Analysis, modify and apply in IIOT form light-weight PSI in CM20

Zhuang Shan(单壮)[1], Leyou Zhang(张乐友)[1,*], Qing Wu(吴青)[2], Qiqi Lai(来齐齐)[3]

June 16, 2024

Abstract

Multi-party computation (MPC) is a major research interest in modern cryptography, and Privacy Set Intersection (PSI) is an important research topic within MPC. Its main function is to allow two parties to compute the intersection of their private sets without revealing any other information. Therefore, PSI can be applied to various real-world scenarios, such as the Industrial Internet of Things (IIOT). Chase and Miao presented a paper on "Light-weight PSI" at CRYPTO 2020, highlighting its convenient structure and optimal communication cost. However, the drawback is that this protocol is deterministically encrypted and it was discovered through simulation that it is not resistant to probabilistic attacks. Building upon the ideas from CM20, this paper introduces the concept of a *perturbed pseudorandom generator*, constructs and proves its security, and replaces one of the hash functions (originally there were two) from CM20. In order to demonstrate the security of the PSI protocol proposed in this paper, a dedicated definition of Chosen Plaintext Attack (CPA) security model for this PSI protocol is provided. The paper then proceeds to prove that the PSI protocol satisfies this defined security model. Efficiency analysis shows that the PSI in this paper is comparable to CM20's PSI, whether on PC, pad, or phone.

Keywords: MPC; PSI; Pseudorandom generator.

1 Introduction

With the continuous development of communication and manufacturing technology, the industry is changing from traditional to intelligent production mode. The Industrial Internet of Things (IIoT) was introduced to improve industrial equipment monitoring, safety production management, and production process optimization [BATB20]. The proposal of "Industry4.0" in Germany and the "made in China 2025" plan indicate the beginning of the fourth industrial revolution [SSH+18]. As the core part of Industry 4.0, the smart factory is critical to industrial intelligence, using the IIoT to realize on-site IoT interconnection and data sharing. Hence, privacy protection and security sharing of data on the IIoT platform are important issues [ARKA+20, WLI+19].

Privacy set intersection (PSI) is a privacy protection technique [HFH99, CM20] that allows performing intersection operations between two or more data sets without exposing individual data. For the IIoT, PSI can bring several benefits:

[1] School of Mathematics and Statistics, Xidian University, Xi'an 710126, China; arcsec30@163.com

[2] School of Automation, Xi'an University of Posts and Telecommunications, Xi'an 710121, China

[3] School of Computer Science, Shaanxi Normal University, Xi'an, 710062, Shaanxi, China

- IIoT often involves the collection and processing of sensitive information, such as personal health data, location information, etc. Using PSI enables data analysis and sharing while protecting user privacy. Since intersection operations can be performed without exposing the data itself, only verifying the presence of data in the set.

- IIoT often involves the collection and processing of sensitive information, such as personal health data, location information, etc. Using PSI enables data analysis and sharing while protecting user privacy. Since intersection operations can be performed without exposing the data itself, only verifying the presence of data in the set.

- PSI enhances system security by allowing data matching and validation without exposing individual data to unauthorized third parties. This reduces the risks of data leakage and misuse during transmission and processing.

- PSI helps IIoT systems identify common interests or characteristics among different devices or users, enabling the provision of personalized services and intelligent decision-making. For example, in health monitoring, cross-analyzing health data from multiple users can provide personalized health recommendations and preventive measures.

In summary, PSI provides an effective mechanism for privacy protection and data sharing in the IIoT, promoting the development of data-driven intelligent applications and services while safeguarding user privacy rights.

In 2020, Chase and Miao presented a lightweight Private Set Intersection (PSI) protocol at CRYPTO 2020. The construction of this protocol (referred to as protocol CM20 hereafter) includes oblivious transfer (OT)[Rab05], pseudorandom function (PRF)[GGM86], hashing, symmetric-key, and bitwise operations. According to the Performance Evaluation in [CM20], the protocol indeed achieves a balance between computation cost and communication cost. However, protocol CM20's use of PRF and hashing renders the protocol weakly secure [HDWD23], i.e., deterministic encryption. Assuming the existence of an adversary capable of intercepting ciphertext, probabilistic knowledge, and the ability to estimate the probability distribution of discarded plaintext, in addition to assuming that the probability distribution of discarded plaintext is almost consistent with the present, the adversary can leverage probabilistic attacks to break protocol CM20 with a significant advantage that cannot be ignored.

