Practical Traceable Over-Threshold Multi-Party Private Set Intersection

Le Yang*, Weijing You†✉, Huiyang He*, Kailiang Ji‡ and Jingqiang Lin*✉
*School of Cyber Science and Technology, University of Science and Technology of China
†Fujian Provincial Key Laboratory of Network Security and Cryptology,
College of Computer and Cyber Security, Fujian Normal University
‡NIO Inc.
{leyang, hhe}@mail.ustc.edu.cn, youweijing@fjnu.edu.cn,
kailiang.ji@nio.com, linjq@ustc.edu.cn

***Abstract*—Multi-Party Private Set Intersection (MP-PSI) with threshold enhances the flexibility o f M P-PSI b y d isclosing elements present in at least t participants' sets, rather than requiring elements to appear in all n sets. In scenarios where each participant is responsible for its dataset, e.g., digital forensics, MP-PSI with threshold is expected to disclose both intersection elements and corresponding holders such that elements are traceable and hence the reliability of intersection is guaranteed. We refer to MP-PSI with threshold supporting traceability as Traceable Over-Threshold Multi-Party Private Set Intersection (T-OT-MP-PSI). However, research on such protocols remains limited, and the current solution is resistant to $t-2$ semi-honest participants at the cost of considerable computational overhead.**

In this paper, we propose two novel Traceable OT-MP-PSI protocols. The first protocol is the Efficient Traceable OT-MP-PSI (ET-OT-MP-PSI), which combines Shamir's secret sharing with oblivious programmable pseudorandom function, achieving significantly improved efficiency with resistance to at most $t-2$ semi-honest participants. The second one is the Security-enhanced Traceable OT-MP-PSI (ST-OT-MP-PSI), which achieves security against up to $n-1$ semi-honest participants by further leveraging oblivious linear evaluation protocol.

Compared to the recent Traceable OT-MP-PSI protocol by Mahdavi et al., our protocols eliminate the security assumption that certain special parties do not collude and provide stronger security guarantees. We implemented our proposed protocols and conducted extensive experiments under various settings. We compared the performance of our protocols with that of Mahdavi et al.'s protocol. While our Traceable OT-MP-PSI protocols enhance security, experimental results demonstrate high efficiency. For instance, given 5 participants, with threshold of 3, and set sizes are 2^{14}, our ET-OT-MP-PSI protocol is 15056× faster, and the ST-OT-MP-PSI is 505× faster, compared to Mahdavi et al.'s protocol.

I. Introduction

Multi-Party PSI (MP-PSI) allows three or more participants, each holding a private set, to learn nothing but the intersection of their sets. Currently, MP-PSI is widely used in privacy-sensitive domains, such as cache sharing in edge computing [1], federated learning [2], [3], and anomaly detection [4], [5], [6]. Implicitly, typical MP-PSI only identifies elements that are possessed by all participants. For participants interested in computing the intersection over elements held not necessarily by all, but by at least a predefined number of participants, we have the MP-PSI with threshold.

Intuitively, MP-PSI with threshold should provide ideal privacy protection, and hence most existing MP-PSI with threshold protocols [7], [8], [9] are fully anonymous, i.e., they reveal nothing but the intersection itself. However, in certain scenarios, full anonymity may be problematic:

- *Network Anomaly Detection:* In distributed environments, anomaly detection systems deployed at different nodes independently report suspicious behaviors. Previous works [10], [11] have demonstrated that the traceability of alerts across distributed detectors significantly improves anomaly attribution and operational response. However, conventional MP-PSI with threshold lacks such a capability, thereby impeding accurate localization and coordinated mitigation.
- *Digital Forensics Investigation:* In digital forensics, the evidence is distributed among multiple entities. Studies [12], [13], [14] emphasize that establishing the provenance of digital evidence is essential for constructing reliable evidence chains, ensuring admissibility in court, and coordinating multi-agency investigations. Although conventional MP-PSI with threshold typically lacks traceability, this prevents investigators from identifying which parties hold the evidence.
- *Suspicious Account Analysis:* Taking anti-money laundering (AML) as an example, Kings Research projects that the global AML market will reach \$9.692 billion by 2031 [15]. As part of practical AML initiatives, the Hong Kong Monetary Authority's AMLab leverages network analysis to identify mule accounts. However, the lack of traceability in fully anonymous MP-PSI with threshold hinders effective cross-institutional collaboration, which is explicitly emphasized as a critical requirement in AML guidelines [16].

We observe that, when participants are accountable for their private sets, which are contributed to the community decision, traceability for each intersection element becomes necessary. Therefore, Over-Threshold MP-PSI with traceabil-

Network and Distributed System Security (NDSS) Symposium 2026
23-27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.230038
www.ndss-symposium.org

ity was introduced [17], extending conventional MP-PSI with threshold by not only identifying elements appearing in at least t participants' sets, but also disclosing the identities of the parties holding each intersecting element. However, the existing solution [17] remains limited in both efficiency and security. Specifically, in terms of security, it only resists collusion among up to $t-2$ semi-honest participants under the assumption that certain special parties do not collude. In terms of efficiency, its computational complexity is $O(m(n\log(\frac{m}{t}))^{2t})$, which grows exponentially with the threshold t, where n is the total number of parties and m is the set size. Furthermore, empirical results demonstrate poor performance in practice. For example, when the number of participants is $n = 10$, the threshold is $t = 7$, and the set size is 2^5, the runtime of their protocol exceeds 9 hours in our experiments.

In this paper, we refer to this functionality as **Traceable Over-Threshold Multi-Party Private Set Intersection (T-OT-MP-PSI)** and propose two protocol instantiations: the Efficient Traceable OT-MP-PSI (ET-OT-MP-PSI) and the Security-enhanced Traceable OT-MP-PSI (ST-OT-MP-PSI). The ET-OT-MP-PSI achieves high efficiency with resistance to $t-2$ collusion among participants, while the ST-OT-MP-PSI is secure against collusion by up to $n-1$ participants in the semi-honest adversary model. Here, t denotes the predefined threshold for intersection computation, and n denotes the total number of participants.

A. *The high-level idea of our protocols*

To enable a Traceable OT-MP-PSI, we first consider two basic functionalities: revealing elements and corresponding holders only when the number of holders is at least the threshold. We adopt Shamir's secret sharing, which is also utilized in Mahdavi et al. [17]. In our design, each intersection element is treated as "secret" to be shared and can be reconstructed when enough "shares" from participants are collected. To preserve privacy, the intersection elements should not be revealed during such a secret sharing process. Therefore, we incorporate the Oblivious Programmable Pseudorandom Function (OPPRF) to transmit "shares". The OPPRF ensures that only participants with the same element receive the correct "share", whereas others receive random values.

Each participant may have elements that are not in the final intersection. Note that in the MP-PSI with threshold, only elements possessed by enough participants are revealed, while those appearing in fewer sets should remain private. However, given Shamir's secret sharing and OPPRF, it is easy to identify elements in common by simply observing the "shares" from different participants, even if those elements do not belong to the intersection. This problem was addressed by Mahdavi et al. [17] using homomorphic encryption, which results in high computational cost. In this work, inspired by Herzberg et al. [18], we update the element "shares" with additional shares of zero-value secret which does not modify the underlying secret nor the threshold value.

Efficient Traceable OT-MP-PSI. Based on Shamir's secret sharing, OPPRF and secret shares update, we first propose an Efficient Traceable OT-MP-PSI (ET-OT-MP-PSI) protocol. It is initialized by one participant, who then interacts with the other participants to derive the final intersection in the following three phases: 1) *conditional secret sharing,* in which elements in the set are transformed into secret shares and securely transmitted among all participants using OPPRF; 2) *secret shares update,* in which each participant individually updates their own shares using new shares of zero-value secret before collection; 3) *conditional collection and reconstruction,* in which OPPRF is used again by the leader participant to collect shares, and the elements are reconstructed to obtain the intersection and corresponding holders.

ET-OT-MP-PSI is an extension of MP-PSI (CCS'17) [4] to MP-PSI with threshold. However, based on the Lagrange Interpolation theorem, this extension is only secure against collusion among up to $t-2$ participants. Nevertheless, for stricter security requirements in practice, *is it possible to remove the dependence between security and the threshold?*

Security-enhanced Traceable OT-MP-PSI. In the ET-OT-MP-PSI, participants are able to access shares directly derived from threshold Shamir's secret sharing. Consequently, if more than $t-2$ corrupted participants collude, they can infer the private information of honest parties, thereby compromising the security of the protocol. To address this, we introduce the Oblivious Linear Evaluation (OLE) protocol to enable a three-party interaction during the shares update phase, thereby further imposing requirements on the elements held by participants. That is, successful reconstruction requires not only collecting enough shares, but also ensuring that a sufficient number of parties hold the same element. Hence the resistance of our Traceable OT-MP-PSI to semi-honest adversaries is extended from $t-2$ to $n-1$.

B. *Our Contributions*

Our contributions can be summarized as follows:

- **Efficient Traceable OT-MP-PSI.** We revisit full anonymity and traceability of MP-PSI with threshold from a practical perspective, and propose an Efficient Traceable OT-MP-PSI protocol (ET-OT-MP-PSI) based on threshold Shamir's secret sharing and OPPRF. The ET-OT-MP-PSI is efficient and resistant to collusion among up to $t-2$ semi-honest participants.
- **Security-enhanced Traceable OT-MP-PSI.** We introduce a Security-enhanced Traceable OT-MP-PSI protocol (ST-OT-MP-PSI), which incorporates the OLE protocol into the ET-OT-MP-PSI. This protocol could tolerate at most $n-1$ semi-honest adversaries at a modest performance cost.
- **Security analysis, implementation and performance evaluation.** We conduct a security analysis of the two proposed protocols. In addition, we implement and evaluate them under various experimental settings. Experimental results demonstrate that both of our protocols outperform the recent work with similar functionality.

II. Preliminaries

A. Notations

Notations	Descriptions
n	The number of parties
m	Set size of each party
t	The threshold value
$[c, d]$ or $[a]$	Denotes the set $\{c, c+1, ..., d\}$ or $\{0, 1, ..., a\}$
P_i	The party with index $i, i \in [n]$
S_i	The set of party P_i, i.e., $S_i = \{e_0^i, \dots, e_{m-1}^i\}$
m_b	The size of the Simple or Cuckoo hashing table
λ	The statistical security parameter
κ	The computational security parameter
$B_S[b]$ or $B_C[b]$	The bth bin of Simple or Cuckoo hashing table
$x \leftarrow \mathbb{F}_p$	x is sampled uniformly over the field $\mathbb{F}_p$

B. Security Model

Consistent with the majority of the prior research on MP-PSI [4], [5], [8], [19], [20], our proposed protocols primarily focus on the *semi-honest* adversarial model [21]. In this model, adversaries may try to learn as much information as possible from the protocol execution but will not deviate from the execution steps. These adversaries are also referred to as honest-but-curious.

The view of a party consists of its private input, its random tape , and the list of all messages received during the protocol. The view of the adversary comprises the combined views of all corrupted parties, potentially allowing multiple colluding parties to aggregate their information and infer private data. To prove the security of a protocol, it is common to construct a simulator Sim that, given the adversaries' inputs X and outputs Y, generates simulated views that are computationally indistinguishable from the adversaries' real views in the protocol execution [4], [22], [20], [23].

Definition 1. ***Semi-honest Secure.*** *A protocol π securely realizes functionality $\mathcal{F}$ in the presence of semi-honest adversaries if there exists a simulator* Sim *such that, for any subset of corrupt parties $\{P_i \in \mathbb{C}\}$, the views $\{\mathsf{Sim}(X, Y, \mathbb{C})\}$ generated by the simulator are computationally indistinguishable from the views $\{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}$ obtained by the adversaries in the real execution. Formally, this can be expressed as:*

$$\{\mathsf{Sim}(X, Y, \mathbb{C})\} \overset{c}{\equiv} \{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}.$$

C. Shamir's Secret Sharing

In the (t, n)-Shamir's secret sharing scheme, the dealer distributes the secret S to n participants, with each participant possessing a share of the secret. When t or more participants collaborate, they reconstruct the secret S together. If fewer than t participants are involved, they will not gain any information about the secret.

