present can be strictly bounded.

It is natural to combine DP with PSI to further protect the Boolean information corresponding to each item. In the pioneering research, Kacsmar et al. [31] defined the notion of Differentially private Set Intersection (DPSI) and constructed a DPSI protocol using Fully Homomorphic Encryption (FHE) [6,18]. Recently, Mahdavi et al. [37] showed that the construction is essentially a circuit-PSI [9, 44, 48] (equipped with a DP mechanism). In circuit-PSI, $\mathcal{P}_1$ and $\mathcal{P}_2$ receive Boolean shares of the intersection instead of learning the intersection in the clear. These shares can be used for subsequent secure computations. By obliviously permuting the Boolean shares then revealing to $\mathcal{P}_2$, we enable $\mathcal{P}_2$ to learn an intersection while maintaining a DP guarantee.

Although being asymptotically efficient, the concrete performance of [31] shows its sub-optimal performance. Even for small set sizes $n = m = 2^{16}$ with $n = |X|$ and $m = |Y|$, the running time reaches about 8 minutes, which is inefficient compared to typical PSI-based applications that only require several seconds for set sizes $n = m = 2^{20}$ [3, 49]. In contrast, circuit-PSI is the intrinsic building block for the DPSI construction of [31], and the state-of-the-art (SOTA) circuit-PSI protocol only needs less than one minute for both parties' set sizes $n = m = 2^{20}$ [9, 44, 48]. Thus, it is necessary to investigate and close the gap between DPSI and existing PSI-related tools for optimal efficiency. Since PSI is a very active research area, and more PSI-related tools are being proposed, a deeper understanding can lead to generic DPSI constructions from a broader range of future tools.

1.1 Our Contributions

In this paper, we revisit the existing definition of DPSI [31] and formally propose its ideal functionality $\mathcal{F}_{\mathsf{DPSI}}$. By considering the privacy and security properties formalized in $\mathcal{F}_{\mathsf{DPSI}}$, we identify the critical components and procedures in DPSI that create spaces for enhancement and different trade-offs. These observations are illustrated in Fig. 1. Intuitively, for a sender $\mathcal{P}_1$ who holds X and a receiver $\mathcal{P}_2$ who holds Y, any PSI-related tool that enables the sender to perturb the membership information about Y *without knowing the correct intersection* can be used to construct DPSI. We suggest two such components: circuit-PSI [9,44,48] and multi-query Reverse Private Membership Test (mqRPMT) [10,55].

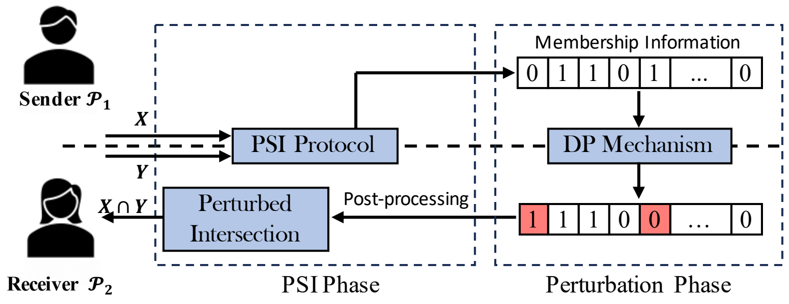

Fig. 1: The critical components and procedures in DPSI

Following the insight, we propose two generic constructions for DPSI. The first one is called circuit-DPSI, and its core component is circuit-PSI. In this construction, the parties first get a shared Boolean vector of the intersection by *executing any circuit-PSI*. Then, $\mathcal{P}_1$ obliviously perturbs his shared Boolean vector via a DP mechanism. $\mathcal{P}_2$ finally gets the DP intersection by XORing the $\mathcal{P}_1$'s perturbed Boolean vector with its own share. We note that the pioneering DPSI construction [31] follows this routine. By plugging in a more efficient circuit-PSI instance, we obtain DPSI with improved performance.

The second construction, mqRPMT-DPSI, leads to an even more efficient DPSI. As the name suggests, it leverages a recently proposed building block named multi-query Reverse Private Membership Test (mqRPMT) [10, 55]. In mqRPMT, a sender holding a vector $X = (x_1, \ldots, x_n)$ interacts with a receiver holding a set Y, and eventually, the sender learns only a Boolean vector $(e_1, \ldots, e_n)$ indicating whether $x_i \in Y$ without learning any other items of Y, while the receiver learns nothing. mqRPMT were previously used to construct Private Set Operation (PSO) [10,55]. In our construction, two parties first invoke mqRPMT with their input sets X and Y, and $\mathcal{P}_1$ gets the indication Boolean vector containing the membership information. Then, $\mathcal{P}_1$ perturbs the indication Boolean vector via a DP mechanism and sends it to $\mathcal{P}_2$, who gets the final DP intersection. Thanks to the efficiency of the SOTA mqRPMT protocol [10], we can construct DPSI with near-optimal concrete performance.

However, there is a remaining challenge. The indication Boolean vector output by mqRPMT unavoidably leaks the cardinality of the intersection set to the sender, which may lead to attacks [21, 30] that de-anonymize specific items in the input sets. We propose an additional procedure in mqRPMT-DPSI to alleviate this privacy leakage. In a nutshell, we ask both parties to insert dummy items into their sets via a Random Dummy Item Padding mechanism ahead of time and take the padded X' and Y' as mqRPMT inputs. Random Dummy Item Padding guarantees that the leaked cardinality satisfies (ϵ, δ)-DP. By plugging in the SOTA mqRMPT instance [10], the final DPSI remains efficient with bounded cardinality leakage regarding DP. As an independent interest, our work shows a new use case for mqRPMT besides PSO.

1.2 Organizations

The remainder of this paper is organized as follows. Section 2 reviews the related works. Section 3 provides the preliminaries. Section 4 revisits the definition of DPSI and formalizes the new DPSI ideal functionality. Section 5 and Section 6 present details of the circuit-DPSI and the mqRPMT-DPSI construction, respectively. Section 7 summarizes our experimental results. Finally, Section 8 gives conclusions and discusses possible future work.

2 Related Works

PSI. The idea of PSI dates back to the 1980s, and the initial PSI protocol was constructed from the Diffie-Hellman (DH) key agreement [38]. Since then, constructions based on Public Key Cryptography [7, 12, 28], Oblivious Transfers (OT) [29], FHE [51] and Garbled Circuits (GC) [52] have been proposed. These proposals offer different trade-offs between computation and communication overheads. Among them, OT-based protocols are usually faster than other variants. The first OT-based PSI protocol was proposed by Pinkas et al. [43], which incorporates OT extension [1,29,32] as the technical core of their protocol. Kolesnikov et al. [33] proposed the fastest OT-based PSI protocol by designing an efficient primitive called batched Oblivious Pseudo-Random Functions (batched OPRF). However, all these OT-based PSI protocols offer improved efficiency at the expense of increased communication. To further reduce the communication overheads, Pinkas et al. [41] proposed a PSI protocol based on sparse OT extension. Their scheme achieves a better balance between computation costs and communication overheads. Moreover, Pinkas et al. [42] extended their work with a new data structure called Oblivious Key-Value Stores (OKVS) and obtained more efficient PSI protocols under the stronger malicious security model. Shortly afterwards, Rindal et al. [48] proposed an even more computationally efficient

PSI protocol by combining the OKVS with Vector Oblivious Linear Evaluation (VOLE) [5, 11]. Garimella et al. [17] introduced a way of compressing OKVS, cutting down the communication overhead of [42]. By employing the improved OKVS in [48], Raghuraman et al. [45] achieved additional cost reductions in communication complexity. Chen et al. [10] presented a unified framework by taking mqRPMT [55] as the primitive and using the framework to obtain PSI and more general PSO.

Circuit-PSI. The syntax of PSI mentioned above defines the intersection as the output. To support arbitrary secure computations over the intersection without exposing it to either party, Huang et al. [26] introduced the notion of circuit-PSI. The work introduced several Boolean circuits and evaluated them using Yao's GC [52] over the shares. The complexity of this protocol is $O(n \log^2 n)$. By invoking the Oblivious Programmable Pseudo-Random Function (OPPRF), Pinkas et al. [44] proposed the first circuit-PSI protocol that achieves linear complexity. Chandran et al. [9] constructed a new circuit-PSI protocol with linear complexity by invoking Relaxed Batch Oblivious Programmable Pseudo-Random Functions (RB-OPPRF).

DP in Secure Computation. Existing works employ DP as a complementary technology for preventing information leakage not captured in conventional secure computation models. In order to prevent side-channel attacks while answering federated queries, Narayan et al. [39] proposed a primitive for obtaining PSI cardinality (PSI-CA) in a differentially private manner. Addressing the same problem, Bater et al. [2] introduced a private data federation called Shrinkwrap, which utilizes computational DP to minimize padding in intermediate query results. He et al. [25] formulated the problem of Differentially Private Record Linkage (DPRL) and developed corresponding protocols by employing a specific privacy definition called Output Constrained DP. In 2020, for the first time, Kacsmar et al. [31] attempted to construct differentially private PSI (DPSI) by combining FHE-based circuit-PSI with DP. However, the use of FHE results in sub-optimal efficiency in practice. Initiating from an alternative origin, Groce et al. [20] considered improving the performance of secure two-party computation protocols by leveraging DP and showed that if DP leakage is allowed, the cost of Rindal et al.s' [47] PSI protocol can be reduced by up to 63%.

3 Preliminaries

3.1 Notations and Security Model

The computational and statistical security parameters are denoted by κ and σ, respectively. We denote $[a, b]$ with $a, b \in \mathbb{N}$ and $a \leq b$ to the set $\{a, a+1, \ldots, b\}$ and $[b]$ as a shorthand for $[1, b]$. We denote the upper letter $A = (a_1, \ldots, a_n)$ as a vector with length $|A| = n$.

Two parties in DPSI are the sender $\mathcal{P}_1$ and the receiver $\mathcal{P}_2$. They have input sets $X = \{x_i\}_{i \in [n]} \in \mathcal{V}^n$ and $Y = \{y_j\}_{j \in [m]} \in \mathcal{V}^m$, respectively. All items in X

and Y are from the same item domain $\mathcal{V}$. Based on the context, we sometimes use the notation $X[i]$ or x_i to denote the i-th item in the set X.