The idea of this paper is inspired by CM20. To prevent probabilistic attacks from hypothetical adversaries, we introduce the definition of a *perturbed pseudorandom generator* and replace one of the hash functions (originally there were two) in the original CM20 scheme with a perturbed pseudorandom generator. This modification makes the revised CM20 protocol strongly secure, i.e., randomized encryption.

1.1 Related Work

While the majority of existing research has focused on two-party PSI settings, there has been relatively little attention given to multiparty PSI in the literature. This could be attributed to the perceived necessity for interaction between parties, which imposes significant communication costs and renders the protocol practically unfeasible. However, more recent studies, including [GN19, HV17, IOP18, KMP$^+$17], have proposed asymptotically efficient constructions for multiparty PSI in various security models. To put it another way, regardless of the specific cryptographic tools and primitives used in PSI protocols, the common approach is to designate a party that interacts individually with all other parties involved in executing the protocol. This approach resembles a star topology network and helps minimize intermediate exchanges between parties, which can help reduce communication overhead. However, a disadvantage of this approach is that it can place a heavy workload on the designated parties, making it challenging to implement in practical scenarios. Finding ways to distribute the workload among parties while maintaining efficiency is an active area of research in the field of PSI protocols. Recent approaches, such as those presented in [IOP18, KMP$^+$17], have provided concrete and valid constructions, further extended in the malicious security model by [BENOPC22].

In [DCW13], the authors propose a two-party PSI protocol that uses a variant of Bloom filters called garbled Bloom filters, which are based on OT. This protocol has been further extended in [RR16] to achieve malicious security using the cut-and-select technique. Other works, such as [BKM$^+$20, YCP$^+$22], focus on variants of PSI that involve performing different sets of computations at the intersection. Threshold PSI is also an area of interest, as shown by works like [BMRR21, BDP21, GS19, ZC18]. For example, in [ZC18], the authors introduce a protocol for thresholded PSI-based inadvertent polynomial evaluation.

Notably, [GS19] provides insights into the lower limit of communication complexity for two-party threshold PSI. Recent research in [BMRR21] extends these findings to multiparty scenarios, further highlighting the importance of efficient PSI protocols in practical applications. These works demonstrate the diversity and richness of the field of PSI protocols, where researchers continue to explore ways to improve efficiency, security, and scalability.

2 Technical Overview

First, we analyzed the potential attacks that protocol CM20 might face, namely probabilistic attacks. Below is an overview of protocol CM20.

0. P_1 and P_2 agree on security parameters λ, σ, protocol parameters $m, \omega, \ell_1, \ell_2$, a hash function $H_1 : \{0,1\}^* \to \{0,1\}^{\ell_1}$, and $H_2 : \{0,1\}^{\omega} \to \{0,1\}^{\ell_2}$, a pseudorandom function $F : \{0,1\}^{\lambda} \times \{0,1\}^{\ell_1} \to [m]^{\omega}$.
1. **Precomputation** Same as [CM20].
2. **Oblivious Transfer** Same as [CM20].
3. **OPRF Evaluation**
   (a) P_2 sends the PRF key k to P_1.
   (b) $\forall x \in \mathcal{X}$, P_1 computes $v = F_k(H_1(x))$ and its OPRF value $\psi = H_2(C_1[v[1]] \| \cdots \| C_\omega[v[\omega]])$ and sends ψ to P_2.
   (c) Let Ψ be the set of OPRF values received from P_1. $\forall y \in \mathcal{Y}$, P_2 computes $v = F_k(y)$ and its OPRF value $\psi = H_2(A_1[v[1]] \| \cdots \| A_\omega[v[\omega]])$ and outputs y iff $\psi \in \Psi$.