1) Secret sharing and Reconstruction: Shamir's secret sharing [24] is a (t, n)-threshold secret sharing scheme. During the secret distribution phase, the dealer chooses a prime number p and randomly picks $t-1$ numbers $a_i, i \in [t-1]$ from the finite field $\mathbb{F}_p$ to construct polynomial with secret S :

$$f(x) = S + a_1 x + \cdots + a_{t-1} x^{t-1}.$$

For each participant i, the dealer evaluates the polynomial and distributes the secret share (x_i, y_i), where $y_i = f(x_i)$ and $x_i \neq 0$. Without loss of generality, and for the sake of clarity, we define $x_i = i+1$ throughout the protocol. This scheme uses Lagrange interpolation theorem. Specifically, t points on the polynomial can uniquely determine a polynomial with degree equal to or less than $t-1$. Given any t secret shares, secret S can be reconstructed by Lagrange interpolation:

$$S = f(0) = \sum_{i=0}^{t-1} y_i \prod_{j=0, j\neq i}^{t-1} (\frac{x_j}{x_j - x_i}).$$

2) Secret shares update: To update the secret shares held by participants, each participant generates a random polynomial $f'(x)$ with a constant term of 0, expressed as

$$f'(x) = 0 + b_1 x + \cdots + b_{t-1} x^{t-1}.$$

Using this polynomial, the participant computes an update share (x_i, y_i') for each participant i, where $y_i' = f'(x_i)$ and $x_i \neq 0$. Similarly, for consistency and ease of presentation, we define $x_i = i + 1$ in our protocol. Upon receiving this update share, each participant updates their secret share by setting $(x_i, y_i + y_i')$ as the new share. Essentially, this process performs a Shamir's secret sharing with a secret value of 0, ensuring that the correctness of reconstruction remains intact while updating the secret shares. This approach is similar to the proactive secret sharing scheme proposed by Herzberg et al. in [18].

D. Hashing Schemes

1) Simple Hashing: In Simple hashing, the hash table consists of m_b bins $B[0], \dots, B[m_b - 1]$. By uniformly selecting a hash function $h : \{0,1\}^* \to [0, m_b - 1]$ at random, element e is mapped to bin $B[h(e)]$ in the hash table and inserted to hash table by appending it to this bin. Obviously, there are multiple elements in the same bin. In practice, multiple hash functions are often employed to reduce the probability of collision and improve load balancing.

2) Cuckoo Hashing: Cuckoo hashing scheme uses hash function $h_1, \dots, h_k : \{0,1\}* \to [m_b]$ to map m elements to m_b bins in hash table. Unlike Simple hashing, Cuckoo hashing is only allowed to store one element per bin. One variant of Cuckoo hashing is Cuckoo hashing with a stash. To insert an element e into hash table do the following [25]: (1) If one of bin $B[h_1(e)], \dots, B[h_k(e)]$ is empty, insert element e into the empty bin. (2) Otherwise, the element e is inserted into the bin $B[h_1(e)]$, evicting its existing content o. The evicted element o is then relocated to a new bin $B[h_i(o)]$, using h_i to determine the new bin location, where $h_i(o) \neq h_1(e)$ for $i \in [1, \dots, k]$. The procedure is repeated until no more evictions are necessary, or until a threshold number of relocations been performed. In the latter case, the last element is placed in a stash. After Cuckoo hashing, element e can be found in the one of following locations: bin $B[h_1(e)], \dots, B[h_k(e)]$ or stash.

Another variant of Cuckoo hashing, as proposed in [4], eliminates the use of a stash by employing two hash tables.

This design avoids the inefficiencies associated with a stash, where every item in one party's stash need to be compared to every item of another party, increasing overhead. Specifically, the procedure starts by using three "primary" Cuckoo hash functions to determine the placement of an element. If these initial attempts are unsuccessful, the process resorts to two "supplementary" Cuckoo hash functions as a fallback mechanism. By adjusting the parameters within the hashing scheme, the process can be ensured to succeed with a negligible failure probability, specifically less than $2^{-\lambda}$.

E. Oblivious Programmable Pseudorandom Function

Oblivious pseudorandom function (OPRF) [26] is a two-party protocol through which the sender learns a pseudo-random function (PRF) key k, and the receiver learns $F(k, q_1), \ldots, F(k, q_v)$, where F is a pseudo-random function and $(q_1, \ldots, q_v)$ are the receiver's inputs. If the sender's input x matches the receiver's input q_i, the sender can compute $F(key, x)$, which equals $F(\text{key}, q_i)$, under the key k.

Oblivious programmable pseudorandom function (OPPRF) [4] is similar to OPRF, with the additional property that on a certain programmed set of inputs the function outputs programmed values. In the OPPRF, the sender inputs a set of points $\{(x_1, y_1), \ldots, (x_u, y_u)\}$, and the receiver inputs $(q_1, \ldots, q_v)$. By running protocol, the receiver ultimately obtains OPPRF output $(hint, F(k, hint, q_1), \ldots, F(k, hint, q_v))$ and the sender gets $(k, hint)$. Within the protocol, when the receiver's input q_i equals the sender's input x_j (i.e., $q_i = x_j$), the receiver is able to obtain the value y_j, which has been programmed by the sender. For the receiver, it is indistinguishable whether the obtained output is a random value or a value programmed by the sender; meanwhile, the sender remains oblivious to the receiver's input. The functionality of OPPRF is presented in Fig. 1.

Kolesnikov et al. proposed three instantiation methods for OPPRF, among which the table-based construction has favorable communication and computational cost [4]. However, this construction allows the receiver to evaluate the programmable PRF on only $v = 1$ point. Therefore, they extended this construction using hashing schemes to support both a large u (the number of programmed points) and a large v (the number of queries). Specifically, in the OPPRF protocol, the sender uses Simple hashing to map $x_i, i \in [1, u]$ into m_b bins, while the receiver uses Cuckoo hashing to map $q_j, j \in [1, v]$ into m_b bins. Now in each bin, the receiver has at most one item q. Therefore, they can run the table-based OPPRF protocol on these inputs. They refer to this protocol as the hashing-based OPPRF protocol.

F. Oblivious Linear Evaluation

Oblivious linear evaluation (OLE) is a two-party protocol and serves as a fundamental building block in multi-party secure computation protocols [27]. In the OLE protocol, the sender inputs a and b, where a and b are elements of a finite field $\mathbb{F}$. The receiver inputs $x \in \mathbb{F}$ and ultimately receives $y \in \mathbb{F}$ such that $y = ax + b$. Throughout the protocol, the sender remains oblivious to the receiver's input x, and the receiver does not learn any information about the sender's inputs a and b.

PARAMETERS:
- A programmable pseudorandom function F.
- The upper bound u on the number of points to be programmed.
- The bound v on the number of queries.

INPUT:
- The sender inputs points $\{(x_1, y_1), \ldots, (x_u, y_u)\}$.
- The receiver inputs $(q_1, \ldots, q_v)$.

OUTPUT:
- For each q_i, if $q_i = x_j$, the receiver obtains y_j; otherwise, the receiver receives a random value.

Fig. 1: The OPPRF functionality.

Vector OLE (VOLE) is the vectorized variant of the OLE protocol, allowing the receiver to learn a linear combination of two vectors held by the sender. Specifically, the sender inputs vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{F}^n$, while the receiver inputs $x \in \mathbb{F}$ and obtains $\boldsymbol{y} \in \mathbb{F}^n$ where $\boldsymbol{y} = \boldsymbol{\alpha} x + \boldsymbol{\beta}$.

Batch OLE (BOLE) is similar to VOLE but extends it by allowing the receiver's input to also be a vector. Specifically, the receiver inputs a vector $\boldsymbol{x} \in \mathbb{F}^n$ and obtains $\boldsymbol{y} \in \mathbb{F}^n$ where $\boldsymbol{y} = \boldsymbol{\alpha}\boldsymbol{x} + \boldsymbol{\beta}$.

III. Traceable Over-Threshold Multi-Party Private Set Intersection

A. Functionality Definition

Before introducing our proposed protocols, we first define the functionality of protocols. The protocols require $n \geq 3$ parties, denoted as $P_0, \ldots, P_{n-1}$, each holding a private set of size m, denoted as $S_0, \ldots, S_{n-1}$, along with a threshold value t. The ultimate goal of the protocols is to enable party P_0 to obtain the following information:

- **Intersection elements:** Party P_0 identifies each element e_i from its own set S_0 that satisfy the threshold condition $c_i \geq t$, where t is the predefined threshold value and c_i denotes the number of parties holding the element e_i.
- **Identity of element holders:** The protocol reveals the specific participants $\{P_j\}$ holding each intersection element.
- **Counting each intersection element:** Since the identities of the holders are disclosed, P_0 naturally infers the number of each intersection element.

We refer to such protocols as the **Traceable Over-Threshold Multi-Party Private Set Intersection**. The ideal functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$ is formally described in Fig. 2.

B. Protocol Overview

Motivated by the limitations of fully anonymous MP-PSI with threshold in regulatory scenarios, we aim to develop Traceable OT-MP-PSI. Considering the characteristics of the protocol, the first challenge arises:

PARAMETERS:

- $n \geq 3$ parties $P_0, \dots, P_{n-1}$, with their respective private sets $S_0, \dots, S_{n-1}$ of size m.

INPUT:

- Each party P_i has a private set S_i as input.
- A threshold value t, where $1 < t \leq n$.

OUTPUT:

- P_0 outputs the intersection set $I = \{(e_i, c_i, \{P_j\})|, e_i \in S_0, c_i \geq t\}$, where e_i is the intersection element, c_i is the number of parties holding element e_i, and $\{P_j\}$ is the set of these parties.
- $P_1, \dots, P_{n-1}$ outputs $\perp$.

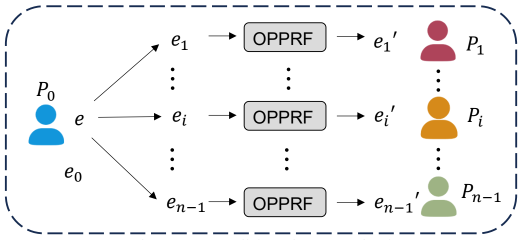
Fig. 2: Traceable OT-MP-PSI functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$.

Challenge 1: *How can we design an MP-PSI with threshold that supports traceability?*

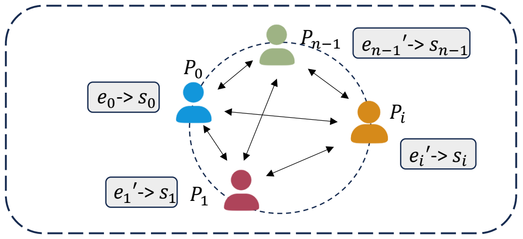
Motivated by Mahdavi et al. (ACSAC'20) [17], we observe that the traceability in MP-PSI with threshold can be achieved by combining Shamir's secret sharing with the additively homomorphic Paillier cryptosystem. However, the traceable MP-PSI construction in [17] still exhibits limitations in both security and performance. Specifically, even assuming no collusion across certain special parties, the protocol is secure against collusion among up to $t-2$ participants, where t is the threshold. For performance, take 10 participants with a threshold of 7 and a set size of 2^5 as an example. In this case, the protocol will consume more than 9 hours. Therefore, we take a step back and are inspired by the work of Kolesnikov et al. (CCS'17) [4]. An approach to extending traditional MP-PSI to MP-PSI with threshold and traceability is to combine Shamir's secret sharing with Oblivious Programmable Pseudorandom Function (OPPRF).

Specifically, each element in P_0's set is processed into distinct shares and assigned to specific participants, enabling *traceability*. To obtain the intersection, during the distribution and collection of shares, shares of each potential element should be received by target participants holding the same element, which is achieved through OPPRF. If the final collected shares for an element successfully reconstruct the original secret, it indicates that at least t participants hold the element. This realizes the *threshold functionality*, confirms its inclusion in the intersection and reveals corresponding holders.

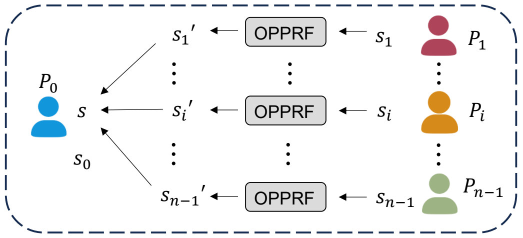
Nevertheless, such a scarecrow Traceable OT-MP-PSI may expose the privacy of participants' sets. Since party P_0, which initializes the protocol, holds all the shares of the secret, it is able to infer whether other participants hold elements not in the intersection. This is achieved by simply comparing the initial shares it sent with the final shares it received, even if the process involves OPPRF. Such inference violates the general requirement for privacy protection in MP-PSI. So another challenge occurs:



Phase 1: Conditional secret sharing



Phase 2: Secret shares update



Phase 3: Conditional collection and reconstruction

Fig. 3: The overall process of ET-OT-MP-PSI.