In this work, we consider the semi-honest model with static corruption, where an adversary $\mathcal{A}$ tries to learn as much information as possible from protocol execution but cannot deviate from the protocol definitions. We assume that all cryptographic building blocks used in this work have been proven secure in the semi-honest model under the standard real-ideal paradigm [19].

3.2 Standard Differential Privacy Definition

Intuitively, DP requires the outputs of algorithms to be approximately the same if any individual's record in the input is added or removed. DP is formally defined as follows [14].

Definition 1 ((ϵ, δ)-Differential Privacy). *A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{F}$ is (ϵ, δ)-DP if for all neighboring inputs $D, D' \in \mathcal{D}$ such that D and D' differ by adding or removing a record, and all subsets $F \in \mathcal{F}$,*

$$\Pr[\mathcal{M}(D) \in F] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in F] + \delta$$

where the probability is taken over the randomness of $\mathcal{M}$.

In Definition 1, ϵ is the privacy budget, and δ can be treated as the failure probability of DP protection. We say that $\mathcal{M}$ satisfies ϵ-DP if $\delta = 0$. When obtaining multiple DP outputs, the sequential composition (Theorem 1) and parallel composition (Theorem 2) are used to measure the total privacy budget.

Theorem 1 (Sequential Composition [14]). *Given k number of (ϵ_i, δ_i)-DP mechanisms $\mathcal{M}_i$ that access the same input D with $1 \leq i \leq k$, the combination of their outputs satisfies (ϵ, δ)-DP, where $\epsilon = \sum_{i=1}^{k} \epsilon_i$ and $\delta = \sum_{i=1}^{k} \delta_i$.*

Theorem 2 (Parallel Composition [14]). *Given k number of (ϵ_i, δ_i)-DP mechanisms $\mathcal{M}_i$ that access disjoint inputs D_i with $1 \leq i \leq k$, the combination of their outputs satisfies (ϵ, δ)-DP, where $\epsilon = \max(\epsilon_1, \ldots, \epsilon_k)$ and $\delta = \max(\delta_1, \ldots, \delta_k)$.*

3.3 Cryptographic Primitives

Cuckoo Hashing. Cuckoo hashing [40] uses $d > 1$ universal hash function $h_1, \ldots, h_d : \{0,1\}^* \to [\beta]$ to map n items to $\beta > n$ bins in the hash table HT, where each bin is restricted to accommodate at most one item. It iteratively inserts a sequence of items $(e_1, \ldots, e_n)$ into HT as follows. Given the item e_i, if one of $HT[h_1(e_i)], ..., HT[h_d(e_i)]$ bins is empty, then insert e_i in the first empty bin. Otherwise, sample $i \in [d]$ uniformly at random, evict the item e_i' present in $HT[h_i(e_i)]$, place e_i in bin $HT[h_i(e_i)]$, and recursively try to insert the evicted item e_i' until the number of attempts reaches a certain threshold and then fails. In this way, the item e_i could be placed into one bin among $h_1(e_i), \ldots, h_d(e_i)$.

Oblivious Programmable Pseudo-Random Function. Let $F : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^\ell$ be a Pseudo-Random Function (PRF) that maps an input to a pseudo-random output under a PRF key. Programmable PRF (PPRF) is a variant of PRF that further allows to "program" some PRF outputs, i.e., produce assigned PRF outputs for some specific given inputs.

PPRF is usually defined for handling multiple PRF keys $K = (k_1, \ldots k_\beta)$ in a batch. An (ℓ, β)-PPRF consists of a pair of algorithms $\hat{F} = (Hint, F)$. Given a set of uniformly random and independent PRF keys $K = (k_1, \ldots, k_\beta)$, the disjoint input sets $(X_1, \ldots, X_\beta)$, and the target multi-sets $(T_1, \ldots, T_\beta)$ satisfying $|T_j| = |X_j|$ for all $j \in [\beta]$ and $T_j[i] \in \{0,1\}^\ell$ for all $i \in [|T_j|]$, $Hint(K, X, T)$ outputs $hint$. Later, given the key $k_j \in \{0,1\}^\kappa$, $hint$, and an input $x \in \{0,1\}^*$, $F(k_j, hint, x)$ outputs a PRF value $y \in \{0,1\}^\ell$. It holds that if $x = X_j[i]$ for some $i \in [|X_i|]$, then $y = T_j[i]$. Otherwise, y looks random.

Oblivious PPRF (OPPRF) is a two-party protocol that allows the sender and the receiver to obliviously run PPRF. In OPPRF, the sender takes inputs as the programmed input sets $X = (X_1, \ldots, X_\beta)$ and the target sets $T = (T_1, \ldots, T_\beta)$, while the receiver takes inputs as the batch queries $(q_1, \ldots, q_\beta)$. OPPRF samples PPRF keys $K = (k_1, \ldots, k_\beta)$ for an (ℓ, β)-PPRF to the sender, and gives $hint$ and the PPRF outputs $\{F(k_j, hint, q_j)\}_{j \in [\beta]}$ to the receiver.

Private Set Membership. Private Set Membership (PSM) [9] is a two-party protocol that takes input as a set $X = \{x_i\}_{i \in [n]} \in \mathcal{V}^n$ from the sender and an item $y \in \mathcal{V}$ from the receiver. Define the Boolean value a such that $a = 1$ if $y \in X$ and $a = 0$ otherwise. The PSM functionality outputs Boolean shares of a to the sender and the receiver, i.e., random bits $\langle a \rangle_1$ and $\langle a \rangle_2$ to both parties, respectively, such that $\langle a \rangle_1 \oplus \langle a \rangle_2 = a$.

Circuit-PSI. Circuit-PSI is a two-party protocol that allows the sender and receiver to securely compute arbitrary symmetric functions over the intersection of their private sets by utilizing the secure circuit evaluation. In circuit-PSI, both the sender and the receiver take their sets $X = \{x_1, ..., x_n\}$ and $Y = \{y_1, ..., y_m\}$ as inputs and receive the Boolean shares of the set intersection $Z = X \cap Y$. The ideal functionality of circuit-PSI is formally defined in Fig. 2.

Multi-query Reverse Private Membership Test. Multi-query Reverse Private Membership Test (mqRPMT) [10,55] is a two-party protocol that is initially used as a primitive for constructing PSO. In mqRPMT, a sender with a set X interacts with a receiver holding a set $Y = (y_1, \ldots, y_m)$, and eventually the sender learns only a Boolean vector $(e_1, \ldots, e_m)$ indicating whether $y_i \in X$ without learning the value of y_i, while the receiver learns nothing.[1] The functionality of mqRPMT is formally defined in Fig. 3. Among the protocols that instantiate mqRPMT, the cwPRF-based mqRPMT protocol is the most efficient one that achieves strict linear computation and communication complexity. More specifically, commutative weak PRF (cwPRF) is a family of keyed functions $\{F_{k_i} : \mathcal{K} \times \mathcal{D} \to \mathcal{D}\}$ that satisfy weak pseudo-randomness and commutative prop-

[1] For ease of description, we define mqRPMT in the reversed roles.

$\mathcal{F}_{\text{circuit-PSI}}$

Parameters: The sender's set size n. The receiver's set size m. $H_s : \mathcal{V}^m \to HT_2$, which on input a size-$m$ set outputs a hash table with size $\beta = O(m)$ and each element is assigned to a distinct bin of HT_2.
Input: The sender's set of items $X = \{x_i\}_{i\in[n]}$, and the receiver's set of items $Y = \{y_i\}_{i\in[m]}$.
Output: The sender gets a shared Boolean vector $\{\langle a_j\rangle_1\}_{j\in[\beta]}$. The receiver gets a hash table $HT_2 = H_s(Y)$ with size $\beta = O(m)$ storing items in Y, and a shared Boolean vector $\{\langle a_j\rangle_2\}_{j\in[\beta]}$, where $a_j = \langle a_j\rangle_1 \oplus \langle a_j\rangle_2$ indicates if the item in $HT_2[j]$ is in the intersection.

Fig. 2: Ideal functionality $\mathcal{F}_{\text{circuit-PSI}}$ for circuit-PSI

erty simultaneously, and commutative property means that for all $k_1, k_2 \in \mathcal{K}$ and $x \in \mathcal{D}$, a family of keyed functions satisfy $F_{k_1}(F_{k_2}(x)) = F_{k_2}(F_{k_1}(x))$.

One can obtain cwPRF using a finite group $\mathbb{G}$ of prime order p where the Decisional Diffie-Hellman (DDH) assumption holds, by setting $k_1, k_2 \in \mathbb{Z}_p$, selecting a (cryptographic) hash function $H : \{0,1\}^* \to \mathbb{G}$, and defining $F_{k_1}(x) = H(x)^{k_1}, F_{k_2}(x) = H(x)^{k_2}$. The finite group of points on an Elliptic Curve (EC) is a good candidate for $\mathbb{G}$.

$\mathcal{F}_{\text{mqRPMT}}$

Parameters: The sender's set size n. The receiver's set size m.
Input: The sender's set of items $X = \{x_i\}_{i\in[n]}$, and the receiver's set of items $Y = \{y_i\}_{i\in[m]}$.
Output: The sender gets a vector $(e_1, \ldots, e_m)$, where $e_i = 1$ if $y_i \in X$ and $e_i = 0$ otherwise. The receiver gets nothing.

Fig. 3: Ideal functionality $\mathcal{F}_{\text{mqRPMT}}$ for mqRPMT

4 Differentially Private Set Intersection

4.1 DPSI Definition

We first revisit the DPSI definition proposed by Kacsmar et al. [31] shown in Definition 1. The key to formalizing DPSI is precisely defining the neighboring datasets and the output range. PSI has two inputs: the set X from the sender and the set Y from the receiver. As the DP definition suggests, one must consider the output distribution for all neighboring datasets D' and D that differ in any single item. Kacsmar et al. [31] consider a one-sided PSI where the sender learns nothing about the receiver's input set Y, and only the receiver learns a DP version of the intersection. This is reasonable. In the DP context, one can think of the sender's input X being the database to be queried, and the receiver asking a query with the input data Y. That is, the receiver's query is of the form:

Which items in Y are also in your database X?

Since PSI only allows the receiver to get the intersection $X \cap Y$, PSI only reveals X's information to the receiver. Therefore, we only need to consider privacy from the sender's input X by defining the neighboring dataset $D' = (X', Y)$ for a fixed Y, where $X' = X \cup \{v_t\}$ for $v_t \notin X$ or $X' = X \backslash \{v_r\}$ for $v_r \in X$.

Although the neighboring dataset can be any dataset by adding or removing one item in X, the output intersection Z must be a subset of the fixed Y. Therefore, the output range can be defined as a set description with the capability of including all items in Y. Kacsmar et al. [31] chose the output range in the most compressed form, that is, a Boolean vector $F = \{0,1\}^m$ in which the Boolean value f_i indicates whether $y_i \in Z$. Combining it with the definition of the neighboring dataset leads to the DPSI definition described as follows.

Definition 2 (ϵ-Differentially Private Set Intersection [31]). *The randomized two-party mechanism $\mathcal{M} : \mathcal{V}^n \times \mathcal{V}^m \to \{0,1\}^m$ that takes $\mathcal{P}_1$'s set X and $\mathcal{P}_2$'s set Y as inputs and returns a Boolean vector $F \in \{0,1\}^m$ to $\mathcal{P}_2$ achieves ϵ-DPSI if and only if for all $F \in \{0,1\}^m$ and all neighboring datasets $D = (X,Y)$ and $D' = (X',Y)$ where $X' = X \cup \{v_t\}$ or $X' = X \setminus \{v_r\}$, $v_t, v_r \in \mathcal{V}$,*

$$\Pr[\mathcal{M}(D) = F] \le e^{\epsilon} \Pr[\mathcal{M}(D') = F]$$

where the probability is taken over the randomness of $\mathcal{M}$.

Definition 2 formally clarifies the privacy concerns of DPSI. However, since a DPSI protocol inherently relates to both data **security** and data **privacy**, a definition from the security aspect is also required, which is necessary for protocol constructions but *does not formally provided in [31]*. We formulate the ideal functionality $\mathcal{F}_{\mathsf{DPSI}}$ in Fig. 4, which takes X from $\mathcal{P}_1$ and Y from $\mathcal{P}_2$ as inputs and only outputs a perturbed Boolean vector to $\mathcal{P}_2$. It is parameterized by a DP mechanism $\mathcal{M}_{dp}$ constraint with the hamming distance $HM(\bar{F}, \bar{F}') = 1$ between (non-private) outputs from neighboring dataset X and X'. The output of $\mathcal{F}_{\mathsf{DPSI}}$ clearly achieves ϵ-DPSI.

Finally, a protocol provides ϵ-SIM-CDPSI (simulation-based computational differentially private PSI) security if it securely realizes $\mathcal{F}_{\mathsf{DPSI}}$ parameterized by a certain DP mechanism $\mathcal{M}_{dp}$ against a semi-honest non-uniform probabilistic polynomial time adversary under the simulation-based security paradigm. This definition is shown in Definition 7 of [31]. We omit it here and refer to [31] for the details.

4.2 Tools for Constructing DPSI

Recall that the output range in DPSI definition (Definition 2) is $\mathcal{F} = \{0,1\}^m$. The output range indicates that any item in Y can appear in the output intersection Z regardless of whether the same item is contained in X or not. To this

$\mathcal{F}_{\mathsf{DPSI}}$

Parameters: $\mathcal{P}_1$'s input set size n, and $\mathcal{P}_2$'s input set size m. A DP mechanism $\mathcal{M}_{dp}$ parameterized by the privacy budget ϵ satisfying $\Pr[\mathcal{M}_{dp}(\bar{F}) = F] \leq e^{\epsilon} \Pr[\mathcal{M}_{dp}(\bar{F}') = F]$ for all $\bar{F}, \bar{F}' \in \{0,1\}^m$ with $HM(\bar{F}, \bar{F}') = 1$.
Input: $\mathcal{P}_1$'s set $X = \{x_i\}_{i \in [n]} \in \mathcal{V}^n$, and $\mathcal{P}_2$'s set $Y = \{y_i\}_{i \in [m]} \in \mathcal{V}^m$.
Functionality: Upon receiving X from $\mathcal{P}_1$, Y from $\mathcal{P}_2$, compute $\bar{F} \in \{0,1\}^m$, where $\bar{f}_i$ indicates if $y_i \in X$. Apply $\mathcal{M}_{dp}$ to $\bar{F}$ and send the result $F = \mathcal{M}_{dp}(\bar{F})$ to $\mathcal{P}_2$.

Fig. 4: Ideal functionality $\mathcal{F}_{\mathsf{DPSI}}$ for DPSI.

end, we require that the randomized mechanism $\mathcal{M}$ for DPSI is able to perturb items in $Y \backslash X$ to items in the output intersection Z. Although the conclusion is intuitive, finding suitable PSI tools takes work. $\mathcal{F}_{\mathsf{DPSI}}$ indicates that in the absence of a trusted third party, the perturbation can only occur on $\mathcal{P}_1$'s side. This means $\mathcal{P}_1$ paradoxically cannot directly obtain the output so that it can only *obliviously perturb without knowing the output.*

Consequently, we identify the key desiderata required by PSI-related tools to construct DPSI: $\mathcal{P}_1$ must get the membership information of Y and can map the items in its set into the output intersection set via a randomized algorithm. Still now, two PSI-related tools satisfy this requirement: circuit-PSI [9,44,48] and mqRPMT [55]. In circuit-PSI, both parties get a shared Boolean vector of length $\beta = O(|Y|)$ obliviously indicating whether or not an item is in the intersection. Kacsmar et al.'s DPSI protocol also takes a circuit-PSI as a primitive which supports arbitrary subsequent circuits. In their protocol, the sender acquires the membership information of Y in the form of FHE's ciphertext. In mqRPMT, the sender gets a Boolean vector $(e_1, \ldots, e_n)$ indicating whether $x_i \in Y$ or not without learning the value of x_i. These two PSI-related tools allow the sender to randomly perturb the final binary output using a DP mechanism.

The remaining procedure is straightforward. We can leverage the Random Response (RR) mechanism $\mathcal{M}_{RR}$ [50], a basic DP mechanism for binary data, to do the perturbation. Take a binary input $v \in \{0,1\}$, $\mathcal{M}_{RR}$ outputs $\tilde{v} = v$ with probability p and $\tilde{v} = 1 - v$ with probability $q = 1 - p$. $\mathcal{M}_{RR}$ is defined as:

$$\Pr[\tilde{v} = 1] = \begin{cases} p = \frac{e^{\epsilon}}{e^{\epsilon}+1} & \text{if } v = 1 \\ q = \frac{1}{e^{\epsilon}+1} & \text{if } v = 0 \end{cases}. \tag{1}$$

Based on these eligible tools, we present two generic frameworks for constructing DPSI. As indicated by their respective designations, the first framework operates on the circuit-PSI, while the second framework is founded on the mqRPMT. Although both can achieve linear computation and communication complexity, they differ in terms of efficiency and security. The circuit-DPSI construction can be viewed as a generalization of Kacsmar et al.'s FHE-based protocol. In this framework, the sender learns nothing from the additive shares,

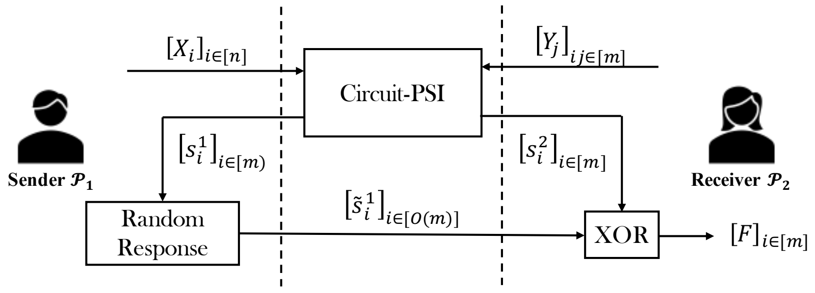

Fig. 5: Workflow of circuit-DPSI.

but can perturb the output indirectly by flipping the bits in these shares, which rigorously satisfies $\mathcal{F}_{\mathsf{DPSI}}$. In contrast, the mqRPMT-DPSI framework offers improved efficiency at the price of leaking the intersection cardinality. To make the leakage acceptable without completely undermining the idea of computation, we further introduce a method to bound such leakage with (ϵ, δ)-DP.

4.3 The Practical Significance and Limitations

As a combination of DP and PSI, our DPSI construction ensures that inferences about the presence or absence of any specific item in the output intersection are limited, while the intersection as a whole remains sufficiently accurate for substantive usage. The utility-privacy trade-off for employing our frameworks manifests in the form of false positives and negatives within the intersection. Nonetheless, such a trade-off can be tailored to suit the specific requirements of the relevant application, as illustrated in the subsequent case.

Recall that the case of a card-payment company and an ads platform identify the set of customers that they have in common. By using the constructions of our DPSI, these two parties can agree on a differentially private set intersection that protects any item in the set from the card-payment company while preserving the effectiveness of the ads. Depending on their privacy preference, the card-payment company may choose a different utility-privacy trade-off. If the card payment company is privacy-sensitive, it may choose to minimize potential inferences about individual items by accepting a higher rate of false negatives and false positives in the intersection. If the card-payment company prioritizes the effectiveness of the ads, they may ask for a lower rate of false negatives and false positives in the intersection.

Note that if the same DPSI protocol is used repeatedly to compute the intersection of the same two sets, the probability of item leakage in the intersection will increase. Such leakage is actually the DPSI functionality: each time the DPSI protocol is used to compute an intersection, it will consume a portion of the privacy budget, denoted as ϵ. According to the parallel composition described in Theorem 2, the combination of all computed intersections satisfies $\sum_{i=1}^{n} \epsilon$-DPSI, where n represents the number of times the protocol is reused.