Challenge 2: *How can we preserve the privacy of non-intersecting elements while ensuring the functionality of the protocol?*

Updating the secret shares held by each participant is an option. Specifically, we employ zero-value Shamir's secret sharing to update the shares held by each party, effectively refreshing the original secret shares to protect the privacy of participants and ensuring the correctness of secret reconstruction, as outlined in Section II-C2. By leveraging Shamir's secret sharing to update secret shares, we achieve updating while avoiding additional costly operations, thereby proposing the Efficient Traceable OT-MP-PSI (ET-OT-MP-PSI).

In such an extension from MP-PSI [4] to MP-PSI with threshold that supports traceability, each party can directly obtain the correct values for updating their shares. As a result, any collusion of $t-1$ corrupted parties (including P_0) compromises the privacy of the honest parties. Consequently, the ET-OT-MP-PSI can only withstand collusion by up to $t-2$ parties. The overall process of the protocol is illustrated in Fig. 3.

To meet higher security requirements, we now take a step forward in the level of security. A Traceable OT-MP-PSI protocol should be robust against collusion among an arbitrary number of participants. We now face the final challenge:

Challenge 3: *How can we design a Traceable OT-MP-PSI that is secure against arbitrary collusion among semi-honest participants?*

To enhance security, we introduce the OLE protocol to enable a three-party interaction during the shares update phase. In this design, party P_i receives the correct values for updating shares from party P_j only if P_i holds the same element as party P_0. Hence, $t-1$ colluding parties can infer whether the honest party P_i holds a specific element e only if all of the colluding parties also hold the same element. When $n-1$ parties collude, if fewer than $t-1$ of them possess the element e, the colluding parties are unable to compromise the protocol's security. Conversely, if at least $t-1$ of the colluding parties hold e, the protocol output reveals whether e is part of the intersection and discloses its associated holders. Since this information is explicitly included in the output of the protocol, such inference does not result in any additional privacy leakage. Therefore, the Security-enhanced Traceable OT-MP-PSI is secure against arbitrary collusion among semi-honest participants.

In conclusion, both of our proposed Traceable OT-MP-PSI protocols consist of the following three main phases:

- **Conditional secret sharing.** Elements in the set are transformed into secret shares and securely transmitted among all participants using OPPRF.
- **Secret shares update.** Each participant individually updates their own shares using new shares of zero-value secret before collection.
- **Conditional collection and reconstruction.** Use OPPRF again to collect shares and reconstruct elements to obtain the intersection and corresponding holders.

In the following sections, we present the Efficient Traceable OT-MP-PSI (ET-OT-MP-PSI) and the Security-enhanced Traceable OT-MP-PSI (ST-OT-MP-PSI) in detail.

C. Details of the Efficient Traceable OT-MP-PSI

We now present our first protocol, ET-OT-MP-PSI. A formal description of the protocol is in Fig. 4. Prior to the protocol execution, each participant maps their set using both Cuckoo hashing and Simple hashing, as described in Section II-D. When applying Cuckoo hashing to the set elements, if the b^{th} bin is empty, it is padded with a dummy element. Similarly, when using Simple hashing, each bin is padded with dummy elements so that its total size reaches the maximum bin size β. we adopt the method by Kolesnikov et al. [4] for calculating β. The padding is performed to hide the number of elements that were mapped to a specific bin, which would leak information about the input.

In the conditional secret sharing phase, for each element $e_k^0, k \in [m]$ in S_0, P_0 performs (t,n)-Shamir's secret sharing, generating n shares $s_k^{0,0}, \ldots, s_k^{0,n-1}$. Subsequently, P_0 executes the OPPRF protocol with each of the other parties $P_i, i \in [1, n-1]$, following Section II-E. P_0 programs the

PARAMETERS:

- n parties $P_0, \ldots, P_{n-1}$.
- A prime p, used for (t,n)-Shamir's secret sharing.
- Hash functions for hashing scheme.

INPUT:

- Each party P_i uses its private set S_i as input.
- The threshold t.

PROTOCOL:

<u>*Conditional Secret Sharing*</u>

(1) Each party maps its set S_i into bins using Cuckoo and Simple hashing schemes, obtaining $B_C[\cdot]$ and $B_S[\cdot]$. Each empty bin in $B_C[\cdot]$ is padded with a dummy element, while each bin in $B_S[\cdot]$ is padded with dummy elements to the maximum bin size β.

(2) For each element $e_k^0 \in S_0, k \in [m]$, P_0 treats it as a secret and performs (t,n)-Shamir's secret sharing, generating n shares $s_k^{0,0}, \ldots, s_k^{0,n-1}$.

(3) For the b^{th} bin, $b \in [m_b]$, P_0 invokes an OPPRF protocol with every other party P_i, where $i \in [1, n-1]$.
 - P_0 is the sender with input $\{(e_k^0, s_k^{0,i}) | e_k^0 \in B_S[b]\}$.
 - P_i acts as receiver with input $\{e_k^i | e_k^i \in B_C[b]\}$. As a result, for every $e_k^i \in S_i$, P_i obtains a corresponding OPPRF output, denoted as $\hat{s}_k^{0,i}$.

<u>*Secret Shares Update*</u>

(4) For the b^{th} bin, each $P_i, i \in [1, n-1]$ performs secret shares update mentioned in Section II-C2 to get shares $(j+1, f_{i,b}(j+1))$ and directly sends to $P_j, j \in [n]$.

(5) Upon receiving the values sent by the other participants, for the b^{th} bin, party P_j sums the $n-1$ values to obtain $\delta_b = f_{1,b}(j+1) + \cdots + f_{n-1,b}(j+1)$. For $e_k^0 \in B_C[b]$, P_0 updates its shares: $y_k^0 = s_k^{0,0} + \delta_b$.

<u>*Conditional Collection and Reconstruction*</u>

(6) For the b^{th} bin, each pair of P_i and P_0 invokes an OPPRF.
 - P_i is the sender with input $\{(e_k^i, \mu_k^{0,i}) | e_k^i \in B_S[b]\}$, where $\mu_k^{0,i} = \hat{s}_k^{0,i} + \delta_b$.
 - P_0 is the receiver with input $\{e_k^0 | e_k^0 \in B_C[b]\}$ and obtains y_k^i which represents the corresponding OPPRF output.

(7) For each element $e_k^0 \in S_0$, P_0 applies Lagrange interpolation over all subsets of t shares among the n values y_k^i, always including its own share, to compute $Recon(y_k^i) = f_k(\cdot)$. If $f_k(0) = e_k^0$, then e_k^0 is identified as an intersection element. Subsequently, P_0 determines all the holders of e_k^0 by checking whether $f_k(i+1) = y_k^i$ holds for each $i \in [1, n-1]$. Finally, P_0 obtains the intersection set I.

Fig. 4: ET-OT-MP-PSI protocol.

OPPRF using $\{(e_k^0, s_k^{0,i}) | k \in [m]\}$, and P_i acts as the receiver with input S_i. After the OPPRF is executed, each party P_i obtains a corresponding OPPRF output for each e_k^i, denoted as $\hat{s}_k^{0,i}$. According to the properties of the OPPRF protocol, if $e_k^0 = e_k^i$, then $s_k^{0,i} = \hat{s}_k^{0,i}$. Moreover, P_i does not know whether the received values are the real shares or random values.

In the shares update phase, $P_i, i \in [1, n-1]$ performs secret shares update as described in Section II-C2. Specifically, for the the b^{th} bin, P_i generates a random polynomial with a constant term of 0, a degree of $t-1$:

$$f_{i,b}(x) = 0 + a_1 x + \cdots + a_{t-1} x^{t-1}.$$

Subsequently, P_i generates the corresponding secret shares $(j+1, f_{i,b}(j+1))$ for each party and directly sends the secret shares to the other parties $P_j, j \in [n]$. For the b^{th} bin, P_j sums $n-1$ obtained shares to obtain values δ_b, for updating the original secret shares. Then, for $e_k^0 \in B_C[b]$, P_0 updates its shares: $y_k^0 = s_k^{0,0} + \delta_b$.

Finally, for each element e_k^0, P_0 enumerates all possible subsets of t shares from the n values and applies Lagrange interpolation to each subset to compute $Recon(y_k^i) = f_k(\cdot)$. This is equivalent to selecting all subsets of $t-1$ shares from the remaining $n-1$ values, since P_0's own share is always correct and included in each reconstruction attempt. If any reconstruction satisfies $f_k(0) = e_k^0$, it indicates that e_k^0 is an intersection element. Furthermore, P_0 identifies all holders of this element by checking whether $f_k(i+1) = y_k^i$ holds for each $i \in [1, n-1]$.

D. Security-Enhanced Traceable OT-MP-PSI

1) Design Rationale: In ET-OT-MP-PSI, collusion among $t-1$ participants (including P_0) reveals whether an honest party P_i holds a specific element $e \in S_0$ without secret reconstruction.

Specifically, Shamir's secret sharing relies on the Lagrange interpolation theorem, which states that a polynomial of degree at most $t-1$ is uniquely determined by t points. Now, suppose $P_0, \ldots, P_{t-2}$ collude, and P_i is the honest party. In the first phase, P_0 shares the element e as a secret using Shamir's secret sharing, generating a share x_i for P_i. Then, P_i obtains the corresponding OPPRF output y_i. The updated share, denoted as y_i' in the second phase, is calculated as

$$y_i' = y_i + f_1(i+1) + \cdots + f_{n-1}(i+1). \qquad (1)$$

In Equation (1), $f_1(i+1), \ldots, f_{t-2}(i+1)$ are the values generated by the polynomials of parties $P_1, \ldots, P_{t-2}$. The polynomials $f_{t-1}(\cdot), \ldots, f_{n-1}(\cdot)$ can be reconstructed by the $t-1$ colluding parties $P_0, \ldots, P_{t-2}$ according to the Lagrange interpolation theorem, enabling them to obtain the values $f_{t-1}(i+1), \ldots, f_{n-1}(i+1)$. Thus, in the final stage, after the OPPRF execution, P_0 determines whether the honest party P_i holds the element e by verifying if the OPPRF output satisfies:

$$\text{OPPRF output} \overset{?}{=} x_i + f_1(i+1) + \cdots + f_{n-1}(i+1). \qquad (2)$$

If Equation (2) holds, it signifies party P_i possesses the element e.

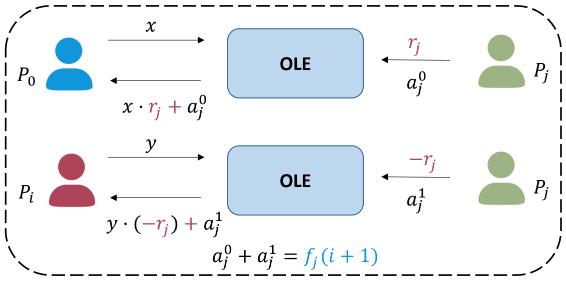


Fig. 5: The core idea of the shares update phase in ST-OT-MP-PSI.

The root cause of such inference is that the values for updating the shares are directly sent to the parties, and reconstruction only relies on zero-value Shamir's secret sharing. Therefore, the security of the protocol can be compromised if $t-1$ corrupted parties collude, i.e., obtain enough shares to reconstruct the polynomial. To prevent such inference, we further impose requirements on the elements held by participants, such that reconstruction is only feasible when a sufficient number of parties possess the same element. That is, successful reconstruction requires not only collecting enough shares but also ensuring that enough parties possess the identical element.

To implement the above idea, OLE is introduced to enable a three-party interaction during the shares update process. Specifically, P_0 and P_i independently perform the OLE protocol with party P_j such that the OLE outputs received by P_0 and P_i collectively produce the correct value for updating P_i's secret share, only if P_0 and P_i hold the same element. Consequently, P_i's share is updated correctly. As shown in Fig. 5, if P_0 and P_i hold the same element, i.e., $x = y$, the sum of their outputs equals $f_j(i+1)$, which is the correct value for updating P_i's corresponding share. By incorporating OLE, we develop the Security-enhanced Traceable OT-MP-PSI (ST-OT-MP-PSI). In this protocol, when $n-1$ parties collude:

- If fewer than $t-1$ of them hold the element e, the colluding parties cannot compromise the protocol's security through the inference described earlier.
- If at least $t-1$ of the colluding parties hold e, they can directly learn from the protocol output whether e is in the intersection, and, if so, identify its holders. Since this information is explicitly included in the output of the protocol, such inference does not lead to any additional privacy leakage.

As a result, ST-OT-MP-PSI is secure against arbitrary collusion in the semi-honest model. The formal security proof is provided in Section IV-A.

2) Details of ST-OT-MP-PSI: The ST-OT-MP-PSI follows the same basic steps as the ET-OT-MP-PSI, consisting of three phases, but there are some differences in certain aspects. The formal description of this protocol is given in Fig. 6.

At the outset, each party maps their set using Cuckoo hashing and Simple hashing, obtaining $B_C[\cdot]$ and $B_S[\cdot]$. Similarly, to prevent information leakage, each party pads bins according to the corresponding hashing scheme.

In the secret sharing phase, P_0 generates a random value $e_k^{0\prime}$ as the secret, which is uniquely mapped to an element e_k^0 in its set S_0, and performs (t,n)-Shamir's secret sharing to obtain n shares $s_k^{0,0},\dots,s_k^{0,n-1}$. Subsequently, these shares are conditionally distributed to other parties through the OPPRF. The reason for not directly using P_0's elements as secrets for secret sharing is to prevent t colluding parties from reconstructing with shares obtained through the OPPRF, which could potentially reveal information about P_0's elements.

In the shares update process, for the b^{th} bin, $P_j, j \in [1, n-1]$ generates a polynomial $f_{j,b}(\cdot)$ and directly sends the value $f_{j,b}(1)$ to P_0. Then, P_0 uses these values to directly update its own share. Afterward, both P_i and P_0, acting as receivers, execute the OLE protocol with each of the other parties $P_j, i,j \in [1,n-1]$. For each bin $B_C[b]$, P_0 and P_j execute the OLE protocol β times, where β is the maximum bin size of B_S. In each execution, P_j, acting as the sender, inputs different r_j^v and $a_j^{0,v}, v \in [\beta]$, while P_0, acting as the receiver, inputs the element e_k^0, which is located in its bin $B_C[b]$. As a result, P_0 obtains $e_k^0 \cdot r_j^v + a_j^{0,v}$. Similarly, for each bin $B_S[b]$, P_i and P_j invoke β instances of the OLE protocol. In each execution, P_j, acting as the sender, inputs $-r_j^v$ and $a_j^{1,v}$, while P_i, acting as the receiver, inputs the element e_k^i from its bin $B_S[b]$ and receives $e_k^i \cdot -r_j^v + a_j^{1,v}$. Here, r_j^v is a random value, and $a_j^{0,v} + a_j^{1,v} = f_{j,b}(i+1)$. After completing the OLE protocol with all parties P_j, P_0 obtains:

$$z_0^v = (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^0 + (a_1^{0,v} + \cdots + a_{n-1}^{0,v})$$

and P_i obtains:

$$z_1^v = -(r_1^v + \cdots + r_{n-1}^v) \cdot e_k^i + (a_1^{1,v} + \cdots + a_{n-1}^{1,v}).$$

At this point, if P_0 and P_i hold the same element, then after completing the third phase, which involves the OPPRF, P_0 obtains:

$$y_k^i = s_k^{0,i} + z_1^v = s_k^{0,i} - (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^i + (a_1^{1,v} + \cdots + a_{n-1}^{1,v}).$$

For each element $e_k^0 \in S_0$, there are β OLE results related to each P_i for P_0. Therefore, it is necessary for P_0 to use the OPPRF with each party P_i to conditionally receive the OLE index v corresponding to e_k^0, so that it can retrieve the corresponding OLE output, denoted as $z_0^v, v \in [\beta]$, and obtain the updated share from P_i. After completing the above steps, we observe that if P_0 and P_i hold the same element, i.e., $e_k^0 = e_k^i$, P_0 obtains the correctly updated share from party P_i. Otherwise, P_0 obtains a random value. The details are as follows:

$$\begin{aligned}
y_k^i &= y_k^i + z_0^v \\
&= s_k^{0,i} + (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^0 + (a_1^{0,v} + \cdots + a_{n-1}^{0,v}) \\
&\quad - (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^i + (a_1^{1,v} + \cdots + a_{n-1}^{1,v}) \\
&= s_k^{0,i} + ((r_1^v + \cdots + r_{n-1}^v) \cdot (e_k^0 - e_k^i) \\
&\quad + ((a_1^{0,v} + a_1^{1,v}) + \cdots + (a_{n-1}^{0,v} + a_{n-1}^{1,v})) \\
&= s_k^{0,i} + (f_{1,b}(i+1) + \cdots + f_{n-1,b}(i+1))
\end{aligned}$$

Ultimately, for each element $e_k^0 \in S_0$, P_0 performs secret reconstruction using the n updated shares y_k^i. In each attempt, P_0 selects a subset of $t-1$ shares from the other $n-1$ parties, combines them with its own share, and applies Lagrange interpolation to compute $Recon(y_k^i) = f_k(\cdot)$. If any reconstruction satisfies $f_k(0) = e_k^0$, then e_k^0 is identified as an intersection element. Furthermore, P_0 identifies all holders of this element by checking whether $f_k(i+1) = y_k^i$ holds for each $i \in [1, n-1]$.

IV. Theoretical Analysis

A. Correctness and Security Analysis

Theorem 1. *The ET-OT-MP-PSI realizes the functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$ and is secure against collusion among up to $t-2$ parties in the semi-honest model, given the statistical security parameter λ and the computational security parameter κ.*

Proof. The proof consists of two parts: correctness and security. Due to the page limit, we put the formal correctness and security proofs of the ET-OT-MP-PSI in Appendix A. □

Theorem 2. *The ST-OT-MP-PSI realizes the functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$ and is secure against collusion among up to $n-1$ parties in the semi-honest model, given the statistical security parameter λ and the computational security parameter κ.*

Proof. Similar to Theorem 1, the proof also consists of correctness and security. For brevity, we defer the full proof of the ST-OT-MP-PSI protocol in Appendix B. □

B. Complexity Analysis

The complexity comparison between our Traceable OT-MP-PSI protocols and related work with traceability [17] is illustrated in Table I. To ensure consistency in comparison, we refer to P_0 as the Leader and all other participants P_i as the Clients throughout this section.

Communication complexity. In ET-OT-MP-PSI, the *conditional secret sharing* phase involves the Leader engaging with $n-1$ Clients to execute the OPPRF protocol, incurring a communication complexity of $O(nm\lambda)$. As part of the *secret shares update* step, each Client transmits m_b values to every other participant to update shares, resulting in an additional communication cost of $O(nm\lambda)$. The subsequent OPPRF with the Leader in the *conditional collection and reconstruction* phase contributes a further $O(m\lambda)$ to the overall communication complexity. In ST-OT-MP-PSI, the *secret share update* procedure requires the Leader to perform $n^2 m_b$ additional OLE protocols, which involve the exchange of $O(n^2 m\lambda)$ ciphertexts. Similarly, each Client executes nm_b OLE protocols, corresponding to $O(nm\lambda)$ ciphertexts.

Computation complexity. In ET-OT-MP-PSI, during *conditional secret sharing*, the Leader generates m polynomials of degree $t-1$ and evaluates them at n points, resulting in a computational complexity of $O(nmt)$. The Leader also

PARAMETERS:

- The same as the protocol described in Fig. 4.

INPUT:

- Each party P_i uses its private set S_i as input.
- The threshold t.

PROTOCOL:

Conditional Secret Sharing

(1) Each party maps their set S_i into bins using Cuckoo and Simple hashing as described in Section II-D, obtaining $B_S[\cdot]$ and $B_C[\cdot]$. Each empty bin in $B_C[\cdot]$ is padded with a dummy element, while each bin in $B_S[\cdot]$ is padded with dummy elements to the maximum bin size β.

(2) For each element $e_k^0 \in S_0$, P_0 generates a corresponding random value $e_k^{0\prime}$. Subsequently, P_0 uses $e_k^{0\prime}$ as the secret and performs a (t, n)-Shamir's secret sharing, generating n shares $s_k^{0,0}, \ldots, s_k^{0,n-1}$.

(3) For the b^{th} bin, $b \in [m_b]$, P_0 and each party $P_i, i \in [1, n-1]$ execute an OPPRF.
   - P_0 is the sender with input $\{(e_k^0, s_k^{0,i}) | e_k^0 \in B_S[b]\}$.
   - P_i is the receiver with input $\{e_k^i | e_k^i \in B_C[b]\}$ and obtains a corresponding output $\hat{s}_k^{0,i}$ for every $e_k^i \in S_i$.

Secret Shares Update

(4) For the b^{th} bin, P_j performs secret shares update mentioned in Section II-C2, generating n shares $(i+1, f_{j,b}(i+1)), i, j \in [n]$ and directly sends the value $f_{j,b}(1)$ to P_0. For $e_k^0 \in B_C[b]$, P_0 updates its shares: $y_k^0 = s_k^{0,0} + f_{1,b}(1) + \cdots + f_{n-1,b}(1)$.

(5) For each bin $B_C[b]$, P_0 and P_j execute β instances of the OLE protocol, where β is the maximum bin size of B_S.
   - P_j acts as the sender with inputs r_j^v and $a_j^{0,v}$, where $v \in [\beta]$ and v is the index of OLE execution. Both inputs are random values.
   - P_0 acts as the receiver with input $e_k^0 \in B_C[b]$, which may be a dummy element. The OLE output for P_0 is $e_k^0 \cdot r_j^v + a_j^{0,v}$.

   After completing the OLE protocol with all P_j, P_0 obtains $z_0^v = (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^0 + (a_1^{0,v} + \cdots + a_{n-1}^{0,v})$.

(6) For each bin $B_S[b]$, P_i and P_j execute β instances of the OLE protocol.
   - P_j acts as the sender with inputs $-r_j^v$ and $a_j^{1,v}$, where $a_j^{0,v} + a_j^{1,v} = f_{j,b}(i+1)$ and v is the index of OLE execution.
   - P_i acts as the receiver with input $e_k^i \in B_S[b]$, which may be a dummy element. The OLE output for P_i is $e_k^i \cdot -r_j^v + a_j^{1,v}$.

   After completing the OLE protocol with all P_j, P_i obtains $z_1^v = -(r_1^v + \cdots + r_{n-1}^v) \cdot e_k^i + (a_1^{1,v} + \cdots + a_{n-1}^{1,v})$.

Conditional Collection and Reconstruction

(7) For the b^{th} bin, P_0 and P_i invoke an OPPRF.
   - P_i is the sender with input $\{(e_k^i, \mu_k^{0,i}) | e_k^i \in B_S[b]\}$, where $\mu_k^{0,i} = \hat{s}_k^{0,i} + z_1^v$.
   - P_0 is the receiver with input $\{e_k^0 | e_k^0 \in B_C[b]\}$ and obtains a corresponding output y_k^i for each $e_k^0 \in S_0$.

(8) For the b^{th} bin, P_0 and P_i invoke an OPPRF.
   - P_i is the sender with input $\{(e_k^i, v) | e_k^i \in B_S[b]\}$, where v is the index of OLE execution about element e_k^i in $B_S[b]$.
   - P_0 is the receiver with input $\{e_k^0 | e_k^0 \in B_C[b]\}$ and obtains a corresponding output v' for each $e_k^0 \in S_0$.

(9) For each e_k^0, P_0 identifies its position b in B_C, retrieves $z_0^{v'}$ from the b^{th} bin and updates the share as $y_k^i = y_k^i + z_0^{v'}$. Then, for each element $e_k^0 \in S_0$, P_0 applies Lagrange interpolation to compute $Recon(y_k^i) = f_k(\cdot)$ over all subsets of t shares among the n values, always including its own share. If any reconstruction yields $f_k(0) = e_k^{0\prime}$, the element e_k^0 is confirmed to be in the intersection. Subsequently, P_0 determines all parties holding this element by verifying whether $f_k(i+1) = y_k^i$ for each i. Finally, P_0 obtains the complete intersection set I.

Fig. 6: ST-OT-MP-PSI protocol.

TABLE I: Analytic comparison of related work with our protocols. n is the number of parties. t is the threshold. Each party holds a set of size m. λ and κ are statistical and computational security parameters, respectively. e is Euler's constant.