5 Circuit-DPSI Construction

$\mathbf{\Pi}_{\text{circuit-DPSI}}$

Parameters: $\mathcal{P}_1$'s input set size n, and $\mathcal{P}_2$'s input set size m. The RR mechanism $\mathcal{M}_{RR}$ parameterized with the DP parameter ϵ.
Input: $\mathcal{P}_1$'s set $X = \{x_i\}_{i\in[n]}$, and $\mathcal{P}_2$'s set $Y = \{y_i\}_{i\in[m]}$.
Protocol:

1. **(Circuit-PSI Invocation)** $\mathcal{P}_1$ and $\mathcal{P}_2$ invoke $\mathcal{F}_{\text{circuit-PSI}}$ with inputs X and Y and get the Boolean shares $\langle a_j\rangle_1, \langle a_j\rangle_2 \in \{0,1\}, j \in [\beta]$ as output, where $\beta = (1+\alpha)m$ with $\alpha \geq 0$.
2. **(Boolean Shares Perturbation)** $\mathcal{P}_1$ invokes $\mathcal{M}_{RR}$ with input $\langle a_1\rangle_1, ..., \langle a_\beta\rangle_1$ and gets the perturbed $\langle \tilde{a_1}\rangle_1, ..., \langle \tilde{a_\beta}\rangle_1$ as output.
3. **(Intersection Computation)** $\mathcal{P}_2$ receives the perturbed $\langle \tilde{a_1}\rangle_1, ..., \langle \tilde{a_\beta}\rangle_1$ from $\mathcal{P}_1$ and computes $\mathcal{O} = \{\langle \tilde{a_j}\rangle_1 \oplus \langle a_j\rangle_2\}_{j\in[\beta]}$.
4. **(Dummy Items Deletion)** $\mathcal{P}_2$ gets the output $\tilde{F} \in \{0,1\}^m$ by setting bits that correspond to random dummy items from $\mathcal{O}$ to be 0.

Fig. 6: Generic construction 1: circuit-DPSI $\mathbf{\Pi}_{\text{circuit-DPSI}}$.

We first show the flow of the circuit-DPSI in Fig. 5 and give the details in Fig. 6. In the circuit-DPSI, $\mathcal{P}_1$ and $\mathcal{P}_2$ first invoke a construction of circuit-PSI with inputs X and Y. Then, each of $\mathcal{P}_1$ and $\mathcal{P}_2$ gets a piece of Boolean shares $\{\langle a_j\rangle_i\}_{j\in[\beta]} \in \{0,1\}$, $i \in \{1,2\}$, as the output. The Boolean shares have $\beta = (1+\alpha)m$ bits, which contain not only part of the information about the membership of the items in Y but also that of some random dummy values added during the Cuckoo Hashing of circuit-PSI protocol. After getting $\{\langle a_j\rangle_1\}_{j\in[\beta]}$, $\mathcal{P}_1$ perturbs it bit by bit via the RR mechanism $\mathcal{M}_{RR}$ and sends the perturbed Boolean shares $\{\langle \tilde{a}_j\rangle_1\}_{j\in[\beta]}$ to $\mathcal{P}_2$. Finally, $\mathcal{P}_2$ receives $\{\langle \tilde{a}_j\rangle_1\}_{j\in[\beta]}$ and computes a Boolean vector $\mathcal{O} = \{\langle \tilde{a_j}\rangle_1 \oplus \langle a_j\rangle_2\}_{j\in[\beta]}$, which indicates the membership of the items in Y and the random dummy values. By deleting the bits that correspond to the random dummy value from $\mathcal{O}$, $\mathcal{P}_2$ obtains $F = \{0,1\}^m$, a set where each position i contains a one if the item corresponding to Y_i is part of the differentially private intersection, and zeros in all other positions.

It is worth noting that within the circuit-DPSI construction, the integration of the circuit-PSI protocol is designed to be modular. By selecting and incorporating an appropriate circuit-PSI protocol, we achieve a more efficient circuit-DPSI protocol.

As shown by Mahdavi et al. [37], the FHE-based DPSI protocol proposed by Kacsmar et al. [31] can be regarded as a variant of the circuit-PSI construction if we remove the RR mechanism. In this protocol, $\mathcal{P}_2$ encrypts their items using the BGV [51] scheme and sends the ciphertext to $\mathcal{P}_1$. $\mathcal{P}_1$ also encrypts their items and computes encryption of a vector containing ones in positions where

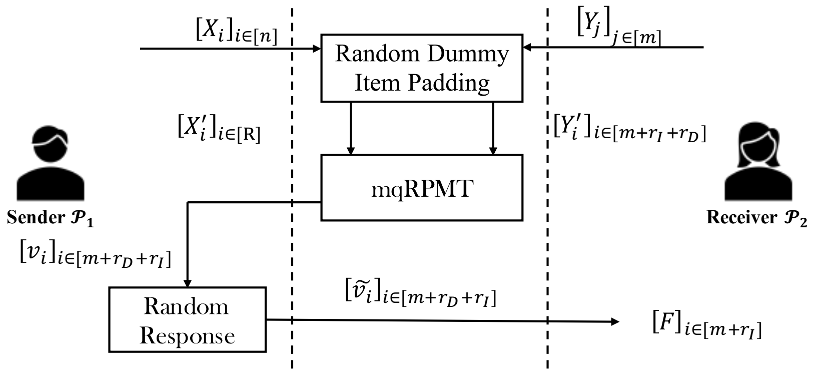

Fig. 7: Workflow of mqRPMT-DPSI

the corresponding items are found in the intersection. Moreover, $\mathcal{P}_1$ perturbs the encrypted vector using the RR mechanism $\mathcal{M}_{RR}$ and sends the results to $\mathcal{P}_2$. Finally, $\mathcal{P}_2$ decrypts the vector and obtains the perturbed intersection. It is evident that the workflow of this FHE-based DPSI protocol is similar to our circuit-DPSI construction, as both involve obtaining a vector containing membership information and perturbing it using the RR mechanism $\mathcal{M}_{RR}$.

The security and privacy result of $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$ is formally given in Theorem 3. The detailed proof is shown in Appendix A.

Theorem 3. $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$ *in Fig. 6 satisfies ϵ-SIM-*DPSI.

6 mqRPMT-based DPSI Construction

By removing the dependency on FHE, the circuit-DPSI construction is much more efficient than the existing construction of DPSI. However, SOTA circuit-PSI constructions [9,44] still utilizes sub-protocols, namely, Private Equality Test (PEQT) [44] or PSM [9]), that follows the general secure computation paradigm to compute the shares of $\bar{F}$, which accounts for 96% of the overall computational overhead of the circuit-PSI primitive. Therefore, we intuitively consider whether we can improve the efficiency of DPSI by performing the membership test without relying on general secure computation sub-protocols. To this end, we propose the mqRPMT-DPSI construction. Different from the circuit-PSI where both parties learn the secret sharing of the membership vector based on general secure computation sub-protocols, mqRPMT allows the sender to learn the complete membership vector efficiently.

The mqRPMT-DPSI construction offers improved efficiency at the price of relaxing security by allowing the protocol to leak some "extra" information. The functionality of mqRPMT allows the sender to obtain the membership vector v in plaintext. Although a permutation operator performed by the receiver can make sure the membership bits are of the permuted order unknown to the sender, the sender can still learn the cardinality of intersection (PSI-CA) by counting the number of "1" in v, or the cardinality of $Y \backslash (X \bigcap Y)$ by counting the number of "0" in v. Recent studies [21, 30] have found that adversaries can launch an efficient attack on the size-revealing PSIs, with a binary-search-like strategy, to de-anonymize specific data records in the input sets, thereby leaking membership information about the input sets. To take advantage of mqRPMT without completely undermining the idea of secure computation, a natural candidate is to leak only DP information about the PSI-CA.

A straightforward strategy to ensure that the leakage learned by the sender satisfies DP is to generate a random integer from a truncated discrete Laplace distribution and then add the corresponding number of dummy items to both the intersection and difference sets before the mqRPMT process.

However, in our two-party scenario without a trusted third party, the dummy items cannot be added to the intersection directly and the alternative strategy is that both parties generate a random integer from a truncated discrete Laplace distribution, respectively, and add the corresponding number of dummy items to their own private set. Unfortunately, this alternative strategy fails to satisfy DP. Note that a dummy item can be found in the intersection if and only if both the sender and receiver possess this dummy item. Consequently, the maximum number of dummy items in the intersection is inherently limited by the number of dummy items added by the sender locally. Consider a worst-case scenario in which the sender draws a random integer 0 from a truncated discrete Laplace distribution and adds 0 dummy items into his/her local set, the sender can learn the true PSI-CA regardless of the number of dummy items added by the receiver.

Definition 3 (Truncated Discrete Laplace Mechanism). *Given a query $\mathbf{c} : \mathcal{D} \to \mathbb{N}$, the truncated discrete Laplace mechanism $TLap(D)$ outputs an integer $\max(\mathbf{c}(D) + \eta, \mathbf{c}(D))$, where $\eta \in \mathbb{Z}$ follows a distribution, denoted by $L(\epsilon, \delta, \Delta\mathbf{c})$ that has a probability density function $\Pr[\eta = x] = p \cdot e^{-(\epsilon/\Delta\mathbf{c})|x-\eta^0|}$, where $p = \frac{e^{\epsilon/\Delta\mathbf{c}}-1}{e^{\epsilon/\Delta\mathbf{c}}+1}$, $\eta^0 = \Delta\mathbf{c} - 1 + \frac{\ln[(e^\epsilon+1)(1-\delta)]}{\epsilon}$, $\Delta\mathbf{c}$ is the sensitivity parameter for the query $\mathbf{c}$, ϵ is the privacy budget and δ is the error probability.*

Lemma 1 (Truncated Discrete Laplace Mechanism Utility [8]). *For $\eta \in \mathbb{Z}$ that follows $L(\epsilon, \delta, \Delta\mathbf{c})$ and any $m \in \mathbb{N}$, we have $\Pr[\eta \geq m] = \frac{e^{-\epsilon(m-\eta^0-1)}}{e^\epsilon+1}$.*

To this end, we modify the above strategy as follows, which is shown in Algorithm 1. In our strategy, the sender and the receiver first independently pad the items in their own sets to bit strings with "0" at the highest position. Next, to ensure that the number of dummy items in the intersection remains oblivious to the sender, the sender adds R dummy items $\{1|i|0\}_{i\in[R]}$ into X to obtain X', where R is typically a large integer leading to enough redundant dummy items in X'. As a result, the number of dummy items added to the intersection is only determined by the dummy items generated by the receiver, which is oblivious to the sender. Simultaneously, the receiver draws two random integer r_I, r_D from two truncated discrete Laplace distributions, which is described in Definition 3, and adds r_I dummy items $\{1|i|0\}_{i\in[r]}$ and r_D dummy items $\{1|i|1\}_{i\in[r]}$ into Y'.

Algorithm 1: Random Dummy Item Padding $\mathcal{M}_{\mathsf{rd}}$

```
   Parameters: Statistical security parameter $\sigma$. DP parameter $\epsilon_I, \epsilon_D$.
   Input: Sender's input set $X = \{x_1, x_2, ...x_n\}$. Receiver's input set
          $Y = \{y_1, y_2, ...y_m\}$.
   /* Sender side                                                        */
 1 Compute $R$: the minimum number satisfying $\Pr(TLap(\frac{1}{\epsilon_i}) > R) < 2^{-\sigma}$;
 2 Initialize an empty set $X'$;
 3 Add padded real item $\{0|x\}_{x \in X}$ into $X'$;
 4 Add dummy items $\{0|i|0\}_{i \in R}$ into $X'$;
 5 Return $X'$;
   /* Receiver side                                                      */
 6 $r_I = L(\epsilon_I, \delta, 1)$, $r_D = TLap(\epsilon_D, \delta, 1)$;
 7 Initialize an empty set $Y'$;
 8 Add padded real item $\{0|y\}_{y \in X}$ into $Y'$;
 9 Add items $\{0|i|0\}_{i \in r_I}$ into $Y'$ ;            // Dummy items should be in $X'$
10 Add items $\{0|i|1\}_{i \in r_D}$ into $Y'$ ;        // Dummy items should not be in $X'$
11 Return $Y'$.
```

$\Pi_{\mathsf{mqRPMT\text{-}DPSI}}$

Parameters: $\mathcal{P}_1$'s input set size n. $\mathcal{P}_2$'s input set size m. The random dummy item padding mechanism $\mathcal{M}_{\mathsf{rd}}$ parameterized with DP parameters ϵ_i, ϵ_d. The RR mechanism $\mathcal{M}_{RR}$ parameterized with the DP parameter ϵ.
Input: $\mathcal{P}_1$'s set $X = \{x_i\}_{i \in [n]}$, $\mathcal{P}_2$'s set $Y = \{y_i\}_{i \in [m]}$.
Protocol:

1. **(Dummy Item Padding)** $\mathcal{P}_1$ invokes $\mathcal{M}_{\mathsf{rd}}$ with X as input and output a padded set $X' = \{x'_i\}_{i \in [n+R]}$ with R padded items.
2. $\mathcal{P}_2$ invokes $\mathcal{M}_{\mathsf{rd}}$ with Y as input and output a padded set $Y' = \{y'_i\}_{i \in [M]}$, where $M = m + r_I + r_D$ and r_I, r_D are random integers drawn from a truncated discrete Laplace distribution. $\mathcal{P}_2$ permutes the Y' with a random permutation ϕ and obtain Y^*, where $y^*_i = y'_{\phi(i)}$.
3. **(mqRPMT Invocation)** $\mathcal{P}_1$ and $\mathcal{P}_2$ invoke $\mathcal{F}_{\mathsf{mqRPMT}}$ with inputs X' and Y^*. $\mathcal{P}_1$ gets an indication Boolean vector $\{v_i\}_{i \in [M]}$ such that $v_i = 1$ if and only if $y^*_i \in X'$ but without knowing y^*_i.
4. **(indication Boolean vector Perturbation)** $\mathcal{P}_1$ invokes $\mathcal{M}_{RR}$ with input v and gets the perturbed indication Boolean vector $\tilde{v}$ as output.
5. **(Intersection Computation)** $\mathcal{P}_2$ receives the perturbed indication Boolean vector $\tilde{v}$ from $\mathcal{P}_1$, permutes $\tilde{v}$ with ϕ^{-1} to obtain $\tilde{v}'$, and gets the indication Boolean vector $F \in \{0,1\}^m$ by deleting the bits that corresponding with the dummy item from $\tilde{v}'$.

Fig. 8: Generic construction 2: $\Pi_{\mathsf{mqRPMT\text{-}DPSI}}$.

Finally, the sender and receiver exchange $|X'| = n+R$ and $|T'| = m+r_I+r_D$ with each other. Consequently, the overall strategy bounds the leakage to the sender with $(\epsilon_I + \epsilon_D, 2\delta)$-DP if and only if the truncated discrete Laplace mechanism satisfies (ϵ', δ)-DP, where $\epsilon' \in \{\epsilon_I, \epsilon_D\}$.

In the Random Dummy Item Padding Mechanism $\mathcal{M}_{\mathsf{rd}}$, determining the size of R is an important issue, as the magnitude of R simultaneously affects the computation complexity of the entire construction and the inherent error rate of the padding mechanism. In our implementation, we set R as the minimum number satisfying $\Pr[r_I > R] \leq 2^{-\sigma}$, where $\sigma = 40$ is the statistical security parameter in MPC. It means the number of dummy items added into the intersection satisfying (ϵ_I, δ)-DP with a high enough probability. The result of our experiment shows that: with fixed $\delta = 10^{-5}$, $\epsilon_I = 0.01, R = 2774$; $\epsilon_I = 0.1, R = 279$; $\epsilon_I = 1.0, R = 29$; $\epsilon_I = 10.0, R = 4$.

We show the flow of the mqRPMT-DPSI protocol in Fig. 7 and give the details in Fig. 8. In the mqRPMT-DPSI, $\mathcal{P}_1$ and $\mathcal{P}_2$ first insert some dummy items into their private sets X and Y via the Random Dummy Item Padding Mechanism and get the padded sets X' and Y', respectively. Next, $\mathcal{P}_1$ and $\mathcal{P}_2$ invoke mqRPMT with inputs X' and Y'. As the functionality of mqRPMT, $\mathcal{P}_1$ gets an indication Boolean vector $\{v_i\}_{i\in[M]}$ such that $v_i = 1$ if and only if $y'_i \in X'$ but without knowing y'_i. After getting $\{\langle v_i\rangle\}_{i\in[M]}$, $\mathcal{P}_1$ perturbs it bit by bit via the RR mechanism and sends the perturbed Boolean values $\{\langle \tilde{v}_i\rangle\}_{i\in[M]}$ to $\mathcal{P}_2$. Finally, $\mathcal{P}_2$ learns $F = \{0,1\}^m$ by deleting the bits corresponding to the dummy items from $\{\langle \tilde{v}_i\rangle\}_{i\in[M]}$. Similar to $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$, the $\mathcal{F}_{\mathsf{mqRPMT}}$ in $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$ can be instantiated with any efficient mqRPMT protocols.

The security and privacy result of $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$ is formally given in Theorem 4. The detailed proof is shown in Appendix B.

Theorem 4. *The protocol in Fig. 8 satisfies ϵ-SIM-DPSI with the information leakage to $\mathcal{P}_1$ bounded by $(\epsilon_I + \epsilon_D, 2\delta)$-DP.*

7 Experiments

7.1 Implementation Details

We implement three circuit-DPSI protocols and an mqRPMT-DPSI protocol by plugging the SOTA circuit-PSI protocols [9, 44, 48] and mqRPMT protocol [10] into the proposed DPSI constructions, respectively. Moreover, we also implement all circuit-PSI protocols and mqRPMT-based PSI protocol above for comparison. Specifically, all these protocols mentioned above are written in JAVA and the source code is available at `https://anonymous.4open.science/r/dpsi-58EB`.

We denote the circuit-PSI protocols and mqRPMT-based PSI protocol, which are serving as primitives, as PSTY'19, RS'21, CGS'22, and CZZ'24. The DPSI protocols that are built upon them are denoted as PSTY'19*, RS'21*, CGS'22*, and CZZ'24*. Moreover, we also list the running time and communication cost of the FHE-based circuit-DPSI protocol mentioned in [31] and denote it as KKL+'20. Note that the experimental results of KKL+'20 are copied from [31].

Since this protocol is obviously inefficient even though it is implemented in C/C++, any further detailed comparison between this protocol and our protocols may not be necessary.

Datasets. We conduct different experiments on both synthetic and real datasets. To evaluate the runtime and communication cost of implemented protocols, which are only affected by the size of the dataset, we construct synthetic datasets separately for both parties, each containing distinct keywords with some overlap.

Experimental Setup. For all implemented protocols, we set the computational security parameter $\kappa = 128$ and the statistical security parameter $\sigma = 40$. To evaluate the performance of implemented protocols, we conduct a series of experiments with different sizes of items: $n = m = 2^{14}, 2^{16}, 2^{18}, 2^{20}$, as well as different network environments, including LAN and WAN. For the DP mechanism used in the implemented DPSI protocols, we use the following default values unless specifically stated: $\epsilon = 1.0, \delta = \frac{1}{n}$.

We run all our protocols and related protocols on a single Intel Core i9-9900K with 3.6GHz and 128GB RAM. We simulate the network connection using Linux `tc` command. For the WAN setting, we set the average RTT to be 80 ms and bandwidth to be 100 Mbps. We use `iptables` command to calculate the communication cost, and use the running time to compute the computation complexity, which is the maximum time from the protocol beginning to the end, including the messages transmission time.