Protocol	**Communication**		**Computation**		**Corruption**
	Leader	**Client**	**Leader**	**Client**	**Resilience**
Mahdavi et al. [17]	$O(nmt)$		$O(m(nlog(\frac{m}{t}))^{2t})$		$t-2^*$
ET-OT-MP-PSI	$O(nm\lambda)$	$O(nm\lambda)$	$O(max\{t^2(\frac{e(n-1)}{t-1})^{t-1}), n\kappa\}m)$	$O(max\{\kappa, nt\lambda\}m)$	$t-2$
ST-OT-MP-PSI	$O(n^2m\lambda)$	$O(nm\lambda)$	$O(max\{t^2(\frac{e(n-1)}{t-1})^{t-1}), n^2\lambda, n\kappa\}m)$	$O(max\{\kappa, nt\lambda\}m)$	$n-1$

* In Mahdavi et al.'s protocol, it is required that certain designated roles must not collude.

performs OPPRF with $n-1$ Clients, adding $O(nm\kappa)$. For *reconstruction*, since the Leader's shares are always correct, each of the m elements requires $\binom{n-1}{t-1}$ operations, leading to a total complexity of $O(mt^2\binom{n-1}{t-1})$. Using the approximation $\binom{n-1}{t-1} \leq (e(n-1)/(t-1))^{t-1}$, the complexity is relaxed to $O(mt^2(e(n-1)/(t-1))^{t-1})$, where e is Euler's constant. In the phase of *secret shares update*, each Client generates m_b polynomials of degree $t-1$ and evaluates them at n points, contributing $O(nmt\lambda)$. Additionally, for *conditional collection*, OPPRF between each Client and the Leader adds $O(m\kappa)$. In ST-OT-MP-PSI, during *secret shares update*, the Leader performs n^2m_b additional OLE protocols, requiring $O(n^2m\lambda)$ encryptions and decryptions in our implementation. Clients execute nm_b OLE protocols, involving $O(nm\lambda)$ homomorphic operations.

Comparision to Mahdavi et al.'s. protocol [17]. For communication complexity, our protocols maintain better scalability by avoiding any dependence on the threshold t. In contrast, Mahdavi et al.'s protocol incurs communication costs that grow with t, limiting its practicality in high-threshold settings. For the computational complexity, compared to the computational complexity $O(m(nlog(m/t))^{2t})$ of Mahdavi et al.'s protocol [17], our Traceable OT-MP-PSI protocols demonstrate better efficiency. Specifically, although both our protocols and Mahdavi et al.'s protocol similarly have exponential complexity with t, our protocols achieve a much smaller base and exponent t rather than $2t$, resulting in less computational cost. The enhancement is primarily attributed to the integration of OPPRF with Shamir's secret sharing, which allows each intersection element to be precisely associated with its corresponding shares, instead of exhaustively trying all possible shares from the involved parties in Mahdavi et al.'s protocol when performing reconstruction, thereby significantly reducing the reconstruction time. Lastly, for security, compared to the $t-2$ corruption tolerance in Mahdavi et al.'s protocol, our ET-OT-MP-PSI protocol achieves the same level of resistance while eliminating the assumption that certain special parties do not collude. Meanwhile, our ST-OT-MP-PSI protocol further strengthens security by tolerating collusion among up to $n-1$ semi-honest parties.

V. Performance Evaluation

A. Implementation and Experimental Settings

To evaluate the performance of the proposed Traceable OT-MP-PSI protocols, we implemented both protocols in C++[1]. The implementation relies on the NTL library[2] for large number operations. Communication between the parties is handled using the Boost library, which provides robust tools for message-passing, networking, and parallel processing.

We implemented Shamir's secret sharing using the NTL library, instantiating the finite field modulo the largest 128-bit prime p, which allows us to accommodate the widest possible range of 128-bit elements and aligns with real-world deployment requirements. We adopt the table-based OPPRF construction of Kolesnikov et al. [4], which has favorable communication and computational cost. To satisfy the security assumption that the OPPRF and the Shamir's secret sharing operate over the same finite field $\mathbb{F}_p$, in the implementation, we adjust the table-based OPPRF by replacing XOR operations with modular addition in step 3, and modular subtraction in step 6, respectively. Note that, the programmed and the non-programmed points share the same distribution with such adjustment, and hence the receiver cannot distinguish between the programmed and non-programmed entries.

In the ST-OT-MP-PSI, we utilize the OLE protocol proposed by de Castro et al. [28][3], which is based on Ring Learning with Errors (RLWE). However, the chosen OLE code does not natively support a 128-bit plaintext modulus. To address this limitation, we follow the method described in Section 5.2 of their paper and select p as the product of four smaller 32-bit prime numbers, i.e., $p = \prod_{i=0}^{3} p_i$, thereby extending the original OLE to support a 128-bit plaintext modulus. Since the modulus p is the product of prime numbers, the implementation of this protocol leverages the Chinese Remainder Theorem (CRT). Nevertheless, because we decompose each 128-bit share into four 32-bit CRT residues, our ST-OT-MP-PSI instantiation performs four independent OPPRF evaluations for every share distribution and reconstruction. To this end, we select the four largest 32-bit primes as CRT moduli. Although reducing each residue modulo a 32-bit prime introduces a larger bias than using a single 128-bit modulus, the joint distribution of programmed and non-programmed points remains identical and thus is computationally indistinguishable to the adversary.

Our benchmarking experiments were conducted on a cloud server equipped with an Intel(R) Xeon(R) CPU running at 3.1GHz, featuring 80 vCores and 192GB of RAM, and operating on Ubuntu 22.04. In our experimental setup, each participant operated within a single process, and communication was conducted over a local network without bandwidth or latency constraints. The length of each element is 128 bits. To better evaluate the performance of the proposed protocols, we performed experiments under varying settings of participant numbers and set sizes.

We pick Mahdavi et al.'s protocol [17] as a comparison baseline since both protocols similarly provide traceability in OT-MP-PSI. This comparison highlights that our protocols achieve significantly higher efficiency compared to the existing solution, while maintaining the same functionality. The publicly available source code enables direct implementation[4] and consistent benchmarking under similar conditions. To ensure a fair comparison, we adopted the same elements generation method as that used by Mahdavi et al. Among the two constructions presented in their work, we concentrated on the more efficient variant for benchmarking.

[1]https://github.com/Yank3l/T-OT-MP-PSI
[2]https://libntl.org/
[3]https://github.com/leodec/ole_wahc
[4]https://github.com/cryspuwaterloo/OT-MP-PSI

B. Results Evaluation

Tables II and III present the performance of our proposed Traceable OT-MP-PSI protocols for varying numbers of participants n and the set sizes m. The results indicate a clear linear relationship between runtime of protocols and set size. This observation is consistent with the computational complexity analysis in Section IV-B, where the complexity scales linearly with the set size m. For example, in Table II, with $n = 5$ participants and a threshold of $t = 3$, the runtime increases from 1.73s at $m = 2^{14}$ to 6.23s at $m = 2^{16}$ and 24.76s at $m = 2^{18}$.

The ET-OT-MP-PSI demonstrates strong performance, achieving a runtime of 45.21s for $n = 10$, $t = 5$, and $m = 2^{16}$. Meanwhile, the ST-OT-MP-PSI achieves enhanced security by introducing the OLE, though at the cost of increased computational overhead. Specifically, this protocol requires an additional $O(n^2 m\lambda)$ executions of the OLE protocol, which imposes additional performance overhead. For instance, with $n = 5$, $t = 3$, and $m = 2^{16}$, the protocol completes in approximately 207s. In practice, the choice between the two protocols depends on the specific balance between performance and security requirements. The ET-OT-MP-PSI is ideal for scenarios prioritizing speed, while the ST-OT-MP-PSI is better suited for scenarios where robust security is essential.

TABLE II: The average runtime (in seconds) over 10 trials of the ET-OT-MP-PSI.

m	(n,t)					
	(5,3)	(6,3)	(7,4)	(8,4)	(9,5)	(10,5)
2^{12}	0.68	0.87	1.23	1.50	2.62	3.53
2^{14}	1.73	2.15	3.34	4.34	8.37	11.88
2^{16}	6.23	7.67	12.32	15.71	31.80	45.21
2^{18}	24.76	30.43	48.66	61.50	128.85	182.25

TABLE III: The average runtime (in seconds) over 10 trials of the ST-OT-MP-PSI.

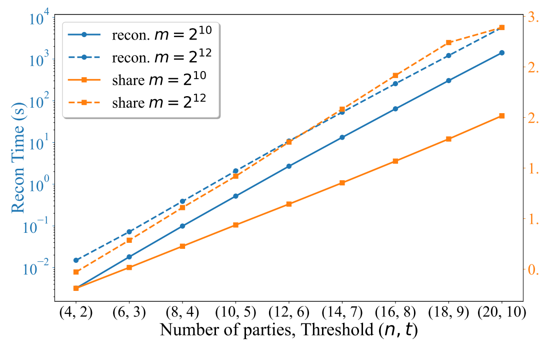
m	(n,t)					
	(5,3)	(6,3)	(7,4)	(8,4)	(9,5)	(10,5)
2^{12}	14.67	20.55	28.11	36.31	47.51	60.04
2^{14}	53.22	76.22	104.71	135.91	181.41	229.09
2^{16}	207.78	298.64	404.89	529.12	716.11	903.28

Fig. 7 illustrates the runtime performance of our protocol under varying numbers of parties n with a threshold setting of $t = n/2$, evaluated for two different set sizes: $m = 2^{10}$ and $m = 2^{12}$. We separately measure the runtime of the two phases of the protocol: the share phase, which includes both the initial Shamir's secret sharing and subsequent share updating, and the reconstruction phase, which performs secret reconstruction. It is worth noting that the left y-axis is presented on a logarithmic scale, while the right y-axis uses a normal (linear) scale to better illustrate the growth trends. When the threshold is fixed at $t = n/2$, we observe that in the ET-OT-MP-PSI, the share phase scales approximately linearly with the number of parties, whereas in the ST-OT-MP-PSI, it exhibits quadratic growth with respect to the number of parties. In contrast, the reconstruction phase shows clear exponential growth with n, which is expected since reconstructing each secret requires iterating over all possible subsets of t shares.

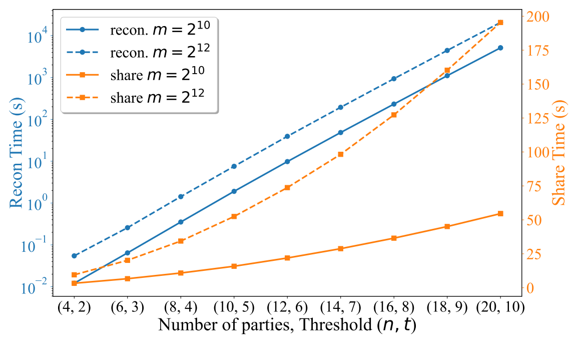
Fig. 8 shows the performance of our protocols and Mahdavi et al.'s protocol [17] across different threshold values t. Due to the excessive runtime of Mahdavi et al.'s protocol, we evaluated it under a smaller setting with $n = 10$ and $m = 2^5$, whereas our protocols were tested up to larger parameters with $n = 20$ and $m = 2^{10}$. Notably, when $t > 7$, Mahdavi et al.'s protocol exceeds the evaluation time limit and is thus not presented in the figure. In the reconstruction phase, the runtime of both our protocols and Mahdavi et al.'s grows exponentially with the threshold t, which aligns with our computational complexity analysis. We observe that Mahdavi et al.'s protocol exhibits a rapid increase in reconstruction runtime, which continues to grow until $t = n$, making it suitable only for small values of t. In contrast, for our protocols, the reconstruction runtime reaches its peak when the number of participants n and set size m are fixed, and the threshold t approaches $(n + 1)/2$. This behavior is expected, as the number of combinations $\binom{n-1}{t-1}$ is maximized near this point. Due to the smaller base and exponent, our protocols experience a much slower rate of growth, and during the growth phase, the runtime is consistently much smaller than that of Mahdavi et al.'s protocol for the same settings.

In the performance comparison, we focused on the more efficient version of the protocol proposed by Mahdavi et al. [17]. To provide a comprehensive evaluation, we tested the protocols in two distinct scenarios: one involving a larger number of participants with smaller sets and the other featuring fewer participants with larger sets. For the first scenario, with more participants and smaller sets, we set the number of participants to 10, the threshold to 5, and the set sizes to $2^4, 2^5, 2^6$, and 2^7. In the second scenario, with fewer participants and larger sets, we set the number of participants to 5, the threshold to 3, and the set sizes to $2^{10}, 2^{12}, 2^{14}$, and 2^{16}. Table IV summarizes the performance of the protocols under these settings.

The results demonstrate that both the ET-OT-MP-PSI and ST-OT-MP-PSI consistently surpass Mahdavi et al.'s protocol in shares generation, reconstruction, and overall runtime across all evaluated scenarios. For instance, in terms of overall runtime, with 10 participants, a threshold of 5, and a set size of 2^7, our protocols are $4312\times$ and $637\times$ faster, respectively, compared to Mahdavi et al.'s protocol. Similarly, with 5 participants, a threshold of 3, and a set size of 2^{14}, our protocols achieve speedups of $15056\times$ and $505\times$, respectively. To gain deeper insights into the performance advantages of our protocols, we analyze the share and reconstruction phases individually. In the share phase, ET-OT-MP-PSI utilizes a combination of Shamir's secret sharing and OPPRF. These techniques are predominantly based on efficient symmetric-

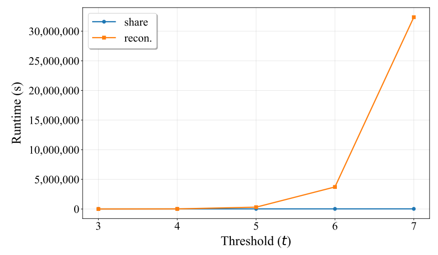


(a) ET-OT-MP-PSI

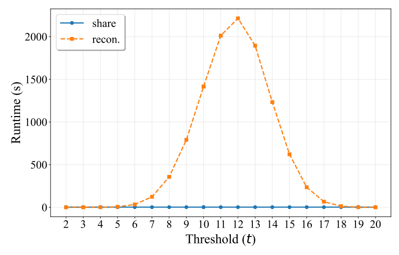


(b) ST-OT-MP-PSI

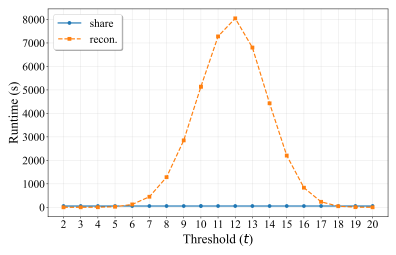
Fig. 7: The average runtime (in seconds) of our protocols with varying numbers of participants.



(a) Mahdavi et al.'s protocol



(b) ET-OT-MP-PSI



(c) ST-OT-MP-PSI

Fig. 8: Comparison of the runtime (in seconds) between Mahdavi et al.'s protocol and our protocols for varying threshold t.

key operations, which are computationally lightweight. In contrast, the share stage in Mahdavi et al.'s protocol relies on Paillier homomorphic encryption, which is significantly more computationally expensive. The ST-OT-MP-PSI further introduces OLE to enable secure share updates, which incurs a moderate computational overhead but remains more efficient than the homomorphic encryption used in Mahdavi et al.'s scheme. In the reconstruction phase, our protocols also demonstrate superior efficiency. For instance, with 10 participants, a threshold of 5, and a set size of 2^7, our protocols are 45215× and 13761× faster, respectively, compared to Mahdavi et al.'s protocol. By leveraging OPPRF and Shamir's secret sharing, we reduce the reconstruction complexity and significantly lowering the computational overhead required for reconstruction operations.

These results clearly show the superior efficiency of our proposed protocols compared to Mahdavi et al.'s protocol. The consistent performance improvements across various settings highlight the practicality of both the ET-OT-MP-PSI and ST-OT-MP-PSI protocols. By offering a flexible trade-off between runtime efficiency and security, our protocols are positioned to address a variety of real-world needs.

VI. Related Work

A. Multi-Party PSI and Variants

With the wide range of applications for MP-PSI, the past decade has witnessed the development of numerous protocols aimed at tackling challenges in both efficiency and security with diverse cryptographic techniques. Freedman et al. [29] proposed the first MP-PSI in the semi-honest model, relying on oblivious polynomial evaluation (OPE) with homomorphic encryption. This approach was later adopted by other works, including [30], [31], [32], [33]. Miyaji and Nishida [19] combined exponential ElGamal encryption with Bloom filters to design an MP-PSI, which relies on a trusted third party and is applied to medical data analysis [34]. Kolesnikov et al. [4] proposed three constructions for instantiating oblivious programmable pseudorandom function (OPPRF) using oblivious transfer. Building on this, the authors combined zero-value secret sharing to develop the first efficient MP-PSI. Inbar et al. [5] extended the two-party PSI by Dong et al. [35] to a multi-party setting by utilizing the mergeability of garbled Bloom filters. The first practically efficient MP-PSI with malicious security, introduced by Ben Efraim et al. [36], skillfully integrates techniques from semi-honest MP-PSI [5] and malicious two-party PSI [37], leveraging oblivious transfer and garbled Bloom filters. Chandran et al. [9] proposed a modification to Kolesnikov et al.'s protocol [4], replacing the construction of secret sharing that XOR to zero with Shamir's secret sharing scheme, resulting in a more efficient MP-PSI. In contrast to [4] and [9], which rely on the OPPRF, Wu et al. [20] adopt the more efficient oblivious PRF (OPRF) and a data structure called the oblivious key-value store (OKVS), leading to the development of two MP-PSI: O-Ring and K-Star, designed to address distinct application needs.

Over time, MP-PSI has evolved into a broader family of protocols tailored to meet diverse application-specific needs. Notable variants of MP-PSI include multi-party private set

TABLE IV: Comparison of the overall runtime (in seconds) between Mahdavi et al.'s protocol and our proposed protocols across various settings.

(n,t)		(10,5)				(5,3)			
m		2^4	2^5	2^6	2^7	2^{10}	2^{12}	2^{14}	2^{16}
Mahdavi et al. [17]	share	83.24	167.05	335.73	672.90	1653.78	6585.70	26395.50	-*
	recon.	79.34	308.37	993.63	3165.06	13.31	62.99	404.94	-
	total	162.58	475.42	1329.36	3837.96	1667.09	6648.69	26800.44	-
ET-OT-MP-PSI	share**	0.77	0.77	0.79	0.79	0.42	0.65	1.56	5.42
	recon.	0.01	0.02	0.03	0.07	0.01	0.05	0.22	0.88
	total	0.78	0.79	0.82	0.86	0.43	0.70	1.78	6.29
ST-OT-MP-PSI	share**	4.23	4.60	4.99	5.79	4.84	14.38	52.26	205.04
	recon.	0.03	0.05	0.11	0.23	0.05	0.19	0.77	3.06
	total	4.26	4.66	5.10	6.02	4.89	14.57	53.02	208.11

* Cells with "-" denote the task could not be completed within the testing time.
** "share" includes both secret sharing and shares update phase.

intersection cardinality (MP-PSI-CA) [38], [39], [40], [41], which calculates the size of the intersection without revealing the intersecting elements, and multi-party delegated PSI [42], which allows parties to outsource the storage of their datasets to a cloud computing service. Other extensions, such as MP-PSI-CA-sum [43], differ from MP-PSI-CA in that MP-PSI-CA-sum additionally outputs the sum of the associated integer values of all the data belonging to the intersection, providing richer insights beyond just the cardinality.

B. Multi-Party PSI with Threshold

As an extension of MP-PSI, variants with threshold such as T-MP-PSI and OT-MP-PSI have attracted the attention of researchers. Kissner and Song [7] presented the first T-MP-PSI and OT-MP-PSI. They leveraged Paillier encryption to develop algorithms for encrypted polynomial operations. Miyaji and Nishida [19] introduced a T-MP-PSI called d-and-over MPSI, combining Bloom filters and exponential ElGamal encryption. However, the protocol relies on the assumption of a trusted third party. Mahdavi et al. [17] proposed a new primitive called oblivious pseudo-random secret sharing (OPR-SS), which leverages oblivious pseudo-random function (OPRF) and Shamir's secret sharing. Building on this primitive, a new OT-MP-PSI with traceability was developed. To improve efficiency, the Paillier cryptosystem was introduced to reduce reconstruction time. Nevertheless, the optimized OT-MP-PSI remains impractical for real-world use. Notably, Mahdavi et al.'s OT-MP-PSI not only computes the intersection but also reveals the holders of the intersecting elements. Bay et al. [8] presented a novel T-MP-PSI that utilizes Bloom filters and threshold Paillier encryption. The protocol verifies whether an element is held by at least t participants through two rounds of multi-party secure comparison protocol (SCP). Chandran et al. [9] introduced a T-MP-PSI called Quorum PSI. A major limitation of their protocol is its dependence on the assumption that the majority of parties are honest. Ma et al. [44] presented a novel OT-MP-PSI by introducing the dual cloud framework. In this design, the clients only need to pre-process the data and delegate the subsequent computation to cloud servers, which substantially reduces both the computational and communication overhead on the clients. Yang et al. [45] proposed the first unbalanced T-MP-PSI based on fully homomorphic encryption. Their construction achieves logarithmic communication complexity in the semi-honest setting, thereby offering a significant improvement in efficiency compared with previous work.

VII. Conclusion

Most MP-PSI protocols with threshold, being fully anonymous, are often unsuitable for regulatory scenarios. Moreover, the existing related scheme with traceability exhibits limitations in terms of both security and performance. This paper introduces two novel Traceable Over-Threshold Multi-Party Private Set Intersection (T-OT-MP-PSI) protocols to address more flexible privacy-preserving set intersection challenges. The first protocol Efficient T-OT-MP-PSI leverages OPPRF and Shamir's secret sharing to achieve high efficiency in the semi-honest model, ensuring resilience against up to $t-2$ colluding participants. The second protocol Security-enhanced T-OT-MP-PSI enhances security by introducing the oblivious linear evaluation protocol, improving its ability to resist collusion by up to $n-1$ participants. Experimental results demonstrate the practicality and strong performance of both protocols, showing significant improvement over the existing solution. For instance, with 5 participants, a threshold value of 3, and set sizes of 2^{14}, our protocols are 15056$\times$ (ET-OT-MP-PSI) and 505$\times$ (ST-OT-MP-PSI) faster than the work of Mahdavi et al., respectively.

Acknowledgment

We thank the anonymous reviewers for their thoughtful comments. This work was supported in part by the National Natural Science Foundation of China Youth Project (No.62202102), Scientific and Technological Project of Fujian Province of China (No.2024J08162), the CCF-NSFOCUS 'Kunpeng' Research Fund (No.CCF-NSFOCUS2024004), National Key RD Plan of China (2020YFB1005803) and Key RD Plan of Shandong Province (2020CXGC010115).

References

[1] D. T. Nguyen and N. Trieu, "Mpccache: privacy-preserving multi-party cooperative cache sharing at the edge," in *International Conference on Financial Cryptography and Data Security*. Springer, 2022, pp. 80–99.

[2] L. Lu and N. Ding, "Multi-party private set intersection in vertical federated learning," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2020, pp. 707–714.

[3] A. R. Elkordy, Y. H. Ezzeldin, and S. Avestimehr, "Federated k-private set intersection," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 436–445.

[4] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu, "Practical multi-party private set intersection from symmetric-key techniques," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1257–1272.

[5] R. Inbar, E. Omri, and B. Pinkas, "Efficient scalable multiparty private set-intersection via garbled bloom filters," in *International conference on security and cryptography for networks*. Springer, 2018, pp. 235–252.

[6] S. Ghosh and T. Nilges, "An algebraic approach to maliciously secure private set intersection," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2019, pp. 154–185.

[7] L. Kissner and D. Song, *Private and threshold set-intersection*. School of Computer Science, Carnegie Mellon University, 2004.

[8] A. Bay, Z. Erkin, J.-H. Hoepman, S. Samardjiska, and J. Vos, "Practical multi-party private set intersection protocols," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1–15, 2021.

[9] N. Chandran, N. Dasgupta, D. Gupta, S. L. B. Obbattu, S. Sekar, and A. Shah, "Efficient linear multiparty psi and extensions to circuit/quorum psi," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1182–1204.

[10] A. Valdes and K. Skinner, "Probabilistic alert correlation," in *Recent Advances in Intrusion Detection: 4th International Symposium, RAID 2001 Davis, CA, USA, October 10–12, 2001 Proceedings 4*. Springer, 2001, pp. 54–68.

[11] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer networks*, vol. 34, no. 4, pp. 579–595, 2000.

[12] M. Li, Y. Shen, G. Ye, J. He, X. Zheng, Z. Zhang, L. Zhu, and M. Conti, "Anonymous, secure, traceable, and efficient decentralized digital forensics," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 5, pp. 1874–1888, 2023.

[13] S. R. Selamat, R. Yusof, S. Sahib, N. H. Hassan, M. F. Abdollah, and Z. Z. Abidin, "Traceability in digital forensic investigation process," in *2011 IEEE Conference on Open Systems*. IEEE, 2011, pp. 101–106.

[14] S. R. Selamat, S. Sahib, N. Hafeizah, R. Yusof, and M. F. Abdollah, "A forensic traceability index in digital forensic investigation," *Journal of Information Security*, vol. 4, no. 1, pp. 19–32, 2013.

[15] J. Siddhi, "Anti-money laundering market," Kings Research, Market Research Report, September 2024, report ID: KR322, Accessed on 2025-04-05. [Online]. Available: https://www.kingsresearch.com/anti-money-laundering-market-report

[16] P.-L. Chatain, *Preventing money laundering and terrorist financing: a practical guide for bank supervisors*. World Bank Publications, 2009.