7.2 Performance Results

Running times. Table 1 shows the running time of all the implemented protocols. We can find that the running time of all DPSI protocols increases linearly with the growth of the data size. Among the PSI and DPSI protocols we evaluated, the mqRPMT-DPSI protocol CZZ'24* consistently outperformed the others in all network environments. Specifically, the CZZ'24* is approximately 3.2-5.8× faster than the other circuit-DPSI protocols. This is consistent with our analyses that mqRPMT-DPSI protocols will be faster than the circuit-DSPI protocols since the latter uses general secure computation sub-protocols as a fundamental technology. We also observe that the PSTY'19* is the fastest among all circuit-DPSI protocols over different sizes of data in the WAN setting and when $n = 2^{14}, 2^{16}$ in the LAN setting. The reason is that we implement the PSTY'19* by combining the circuit-PSI protocol in [44] with the PSM protocol in [9]. Finally, the running time of all DPSI protocols is roughly the same as their corresponding PSI protocols. The reason is that DPSI counterparts only additionally involve the efficient RR mechanism and adding a few dummy items into both parties' sets in advance, both of which introduce marginal costs.

Communication Costs. Table 2 shows the total communication cost of all implemented protocols. We can find that all DPSI protocols achieve strict linear communication complexity. Specifically, except for the CGS'22*, the communication overhead of the other DPSI protocols is relatively consistent. Among them,

Table 1: Runtime of DPSI protocols on $n = m = 2^{14}, 2^{16}, 2^{18}, 2^{20}$ set sizes under LAN and WAN network settings. The results for KKL+'20 are copied from [31].

Primitive	Protocol	Runtime(s)							
		LAN				WAN			
		2^{14}	2^{16}	2^{18}	2^{20}	2^{14}	2^{16}	2^{18}	2^{20}
Circuit-PSI	PSTY'19	4.26	19.57	96.97	457.74	8.52	23.95	107.06	474.90
	RS'21	5.35	20.98	99.96	474.68	8.67	27.36	111.93	488.63
	CGS'22	5.29	21.81	102.67	485.09	8.47	26.25	110.51	501.74
Circuit-DPSI	PSTY'19*	4.14	19.15	94.93	451.62	5.11	19.90	99.92	488.06
	RS'21*	4.95	19.67	97.21	465.22	5.46	21.19	99.11	479.51
	CGS'22*	4.48	20.75	102.25	476.95	9.44	26.03	111.79	503.86
	KKL+'20	-	531.62	-	11719.4	-	-	-	-
mqRPMT-PSI	CZZ'24	1.24	4.80	19.24	76.87	1.44	5.57	21.29	84.31
mqRPMT-DPSI	CZZ'24*	1.19	4.81	19.14	77.22	1.61	5.76	21.55	85.37

the mqRPMT-DPSI protocol has the lowest communication overhead. Moreover, compared with the constructions of the corresponding PSI primitives, all DPSI protocols have similar communication overheads. This is because the perturbation operations involved in the DPSI protocols are performed locally and do not incur additional communication overhead, and the number of dummy items involved in mqPRMT-DPSI is small for reasonable DP parameters ϵ_i and ϵ_d.

Table 2: Communication cost of DPSI protocols on $n = m = 2^{14}, 2^{16}, 2^{18}, 2^{20}$ set sizes. The results for KKL+'20 are copied from [31].

Primitive	Protocol	Total Comm.(MB)			
		2^{14}	2^{16}	2^{18}	2^{20}
PSI	PSTY'19	2.63	8.82	32.08	125.93
	RS'21	3.27	9.56	33.03	127.08
	CGS'22	4.73	17.15	65.97	261.39
	CZZ'24	1.26	5.05	20.74	82.97
circuit-DPSI	PSTY'19*	2.64	9.04	32.16	126.25
	RS'21*	3.27	9.58	33.11	127.4
	CGS'22*	5.74	17.17	66.05	261.76
	KKL+'22	-	133	-	232
mqRPMT-DPSI	CZZ'24*	1.41	5.88	23.5	94

7.3 Utility Results

FPR and FNR. The output of the DPSI protocols is a differentially private set, which contains both false negatives and false positives. According to Equation (1), both the false negative rate (FNR) and false positive rate (FPR) are theoretically $q = \frac{1}{1+e^{\epsilon}}$, which indicates that the FNR and FPR are determined by the privacy budget ϵ: a larger ϵ will lead to smaller FNR and FPR at the expense of weaker privacy guarantee, while a smaller ϵ will result in lower utility. In practice, the parties need to negotiate and define a consensus ϵ in advance to strike a balance between privacy and utility that aligns with their specific needs.

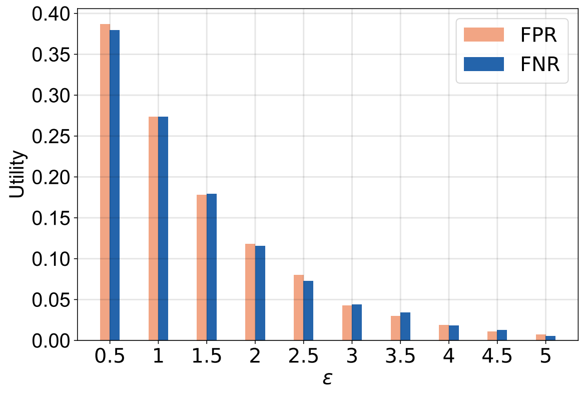

Fig. 9: The FPR and FNR of mqRPMT-DPSI protocol

To verify our theoretical analysis of the utility of DPSI protocols, we additionally design a series of experiments to measure the FPR and FNR of the chosen implemented DPSI protocol while varying the private budget ϵ from 0.5 to 5.0. Specifically, since the utility of all DPSI protocols is the same, we chose the mqRPMT-DPSI protocol as a representative protocol and conducted experiments to evaluate its FPR and FNR under different privacy budgets. The results in Fig. 9 support and validate our analyses: as privacy budget ϵ increases, the FPR and FNR will decrease gradually. It indicates that the parties can strike a balance between privacy and utility by choosing a proper ϵ.

Effectiveness of Random Dummy Item Padding Mechanism. Finally, we evaluate the effectiveness of the Random Dummy Item Padding Mechanism, the key building block in the mqRPMT-DPSI construction. We count the distribution of Random Dummy Item Padding Mechanism's outputs in a different setting. Specifically, we run Random Dummy Item Padding Mechanism for 10^5 times while varying the privacy budget ϵ from 1.0 to 5.0 under $\delta = 0.1$ and $\delta = 0.0001$. In all experiments, we use the default values $n = m = 2^9$, and show the results in Fig. 10 and Fig. 11. It demonstrates that when $\delta = 0.0001$, which is a much more practical setting in the above experiments, as the privacy budget ϵ increases, the distribution of the outputs becomes centralized gradually. It means that a smaller ϵ can lead to a stronger privacy-preserving capability. In contrast, when $\delta = 0.1$, the output is not adequately randomized, implying a higher probability of accidental information leakage.

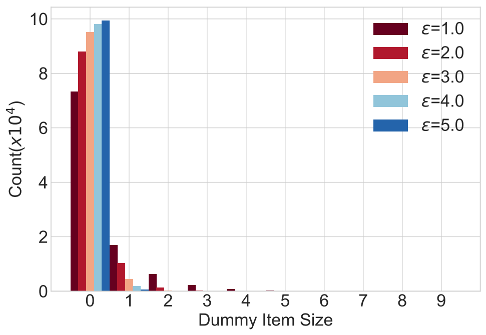

Fig. 10: The distribution of dummy items' size ($\delta = 0.1$)

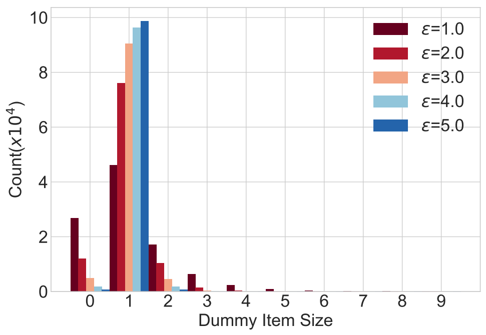

Fig. 11: The distribution of dummy items' size ($\delta = 0.0001$)

8 Conclusion

In this paper, we revisit the definition of DPSI, introduce its ideal functionality, and propose two generic and modular constructions. The first one is $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$, which is secure and satisfies ϵ-DP. The other one is $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$, which offers improved efficiency at the expense of privacy compromise. For $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$, we also propose a Random Dummy Item Padding Mechanism to prevent the exact PSI-CA from leaking. We prove that the released PSI-CA satisfies (ϵ, δ)-DP and the output of $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$ is secure and satisfies ϵ-DP. Experiments show that the instantiations of $\mathbf{\Pi}_{\mathsf{mqRPMT\text{-}DPSI}}$ are arguably the most computation- and communication-efficient DPSI protocols, and the instantiations of $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$ are comparable in performance with the most efficient circuit-PSI protocols.

Although our instantiations outperform existing DPSI construction, its concrete efficiency is still worse than SOTA (non-circuit) PSI and PSI-based applications. Obtaining more efficient DPSI is a promising future direction. Another direction is to obtain DPSI with higher utility under the same privacy budget ϵ.

References

1. Asharov, G., Lindell, Y., Schneider, T., Zohner, M.: More efficient oblivious transfer extensions with security for malicious adversaries. In: Eurocrypto 2015. pp. 673–701. Springer (2015)
2. Bater, J., He, X., Ehrich, W., Machanavajjhala, A., Rogers, J.: Shrinkwrap: efficient sql query processing in differentially private data federations. Proceedings of the VLDB Endowment **12**(3) (2018)
3. Bater, J., Park, Y., He, X., Wang, X., Rogers, J.: SAQE: practical privacy-preserving approximate query processing for data federations. Proc. VLDB Endow. **13**(11), 2691–2705 (2020)
4. Bittau, A., Erlingsson, Ú., Maniatis, P., Mironov, I., Raghunathan, A., Lie, D., Rudominer, M., Kode, U., Tinnes, J., Seefeld, B.: Prochlo: Strong privacy for analytics in the crowd. In: SOSP 2017. pp. 441–459. ACM (2017)
5. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Rindal, P., Scholl, P.: Efficient two-round ot extension and silent non-interactive secure computation. In: CCS 2019. pp. 291–308. ACM (2019)
6. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. ACM Transactions on Computation Theory **6**(3), 1–36 (2014)
7. Buddhavarapu, P., Knox, A., Mohassel, P., Sengupta, S., Taubeneck, E., Vlaskin, V.: Private matching for compute. Cryptology ePrint Archive (2020), `https://eprint.iacr.org/2020/599`
8. Canonne, C.L., Kamath, G., Steinke, T.: The discrete gaussian for differential privacy. In: NeurIPS 2020. NeurIPS Foundation (2020)
9. Chandran, N., Gupta, D., Shah, A.: Circuit-psi with linear complexity via relaxed batch opprf. Proceedings on Privacy Enhancing Technologies Symposium **2022**(1), 353–372 (2022)
10. Chen, Y., Zhang, M., Zhang, C., Dong, M., Liu, W.: Private set operations from multi-query reverse private membership test. In: PKC 2024. pp. 387–416. Springer Nature Switzerland (2024)
11. Couteau, G., Rindal, P., Raghuraman, S.: Silver: silent vole and oblivious transfer from hardness of decoding structured ldpc codes. In: Crypto 2021. pp. 502–534. Springer (2021)
12. Cristofaro, E.D., Tsudik, G.: Practical private set intersection protocols with linear complexity. In: FC 2010. pp. 143–159. Springer (2010)
13. Duchi, J.C., Jordan, M.I., Wainwrigh, M.J.: Local privacy and statistical minimax rates. In: FOCS 2013 (2013)
14. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science **9**(3–4), 211–407 (2014)
15. Edmonds, A., Nikolov, A., Ullman, J.R.: The power of factorization mechanisms in local and central differential privacy. In: STOC 2020. pp. 425–438. ACM (2020)
16. Erlingsson, Ú., Pihur, V., Korolova, A.: RAPPOR: randomized aggregatable privacy-preserving ordinal response. In: CCS 2014. pp. 1054–1067. ACM (2014)
17. Garimella, G., Pinkas, B., Rosulek, M., Trieu, N., Yanai, A.: Oblivious key-value stores and amplification for private set intersection. In: Crypto 2021. pp. 395–425. Springer (2021)
18. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC 2009. pp. 169–178. ACM (2009)

19. Goldreich, O.: Foundations of cryptography: volume 2, Basic Applications. Cambridge University Press (2009)
20. Groce, A., Rindal, P., Rosulek, M.: Cheaper private set intersection via differentially private leakage. Proc. Priv. Enhancing Technol. **2019**(3), 6–25 (2019)
21. Guo, X., Han, Y., Liu, Z., Wang, D., Jia, Y., Li, J.: Birds of a feather flock together: How set bias helps to deanonymize you via revealed intersection sizes. In: USENIX Security 2022. pp. 1487–1504. USENIX (2022)
22. Haney, S., Machanavajjhala, A., Abowd, J.M., Graham, M., Kutzbach, M., Vilhuber, L.: Utility cost of formal privacy for releasing national employer-employee statistics. In: SIGMOD 2017. pp. 1339–1354. ACM (2017)
23. Hardt, M., Ligett, K., McSherry, F.: A simple and practical algorithm for differentially private data release. In: NeurIPS 2012. pp. 2348–2356. NeurIPS Foundation (2012)
24. Hay, M., Rastogi, V., Miklau, G., Suciu, D.: Boosting the accuracy of differentially private histograms through consistency. Proc. VLDB Endow. **3**(1), 1021–1032 (2010)
25. He, X., Machanavajjhala, A., Flynn, C., Srivastava, D.: Composing differential privacy and secure computation: A case study on scaling private record linkage. In: CCS 2017. pp. 1389–1406. ACM (2017)
26. Huang, Y., Evans, D., Katz, J.: Private set intersection: Are garbled circuits better than custom protocols? In: NDSS 2012 (2012)
27. Information Commissioner's Office: Chapter 5: Privacy-enhancing technologies (2022)
28. Ion, M., Kreuter, B., Nergiz, A.E., Patel, S., Saxena, S., Seth, K., Raykova, M., Shanahan, D., Yung, M.: On deploying secure computing: Private intersection-sum-with-cardinality. In: EuroS&P 2020. pp. 370–389. IEEE (2020)
29. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: Crypto 2003. pp. 145–161. Springer (2003)
30. Jiang, B., Du, J., Yan, Q.: Anonpsi: An anonymity assessment framework for PSI. Cryptology ePrint Archive (2024)
31. Kacsmar, B., Khurram, B., Lukas, N., Norton, A., Shafieinejad, M., Shang, Z., Baseri, Y., Sepehri, M., Oya, S., Kerschbaum, F.: Differentially private two-party set operations. In: EuroS&P 2020. pp. 390–404. IEEE (2020)
32. Kolesnikov, V., Kumaresan, R.: Improved ot extension for transferring short secrets. In: Crypto 2013. pp. 54–70. Springer (2013)
33. Kolesnikov, V., Kumaresan, R., Rosulek, M., Trieu, N.: Efficient batched oblivious prf with applications to private set intersection. In: CCS 2016. pp. 818–829. ACM (2016)
34. Li, C., Hay, M., Rastogi, V., Miklau, G., McGregor, A.: Optimizing linear counting queries under differential privacy. In: PODS 2010. pp. 123–134. ACM (2010)
35. Li, C., Miklau, G., Hay, M., McGregor, A., Rastogi, V.: The matrix mechanism: optimizing linear counting queries under differential privacy. Proc. VLDB Endow. **24**(6), 757–781 (2015)
36. Machanavajjhala, A., Kifer, D., Abowd, J.M., Gehrke, J., Vilhuber, L.: Privacy: Theory meets practice on the map. In: ICDE 2008. pp. 277–286. IEEE Computer Society (2008)
37. Mahdavi, R.A., Lukas, N., Ebrahimianghazani, F., Humphries, T., Kacsmar, B., Premkumar, J.A., Li, X., Oya, S., Amjadian, E., Kerschbaum, F.: PEPSI: practically efficient private set intersection in the unbalanced setting. In: USENIX Security 2024. USENIX Association (2024)

38. Meadows, C.: A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In: S&P 1986. pp. 134–134. IEEE (1986)
39. Narayan, A., Haeberlen, A.: Djoin: Differentially private join queries over distributed databases. In: OSDI 2012. pp. 149–162. USENIX (2012)
40. Pagh, R., Rodler, F.F.: Cuckoo hashing. In: ESA 2001. pp. 121–133. Springer (2001)
41. Pinkas, B., Rosulek, M., Trieu, N., Yanai, A.: Spot-light: lightweight private set intersection from sparse ot extension. In: CRYPTO 2019. pp. 401–431. Springer (2019)
42. Pinkas, B., Rosulek, M., Trieu, N., Yanai, A.: Psi from paxos: fast, malicious private set intersection. In: EUROCRYPT 2020. pp. 739–767. Springer (2020)
43. Pinkas, B., Schneider, T., Segev, G., Zohner, M.: Phasing: Private set intersection using permutation-based hashing. In: USENIX Security 2015. pp. 515–530. USENIX (2015)
44. Pinkas, B., Schneider, T., Tkachenko, O., Yanai, A.: Efficient circuit-based psi with linear communication. In: Eurocrypto 2019. pp. 122–153. Springer (2019)
45. Raghuraman, S., Rindal, P.: Blazing fast psi from improved okvs and subfield vole. In: CCS 2022. pp. 2505–2517. ACM (2022)
46. Rastogi, V., Nath, S.: Differentially private aggregation of distributed time-series with transformation and encryption. In: SIGMOD 2010. pp. 735–746. ACM (2010)
47. Rindal, P., Rosulek, M.: Malicious-secure private set intersection via dual execution. In: CCS 2017. pp. 1229–1242. ACM (2017)
48. Rindal, P., Schoppmann, P.: Vole-psi: fast oprf and circuit-psi from vector-ole. In: Eurocrypto 2021. pp. 901–930. Springer (2021)
49. Wang, Y., Yi, K.: Secure yannakakis: Join-aggregate queries over private data. In: SIGMOD 2021. pp. 1969–1981. ACM (2021)
50. Warner, S.L.: Randomized response: A survey technique for eliminating evasive answer bias. Journal of the American Statistical Association **60**(309), 63–69 (1965)
51. Yagisawa, M.: Fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive (2015), `https://eprint.iacr.org/2011/277`
52. Yao, A.C.C.: How to generate and exchange secrets. In: FOCS 1986. pp. 162–167. IEEE (1986)
53. Yuan, G., Yang, Y., Zhang, Z., Hao, Z.: Convex optimization for linear query processing under approximate differential privacy. In: SIGKDD 2016. pp. 2005–2014. ACM (2016)
54. Yuan, G., Zhang, Z., Winslett, M., Xiao, X., Yang, Y., Hao, Z.: Optimizing batch linear queries under exact and approximate differential privacy. ACM Trans. Database Syst. **40**(2), 11:1–11:47 (2015)
55. Zhang, C., Chen, Y., Liu, W., Zhang, M., Lin, D.: Linear private set union from {Multi-Query} reverse private membership test. In: USENIX Security 2023. pp. 337–354. USENIX (2023)

A Proof of Theorem 3

We prove Theorem 3 in two steps. First, we prove that the ideal functionality $\mathbf{\Pi}_{\textsf{circuit-DPSI}}$ satisfies ϵ-DPSI. Then, we prove that the construction securely realizes the ideal functionality $\mathbf{\Pi}_{\textsf{circuit-DPSI}}$. By combining them, the theorem is proved.

Lemma 2. *$\mathcal{F}_{\mathsf{DPSI}}$ in Fig. 4 with $\mathcal{M}_{RR}$ parameterized by ϵ satisfies ϵ-DPSI.*

Proof. Given the inputs X from $\mathcal{P}_1$ and Y from $\mathcal{P}_2$, a neighboring input $X' = X \cup \{v_t\}$ or $X' = X \setminus \{v_r\}$, for any output $F \in \{0,1\}^m$ and the privacy budget ϵ, we need to prove that

$$\left|\frac{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X,Y)=F]}{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X',Y)=F]}\right| \leq e^{\epsilon}.$$

We prove it by considering four different cases: (1) $X' = X \cup \{v_t\}$, $v_t \notin Y$; (2) $X' = X \cup \{v_t\}$, $v_t \in Y$; (3) $X' = X \setminus \{v_r\}$, $v_r \notin Y$; (4) $X' = X \setminus \{v_r\}$, $v_r \in Y$.