[17] R. A. Mahdavi, T. Humphries, B. Kacsmar, S. Krastnikov, N. Lukas, J. A. Premkumar, M. Shafieinejad, S. Oya, F. Kerschbaum, and E.-O. Blass, "Practical over-threshold multi-party private set intersection," in *Proceedings of the 36th Annual Computer Security Applications Conference*, 2020, pp. 772–783.

[18] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Advances in Cryptology—CRYPT0'95: 15th Annual International Cryptology Conference Santa Barbara, California, USA, August 27–31, 1995 Proceedings 15*. Springer, 1995, pp. 339–352.

[19] A. Miyaji and S. Nishida, "A scalable multiparty private set intersection," in *International conference on network and system security*. Springer, 2015, pp. 376–385.

[20] M. Wu, T. H. Yuen, and K. Y. Chan, "O-Ring and K-Star: Efficient multi-party private set intersection," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 6489–6506.

[21] Y. Lindell, "How to simulate it–a tutorial on the simulation proof technique," *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, pp. 277–346, 2017.

[22] Y. Gao, Y. Luo, L. Wang, X. Liu, L. Qi, W. Wang, and M. Zhou, "Efficient scalable multi-party private set intersection (-variants) from bicentric zero-sharing," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 4137–4151.

[23] O. Nevo, N. Trieu, and A. Yanai, "Simple, fast malicious multiparty private set intersection," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1151–1165.

[24] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[25] B. Pinkas, T. Schneider, and M. Zohner, "Scalable private set intersection based on ot extension," *ACM Transactions on Privacy and Security (TOPS)*, vol. 21, no. 2, pp. 1–35, 2018.

[26] V. Kolesnikov, R. Kumaresan, M. Rosulek, and N. Trieu, "Efficient batched oblivious prf with applications to private set intersection," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 818–829.

[27] P. Rindal and P. Schoppmann, "Vole-psi: fast oprf and circuit-psi from vector-ole," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2021, pp. 901–930.

[28] L. de Castro, C. Juvekar, and V. Vaikuntanathan, "Fast vector oblivious linear evaluation from ring learning with errors," in *Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, 2021, pp. 29–41.

[29] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *International conference on the theory and applications of cryptographic techniques*. Springer, 2004, pp. 1–19.

[30] J. H. Cheon, S. Jarecki, and J. H. Seo, "Multi-party privacy-preserving set intersection with quasi-linear complexity," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 95, no. 8, pp. 1366–1378, 2012.

[31] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung, "Secure efficient multiparty computing of multivariate polynomials and applications," in *Applied Cryptography and Network Security: 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings 9*. Springer, 2011, pp. 130–146.

[32] C. Hazay and M. Venkitasubramaniam, "Scalable multi-party private set-intersection," in *IACR international workshop on public key cryptography*. Springer, 2017, pp. 175–203.

[33] Y. Sang and H. Shen, "Privacy preserving set intersection protocol secure against malicious behaviors," in *Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT 2007)*. IEEE, 2007, pp. 461–468.

[34] A. Miyaji, K. Nakasho, and S. Nishida, "Privacy-preserving integration of medical data: a practical multiparty private set intersection," *Journal of medical systems*, vol. 41, pp. 1–10, 2017.

[35] C. Dong, L. Chen, and Z. Wen, "When private set intersection meets big data: an efficient and scalable protocol," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 789–800.

[36] A. Ben-Efraim, O. Nissenbaum, E. Omri, and A. Paskin-Cherniavsky, "Psimple: Practical multiparty maliciously-secure private set intersection," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, 2022, pp. 1098–1112.

[37] P. Rindal and M. Rosulek, "Improved private set intersection against malicious adversaries," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2017, pp. 235–259.

[38] S. K. Debnath, P. Stănică, N. Kundu, and T. Choudhury, "Secure and efficient multiparty private set intersection cardinality." *Advances in Mathematics of Communications*, vol. 15, no. 2, 2021.

[39] N. Trieu, A. Yanai, and J. Gao, "Multiparty private set intersection cardinality and its applications." *IACR Cryptol. ePrint Arch.*, vol. 2022, p. 735, 2022.

[40] B. Liu, M. Zhang, and R. Shi, "Quantum secure multi-party private set intersection cardinality," *International Journal of Theoretical Physics*, vol. 59, pp. 1992–2007, 2020.

[41] R.-H. Shi and Y.-F. Li, "Quantum protocol for secure multiparty logical and with application to multiparty private set intersection cardinality,"

IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 12, pp. 5206–5218, 2022.

[42] A. Abadi, C. Dong, S. J. Murdoch, and S. Terzis, "Multi-party updatable delegated private set intersection," in *International Conference on Financial Cryptography and Data Security*. Springer, 2022, pp. 100–119.

[43] I.-S. U. Arbitrary, "Practical multi-party private set intersection cardinality and intersection-sum under arbitrary collusion," in *Information Security and Cryptology: 18th International Conference, Inscrypt 2022, Beijing, China, December 11–13, 2022, Revised Selected Papers*, vol. 13837. Springer, 2023, p. 169.

[44] L. Ma, H. Wang, Z. Niu, Z. Li, L. Wu, X. Wei, and Y. Su, "Over-threshold multi-party private set operation protocols for lightweight clients," *Computer Standards & Interfaces*, vol. 88, p. 103781, 2024.

[45] X. Yang, L. Cai, Y. Wang, K. Yin, L. Sun, and J. Hu, "Efficient unbalanced quorum psi from homomorphic encryption," in *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, 2024, pp. 1003–1016.

Appendix

A. Correctness and Security Proofs of ET-OT-MP-PSI

Theorem 1. *The ET-OT-MP-PSI realizes the functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$ and is secure against collusion among up to $t-2$ parties in the semi-honest model, given the statistical security parameter λ and the computational security parameter κ.*

1) Correctness:

Proof. To analyze the correctness of the protocol, we consider the following two cases for each element e^0_k in the set of P_0:

- **Case 1:** e^0_k is an element of the intersection, i.e., it is held by at least t parties, including P_0.
- **Case 2:** e^0_k is not an element of the intersection, meaning that fewer than t parties hold this element.

Case 1: e^0_k is an element of the intersection.

1) **Conditional secret sharing.** P_0 treats the element e^0_k as a secret and applies Shamir's secret sharing scheme to generate n shares $s^{0,0}_k, \ldots, s^{0,n-1}_k$. These shares are then conditionally delivered to the other parties via OPPRF. Since e^0_k is an element of the intersection, at least t parties will receive the correct shares, i.e., the OPPRF output satisfies $\hat{s}^{0,i}_k = s^{0,i}_k$ for those parties.
2) **Secret shares update.** For the b^{th} bin, every party P_i, where $i \in [1, n-1]$, generates a polynomial $f_{i,b}(\cdot)$ and sends the evaluation $f_{i,b}(j+1)$ to each other party P_j, where $j \in [n]$. Each party P_j then adds up the $n-1$ received values to compute $\delta_b = f_{1,b}(j+1) + \cdots + f_{n-1,b}(j+1)$. The secret share is then updated as $\mu^{0,i}_k = \hat{s}^{0,i}_k + \delta_b$. Since at least t parties have received correct shares, at least t of the updated shares are also correct.
3) **Conditional collection and reconstruction.** Each party P_i, where $i \in [1, n-1]$, invokes OPPRF protocol with P_0 to conditionally deliver the updated share $\mu^{0,i}_k$. Since the element e^0_k is held by at least t parties, P_0 obtains at least t correct shares from the OPPRF, denoted as y^i_k, where $y^i_k = \mu^{0,i}_k$. Then, P_0 attempts to reconstruct the original secret by selecting t values from the n received outputs. When the selected t shares are all correct, the original secret can be reconstructed, i.e., $Recon(y^i_k) = e^0_k$. Therefore, e^0_k can be identified as an element in the intersection, and the parties holding correct shares are the holders of this intersection element.

Case 2:e^0_k is not an element of the intersection.

1) **Conditional secret sharing.** The procedure is the same as in Case 1. However, since the element e^0_k is not an element of the intersection—i.e., it is held by fewer than t parties—the number of correct secret shares obtained by P_0 through the OPPRF protocol is less than t.
2) **Secret shares update.** The execution process is identical to that of Case 1. Since fewer than t parties obtained correct shares in phase 1, the number of correct updated shares after the shares update phase remains less than t.
3) **Conditional collection and reconstruction.** The number of correct shares that P_0 obtains from the OPPRF outputs is less than t. When P_0 attempts to reconstruct the secret, it fails to recover the original secret with overwhelming probability under the given security parameters λ and κ. As a result, P_0 determines that the element e^0_k is not part of the intersection.

□

2) Security:

Proof. As discussed in Section III-D, the ET-OT-MP-PSI is insecure in the presence of collusion among $t-1$ corrupted parties. Therefore, according to Definition 1, we prove that it is secure in the presence of collusion among up to $t-2$ semi-honest adversaries. The following two distinct collusion scenarios should be considered. Here, we assume that Shamir's secret sharing and the OPPRF operate over the same field $\mathbb{F}_p$. Let $\mathbb{C}$ and $\mathbb{H}$ be a coalition of corrupt and honest participants respectively. And let X and Y be the inputs and outputs of the the coalition $\mathbb{C}$.

- **Case 1:** Party P_0 is honest, and $t-2$ other parties are colluding, i.e. $\mathbb{C} \subseteq \{P_1, \ldots, P_{n-1}\}, |\mathbb{C}| = t-2$.
- **Case 2:** Party P_0 is corrupted and colludes with $t-3$ other parties, i.e. $\mathbb{C} = \{P_0\} \cup \mathbb{C}_1$, where $\mathbb{C}_1 \subseteq \{P_1, \ldots, P_{n-1}\}, |\mathbb{C}_1| = t-3$.

Case 1: Party P_0 is honest.

In this case, the simulator Sim is given the inputs X and outputs $Y = \perp$ of the corrupted parties $\mathbb{C}$, and runs as follows:

1) **Conditional secret sharing.** Sim samples $t-2$ random values $\overline{\hat{s}^{0,i}_k} \leftarrow \mathbb{F}_p$, as the programmed outputs of the OPPRF.
2) **Secret shares update.** For the b^{th} bin, Sim randomly generates polynomial $\overline{f_{j,b}(\cdot)}$ with a constant term of 0 and a degree of at most $t-1$, where $j \in \mathbb{H}$. Then, Sim computes $\overline{\delta_b} = \sum_{j\in\mathbb{C}} f_{j,b}(i+1) + \sum_{j\in\mathbb{H}} \overline{f_{j,b}}(i+1)$ and $\overline{y^k_i} = \overline{\hat{s}^{0,i}_k} + \overline{\delta_b}$.
3) **Conditional collection and reconstruction.** At this stage, the corrupted parties receive no input. Therefore, the simulator Sim generates their corresponding views by faithfully following the protocol steps.

Given the statistical security parameter λ and the computational security parameter κ, both the OPPRF and our secret sharing scheme operate over the same finite field $\mathbb{Z}_p$. Now we argue that the views generated by Sim are computationally indistinguishable from those in the real execution.

- In the real world, according to the obliviousness of OPPRF, each $\hat{s}_k^{0,i}$ is indistinguishable from $\overline{\hat{s}_k^{0,i}}$ for OPPRF receiver, i.e. $\hat{s}_k^{0,i} \overset{c}{\equiv} \overline{\hat{s}_k^{0,i}}$. So the simulated views are computationally indistinguishable from the views in the real execution.
- During the real execution, the honest parties generate polynomials $f_{j,b}(\cdot)$ randomly, and each evaluation $f_{j,b}(i+1) \leftarrow \mathbb{F}_p$. Similarly, the simulator generates values $\overline{f_{j,b}(i+1)} \leftarrow \mathbb{F}_p$, which means $f_{j,b}(i+1) \overset{c}{\equiv} \overline{f_{j,b}(i+1)}$ and hence $\delta_b \overset{c}{\equiv} \overline{\delta_b}$.

Therefore, we have $\{\mathsf{Sim}(X, Y, \mathbb{C})\} \overset{c}{\equiv} \{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}$.

Case 2: Party P_0 is corrupted.