For the first case, when $v_t \notin Y$, v_t must not be in either $X \cap Y$ or $X' \cap Y$. As a result, all other items in the input are the same, we have

$$\frac{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X,Y)=F]}{Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X',Y)=F]} = 1 \leq e^{\epsilon}$$

.

For the third case, when $v_r \notin Y$, v_r must not be in either $X \cap Y$ or $X' \cap Y$. As a result, all other items in the input are the same, we have

$$\frac{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X,Y)=F]}{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X',Y)=F]} = 1 \leq e^{\epsilon}$$

.

For the second and fourth cases, since $v_t \in Y$, W.L.O.G., we assume that the t-th bit b_t in F indicates v_t is in the intersection if $b_t = 1$, and v_t is not in the intersection if $b_t = 0$. By representing the t-th bit in $\mathcal{P}_1$'s Boolean share for input X and X' as $\langle \tilde{a}_t \rangle_1^X$ and $\langle \tilde{a}_t \rangle_1^{X'}$, respectively, we have

$$\begin{aligned} &\left|\frac{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X,Y)=F]}{\Pr[\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}(X',Y)=F]}\right| = \prod_{i\in[m]} \left|\frac{\Pr[\mathcal{F}^i_{\mathsf{circuit\text{-}DPSI}}(X,Y)=b_i]}{\Pr[\mathcal{F}^i_{\mathsf{circuit\text{-}DPSI}}(X',Y)=b_i]}\right| \\ &= \left|\frac{\Pr[\mathcal{F}^t_{\mathsf{circuit\text{-}DPSI}}(X,Y)=b_t]}{\Pr[\mathcal{F}^t_{\mathsf{circuit\text{-}DPSI}}(X',Y)=b_t]}\right| = \left|\frac{\Pr[\mathcal{M}_{RR}(\langle \tilde{a}_t \rangle_1^X)=b_t]}{\Pr[\mathcal{M}_{RR}(\langle \tilde{a}_t \rangle_1^{X'})=b_t]}\right| \leq \frac{p}{q} = e^{\epsilon}. \end{aligned} \tag{2}$$

Therefore, $\mathcal{F}_{\mathsf{circuit\text{-}DPSI}}$ with $\mathcal{M}_{RR}$ parameterized by ϵ satisfies ϵ-DPSI.

Lemma 3. *The protocol $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$ in Fig. 6 securely realizes the ideal functionality $\mathcal{F}_{\mathsf{DPSI}}$ in Fig. 4 using $\mathcal{M}_{RR}$ against a semi-honest adversary in the $\mathcal{F}_{\mathsf{circuit\text{-}PSI}}$-hybrid model.*

Proof. The correctness of $\mathbf{\Pi}_{\mathsf{circuit\text{-}DPSI}}$ is obvious, which follows the correctness of the underlying protocol realizing $\mathcal{F}_{\mathsf{circuit\text{-}PSI}}$. Then we prove the security by constructing simulators. For the corrupt semi-honest receiver, we exhibit the simulator Sim_R as follows.

1. Sim_R receives $\mathcal{P}_2$'s input Y, invokes $\mathcal{F}_{\mathsf{DPSI}}$ to obtain the output F', and appends them to the view.

2. Sim_R uses H_s and Y to honestly compute HT_2 with size β, and selects random bits $\langle a'_j \rangle_2 \leftarrow \{0,1\}$ for $j \in [\beta]$. Then, Sim_R invokes the simulator of $\mathcal{F}_{\mathsf{circuit\text{-}PSI}}$ with input $(Y, \{\langle a'_j \rangle_2\}_{j\in[\beta]})$ and appends the outputs to the view.
3. Sim_R computes a length-β Boolean vector b. For $i \in [\beta]$: If there is an element in the i-th bin in HT_2, say y_j, then $b_i = \langle a'_i \rangle_2 \oplus F'_j$. Otherwise, b_i equals the output of $\mathcal{M}_{RR}$ with input $\langle a'_i \rangle_2$.

The real view of $\mathcal{P}_2$ in an execution can be written as:

$$\texttt{view}_2^{\pi} = \{Y, HT_2, \{\langle a_i \rangle_2\}_{i\in[\beta]}, \{\langle \tilde{a_i} \rangle_1\}_{i\in[\beta]}, F\}m$$

and the output view of Sim_R is:

$$S_2 = \{Y, HT_2, \{\langle a'_i \rangle_2\}_{i\in[\beta]}, \{b_i\}_{i\in[\beta]}, F'\}.$$

Two views are statically indistinguishable, which directly follows the security of the underlying protocol realizing $\mathcal{F}_{\mathsf{circuit\text{-}PSI}}$.

Also, the simulator of the corrupt sender can be easily constructed by simulating β random bits and invoking the simulator of $\mathcal{F}_{\mathsf{circuit\text{-}PSI}}$. Thus, we conclude the proof.

B Proof of Theorem 4

Since both circuit-DPSI and mqRPMT-DPSI perturb the intersection in the same way using $\mathcal{M}_{RR}$, we omit the analysis of the DP property of $\mathcal{P}_2$'s output and prove that the information leaked to $\mathcal{P}_1$ in Fig. 12 is bounded by $(\epsilon_I + \epsilon_D, 2\delta)$-DP. Also, according to the sequential composition of DP ((Theorem 1)), we only need to prove Lemma 4.

$\mathcal{F}_{\mathsf{rDPSI}}$

Parameters: $\mathcal{P}_1$'s input set size n. $\mathcal{P}_2$'s input set size m. The privacy budget $\epsilon_I, \epsilon_D, \epsilon$.
Input: $\mathcal{P}_1$'s set $X = \{x_i\}_{i\in[n]} \in \mathcal{V}^n$. $\mathcal{P}_2$'s set $Y = \{y_i\}_{i\in[m]} \in \mathcal{V}^m$.
Functionality: Upon receiving X from $\mathcal{P}_1$, Y from $\mathcal{P}_2$.

1. Compute $F' \in \{0,1\}^m$, where f'_i indicates if $y_i \in X$. Apply $\mathcal{M}_{RR}$ with parameter ϵ to F' and send the result F to $\mathcal{P}_2$.
2. **DP Leakage:** Sample r_I, r_D with $L(\epsilon_I, \delta, 1)$ and $L(\epsilon_D, \delta, 1)$. Compute and send $n_I = |X \bigcap Y| + r_I$, $n_D = Y - |X \bigcap Y| + r_D$ to $\mathcal{P}_1$.

Fig. 12: Ideal functionality $\mathcal{F}_{\mathsf{rDPSI}}$ with privacy leakage.

Lemma 4. *The truncated discrete Laplace mechanism satisfies* (ϵ, δ)*-DP.*

Proof. For any neighboring database D_1, D_2, let $c_1 = \mathbf{c}(D_1) \geq 0$ and $c_2 = \mathbf{c}(D_2) \geq 0$. W.L.O.G. we consider $c_2 \geq c_1$. It is easy to see that $\Pr[TLap(D_1) \in (-\infty, c_1)] = 0$ and $\Pr[TLap(D_2) \in (-\infty, c_2)] = 0$. For any $o \geq c_2 \geq c_1$, it is true that

$$\begin{aligned}
&\frac{\Pr[TLap(D_2) = o]}{\Pr[TLap(D_1) = o]} = \frac{p \cdot \exp(-(\epsilon/\Delta\mathbf{c})|o - c_1 - \eta^0|)}{p \cdot \exp(-(\epsilon/\Delta\mathbf{c})|o - c_2 - \eta^0|)} \\
&= \exp(\frac{\epsilon(|o - c_2 - \eta^0| - |o - c_1 - \eta^0|)}{\Delta\mathbf{c}}) \\
&\leq \exp(\frac{\epsilon|c_1 - c_2|}{\Delta\mathbf{c}}) \leq \exp(\frac{\epsilon \cdot \Delta\mathbf{c}}{\Delta\mathbf{c}}) = \exp(\epsilon).
\end{aligned} \tag{3}$$

However, when the output $o \in [c_1, c_2)$, $\Pr[TLap(D_2) = o] = 0$, $\Pr[TLap(D_1) = o] > 0$, making the ratio of probabilities unbounded. Nevertheless, we can bound $\Pr[TLap(D_1) \in [c_1, c_2)]$ by δ as shown below.

Let $O^* = (-\infty, c_2)$. Then, we can show that for any output set O of query $\mathbf{c}()$, we have

$$\begin{aligned}
&\Pr[TLap(D_1) \in O] \\
&= \Pr[TLap(D_1) \in (O \cap O^*)] + \Pr[TLap(D_1) \in (O - O^*)] \\
&\leq \Pr[TLap(D_1) \in [c_1, c_2)] + e^\epsilon \Pr[TLap(D_1) \in (O - O^*)] \\
&= \Pr[\eta_1 < \Delta\mathbf{c}] + e^\epsilon \Pr[TLap(D_1) \in O] \\
&= 1 - \Pr[\eta_1 \geq \Delta\mathbf{c}] + e^\epsilon \Pr[TLap(D_1) \in O] \\
&= 1 - \frac{e^{-\epsilon(\Delta\mathbf{c} - \eta^0 - 1)}}{e^\epsilon + 1} + e^\epsilon \Pr[TLap(D_1) \in O] \\
&= \delta + e^\epsilon \Pr[TLap(D_1) \in O].
\end{aligned} \tag{4}$$

Therefore, this mechanism satisfies (ϵ, δ)-DP.

Lemma 5. *The protocol in Fig. 8 securely realizes the functionality in Fig. 12 against a semi-honest adversary in the $\mathcal{F}_{mqRPMT}$-hybrid model.*

Proof. We can prove Lemma 5 by constructing the simulators similar to the proof of Lemma 3.