In this case, colluding parties learn the final outputs I of the protocol. Sim is given the inputs X and outputs $Y = I$ of the corrupted parties $\mathbb{C}$, and runs as follows:

1) **Conditional secret sharing.** Sim selects $t-2$ random values $\overline{\hat{s}_k^{0,i}} \leftarrow \mathbb{F}_p$, as the outputs of the OPPRF.
2) **Secret shares update.** This process is identical to Case 1: the simulator Sim generates the simulated polynomial $\overline{f_{j,b}(\cdot)}$ and computes $\overline{\delta_b}$.
3) **Conditional collection and reconstruction.** If $e_k^0 \notin I$, or $e_k^0 \in I$ but $e_k^0 \notin S_i$, Sim selects random values as the OPPRF outputs $\overline{y_k^i}$ received by P_0 from the honest party P_i. If $e_k^0 \in I$ and $e_k^0 \in S_i$, the simulator Sim computes the correct outputs of the OPPRF protocol:
$$\overline{y_k^i} = s_k^{0,i} + \sum_{P_j \in H} \overline{f_{j,b}(i+1)} + \sum_{P_j \in C} f_{j,b}(i+1).$$

Given the statistical security parameter λ and the computational security parameter κ, both the OPPRF and our secret sharing scheme operate over the same finite field $\mathbb{Z}_p$. Now we argue that the views generated by Sim are computationally indistinguishable from those in the real execution.

- According to the obliviousness of OPPRF, the receiver cannot distinguish between the simulated value $\overline{\hat{s}_k^{0,i}}$ and the actual value $\hat{s}_k^{0,i}$ generated during the real execution of the protocol. Therefore, we have $\hat{s}_k^{0,i} \overset{c}{\equiv} \overline{\hat{s}_k^{0,i}}$.
- Similar to Case 1, Sim samples random polynomial $\overline{f_{j,b}(\cdot)}$ from the same distribution over $\mathbb{F}_p$ as the honest parties do in the real execution. Therefore, we have $f_{j,b}(i+1) \overset{c}{\equiv} \overline{f_{j,b}(i+1)}$ and $\delta_b \overset{c}{\equiv} \overline{\delta_b}$.
- Sim is given the final output I of the protocol. From this, Sim can determine the correct OPPRF outputs for each element in the intersection. If $e_k^0 \notin I$, or $e_k^0 \in I$ but $e_k^0 \notin S_i$, then by the obliviousness of OPPRF, the random value $\overline{y_k^i}$ chosen by Sim satisfies $\overline{y_k^i} \overset{c}{\equiv} \mathsf{output}_{\mathsf{OPPRF}}$. If $e_k^0 \in I$ and $e_k^0 \in S_i$, then Sim can compute the correct OPPRF output. Therefore, the simulated value $\overline{y_k^i}$ satisfies $y_k^i \overset{c}{\equiv} \overline{y_k^i}$.

Therefore, we have $\{\mathsf{Sim}(X, Y, \mathbb{C})\} \overset{c}{\equiv} \{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}$. $\square$

B. Correctness and Security Proofs of ST-OT-MP-PSI

Theorem 2. *The ST-OT-MP-PSI realizes the functionality $\mathcal{F}^{n,m,t}_{\text{T-OT-MP-PSI}}$ and is secure against collusion among up to $n-1$ parties in the semi-honest model, given the statistical security parameter λ and the computational security parameter κ.*

1) Correctness:

Proof. Similarly, for each element e_k^0 in the set of P_0, we analyze the following two distinct cases:

- **Case 1:** e_k^0 is an element of the intersection, i.e., it is held by at least t parties, including P_0.
- **Case 2:** e_k^0 is not an element of the intersection, meaning that fewer than t parties hold this element.

Case 1: e_k^0 is an element of the intersection.

1) **Conditional secret sharing.** The protocol proceeds in essentially the same way as in ET-OT-MP-PSI, except that the shared secret is a random value denoted by $e_k^{0'}$. In this phase, at least t parties obtain correct secret shares.
2) **Secret shares update.** Each party P_j, where $j \in [1, n-1]$, generates a polynomial $f_{j,b}(\cdot)$ and directly sends $f_{j,b}(1)$ to P_0. P_0 then uses this value to correctly update its share. Subsequently, P_0 invokes OLE protocol with P_j, in which P_j obtains the following OLE output: $z_0^v = (r_1^v + \cdots + r_{n-1}^v) \cdot e_k^0 + (a_1^{0,v} + \cdots + a_{n-1}^{0,v})$. Similarly, each party P_i, where $i \in [1, n-1]$, invokes OLE protocol with P_j and obtains the output: $z_1^v = -(r_1^v + \cdots + r_{n-1}^v) \cdot e_k^i + (a_1^{1,v} + \cdots + a_{n-1}^{1,v})$. P_i then uses z_1^v to update its share as follows: $\mu_k^{0,i} = \hat{s}_k^{0,i} + z_1^v$.
3) **Conditional collection and reconstruction.** Each party P_i, where $i \in [1, n-1]$, invokes OPPRF protocol with P_0 to conditionally deliver the updated share $\mu_k^{0,i}$. Since the element e_k^0 is held by at least t parties, P_0 receives at least t correct updated shares, i.e., $y_k^i = \mu_k^{0,i}$ for those parties. Then, P_0 adds each received value y_k^i to the corresponding value z_1^0 to obtain the final updated share: $y_k^i = y_k^i + z_1^0$. Since at least t out of the n updated shares obtained by P_0 are correct, P_0 can successfully reconstruct the original secret by using these correct shares, i.e., $Recon(y_k^i) = e_k^{0'}$. Therefore, P_0 can correctly identify that e_k^0 is in the intersection, and the parties corresponding to the correct shares are the holders of this intersection element.

Case 2:e_k^0 is not an element of the intersection.

1) **Conditional secret sharing.** The process is identical to that of Case 1. However, since fewer than t parties hold the element e_k^0, fewer than t correct secret shares are obtained after the OPPRF.
2) **Secret shares update.** The procedure is exactly the same as in Case 1.
3) **Conditional collection and reconstruction.** After the OPPRF execution, the number of correct updated shares obtained by P_0 is fewer than t. When P_0 attempts to reconstruct the secret using n received values, it fails to recover the original secret with overwhelming probability

under the given security parameters λ and κ. Therefore, P_0 concludes that e_k^0 is not an element of the intersection. $\square$

2) *Security:*

Proof. To prove the security, we consider the following two cases. Both Shamir's secret sharing and the OPPRF are assumed to operate over the same finite field $\mathbb{F}_p$. Let $\mathbb{C}$ and $\mathbb{H}$ denote the colluding parties and honest parties, respectively.

- **Case 1:** Party P_0 is honest, and other parties are colluding.
- **Case 2:** Party P_i is honest, where $i \in [1, n-1]$, while the remaining parties, including P_0, are corrupted.

Case 1: Party P_0 is honest.

In this case, the simulator Sim is given the inputs X and outputs $Y = \perp$ of the corrupted parties $\mathbb{C}$, and runs as follows:

1) **Conditional secret sharing.** Sim samples $n-1$ random values $\overline{\hat{s}_k^{0,i}} \leftarrow \mathbb{F}_p$, as the simulated outputs of the OPPRF.
2) **Secret shares update.** Sim selects random values $\overline{e_k^0} \leftarrow \mathbb{F}_p$ as the simulated inputs of P_0 and the corrupted parties for the OLE protocol.
3) **Conditional collection and reconstruction.** As the corrupted parties obtain no inputs during this phase, Sim can simulate their views according to the protocol, resulting in views that are computationally indistinguishable from the real ones.

Given the statistical security parameter λ and the computational security parameter κ, both the OPPRF and our secret sharing scheme operate over the same finite field $\mathbb{Z}_p$. Now we argue that the views generated by Sim are computationally indistinguishable from those in the real execution.

- First, since both the OPPRF and Shamir's secret sharing operate over the same field $\mathbb{F}_p$, and due to the obliviousness of OPPRF, the value $\overline{\hat{s}_k^{0,i}}$ chosen by Sim is computationally indistinguishable from the real value $\hat{s}_k^{0,i}$ from the receiver's perspective. That is, $\hat{s}_k^{0,i} \overset{c}{\equiv} \overline{\hat{s}_k^{0,i}}$.
- In the real world, P_0's inputs to the OLE are the set elements e_k^0, sampled uniformly from $\mathbb{F}_p$. Therefore, $e_k^0 \overset{c}{\equiv} \overline{e_k^0}$.

Therefore, we have $\{\mathsf{Sim}(X, Y, \mathbb{C})\} \overset{c}{\equiv} \{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}$.

Case 2: Party P_0 is corrupted.

In this case, colluding parties learn the final outputs I of the protocol. Sim is given the inputs X and outputs $Y = I$ of the corrupted parties $\mathbb{C}$, and runs as follows:

1) **Conditional secret sharing.** Sim samples $n-1$ random values $\overline{\hat{s}_k^{0,i}} \leftarrow \mathbb{F}_p$, as the outputs of the OPPRF.
2) **Secret shares update.** For the bth bin, Sim constructs a random polynomial $\overline{f_{i,b}(\cdot)}$ with a constant term of 0 and a degree of at most $t-1$, and sets $\overline{f_{i,b}(1)}$ as the value used by the corrupted party P_0 to update its share. If the element held by the corrupted party P_0 is e_0 and the element held by another corrupted party P_j is e_1, Sim generates the simulated OLE outputs $\overline{c_0} = \overline{r_i} \cdot e_0 + \overline{a_i^0}$ for P_0 and $\overline{c_1} = -\overline{r_i} \cdot e_1 + \overline{a_i^1}$ for P_j, where $\overline{a_i^0} + \overline{a_i^1} = \overline{f_i(j+1)}$ and $\overline{r_i}$ are random values.
3) **Conditional collection and reconstruction.** If $e_k^0 \notin I$, or $e_k^0 \in I$ but $e_k^0 \notin S_i$, Sim generates a random value to simulate the OPPRF output from the honest party P_i. If $e_k^0 \in I$ and $e_k^0 \in S_i$, the simulator deduces the correct OPPRF output:
$$\overline{y_k^i} = s_k^{0,i} - \Big(\sum_{j=1, j \neq i}^{n-1} r_j + \overline{r_i}\Big) \cdot e_k^0 + \Big(\sum_{j=1, j \neq i}^{n-1} a_j^1 + \overline{a_i^1}\Big).$$
Similarly, if $e_k^0 \notin I$, or $e_k^0 \in I$ but $e_k^0 \notin S_i$, then Sim selects a random value $\overline{v'}$ as the OLE index derived from the OPPRF. If $e_k^0 \in I$ and $e_k^0 \in S_i$, then Sim can compute the correct OLE index.

Given the statistical security parameter λ and the computational security parameter κ, both the OPPRF and our secret sharing scheme operate over the same finite field $\mathbb{Z}_p$. Now we argue that the views generated by Sim are computationally indistinguishable from those in the real execution.

- According to the obliviousness of OPPRF, each $\hat{s}_k^{0,i}$ is indistinguishable from $\overline{\hat{s}_k^{0,i}}$ to the receiver, i.e. $\hat{s}_k^{0,i} \overset{c}{\equiv} \overline{\hat{s}_k^{0,i}}$.
- In the real world, polynomials $f_{j,b}(\cdot)$ are generated randomly, and each evaluation $f_{j,b}(i+1) \leftarrow \mathbb{F}_p$. Similarly, Sim generates values $\overline{f_{j,b}(i+1)} \leftarrow \mathbb{F}_p$, which means $f_{j,b}(i+1) \overset{c}{\equiv} \overline{f_{j,b}(i+1)}$. According to the obliviousness of OLE, the receiver cannot computationally distinguish the OLE output from a uniformly random value over the same field $\mathbb{F}_p$. Therefore, the simulated outputs generated by Sim are computationally indistinguishable from those in the real execution, i.e., $c_0 \overset{c}{\equiv} \overline{c_0}$ and $c_1 \overset{c}{\equiv} \overline{c_1}$.
- Sim is given the final output I of the protocol. From this, Sim can determine the correct OPPRF outputs for each element in the intersection. If $e_k^0 \notin I$, or $e_k^0 \in I$ but $e_k^0 \notin S_i$, then by the obliviousness of OPPRF, we have $y_k^i \overset{c}{\equiv} \overline{y_k^i}$ and $v' \overset{c}{\equiv} \overline{v'}$. If $e_k^0 \in I$ and $e_k^0 \in S_i$, then Sim can compute the correct OPPRF output. Consequently, $y_k^i \overset{c}{\equiv} \overline{y_k^i}$ and $v' \overset{c}{\equiv} \overline{v'}$.

Therefore, we have $\{\mathsf{Sim}(X, Y, \mathbb{C})\} \overset{c}{\equiv} \{\mathsf{view}_{\mathbb{C}}^{\pi}(X, Y)\}$. $\square$