Figure 1: Protocol CM20

Assuming that an adversary can intercept the encrypted private set Ψ sent by P_1, and based on the known probability distribution of plaintext privacy, the adversary attempts to guess each plaintext corresponding to each ψ in the privacy set Ψ. Although the success rate is not 100%, the probability of success is certainly not negligible!

Therefore, we hope that the H_2 in Protocol CM20 has random perturbation, while ensuring that this perturbation has a clear upper bound, allowing for the identification of two ciphertexts obtained from encrypting the same plaintext using H_2. As a result, we propose a new cryptographic primitive, the perturbed pseudorandom generator, denoted as $G(\cdot)$. For $x_1, x_2 \in \mathcal{X}$, there exists γ such that:

- When $x_1 = x_2$, the probability that $G(x_1) = G(x_2)$ is $\le \exp(-\Omega(n))$,
- When $x_1 = x_2$, the distance between $G(x_1)$ and $G(x_2)$ is less than γ,

- When $x_1 \neq x_2$, there exists N such that the distance between $G(x_1)$ and $G(x_2)$ is $\geq \gamma \cdot N$, where it is clear that $N = 1$ is the optimal case.

In addition, we believe that the presence of perturbation in the pseudorandom generator enhances the security of the protocol to a higher level, making it no longer a deterministic protocol. Therefore, we attempt to provide the corresponding Chosen Plaintext Attack (CPA) security model [GSM18], although this model may only be applicable to the protocol in this paper and may not be suitable for other PSI protocols. In order to demonstrate the performance of the protocol in this paper and its suitability for application in IIOT, we simulate protocol CM20 and the protocol in this paper on the PC side, the pad on the mobile side, and the phone on the mobile side.

3 Preliminary

Definition 1 ([GSM18], page 32, Section 4.1.6)**.** *We say that $\varepsilon(n)$ is negligible associated with n if $\varepsilon(n)$ can be expressed as*

$$\varepsilon(n) = \frac{1}{O(e^n)},$$

and the notation $O(n)$ represents a quantity that grows at most as fast as n approaches infinity.

Definition 2 ([Ceg12], Definition 2.1.6)**.** *Let $\mathcal{H}$ be a Hilbert space, and let $T : \mathcal{H} \to \mathcal{H}$ be an operator. If $T(\cdot)$ satisfies*

$$\|Tx - Ty\| < \|x - y\|, \ \forall x,\ y \in \mathcal{H},$$

then $T(\cdot)$ is called a contraction operator.

Lemma 1 ([Ceg12], Proposition 2.1.11)**.** *If $\mathcal{H}$ is a closed set (every Cauchy sequence in $\mathcal{H}$ converges to a point within $\mathcal{H}$), and $T(\cdot)$ is a contraction operator, and $Fix(T)$ is a closed convex set, then the algorithm $x_{n+1} = Tx_n$ converges to some $x \in Fix(T)$, where $Fix(T)$ denotes the set of fixed points of the operator $T(\cdot)$.*

Remark 1. *The convergence mentioned in Lemma 1 should be considered as strong convergence. However, this paper does not discuss the difference between strong and weak convergence, because in finite dimensions strong and weak convergence are equivalent.*

Fact 1. *Suppose that $\mathbb{P}_{\{0,1\}} := \{\Pr(0), \Pr(1)\}$, $\mathbb{P}^{(1)}_{\{0,1\}} = \{\frac{1}{2}, \frac{1}{2}\}$, $\mathbb{P}^{(2)}_{\{0,1\}} = \{\frac{1}{\omega}, \frac{\omega-1}{\omega}\}$, then*

$$\begin{aligned}\mathbb{P}^{(1)}_{\{0,1\}} \odot \mathbb{P}^{(2)}_{\{0,1\}} := \mathbb{P}^{(3)}_{\{0,1\}} &:= \left\{\frac{1}{2}\cdot\frac{1}{\omega} + \frac{1}{2}\cdot\frac{1}{\omega} + \frac{1}{2}\cdot\frac{\omega-1}{\omega}, \frac{1}{2}\cdot\frac{\omega-1}{\omega}\right\} \\ &= \left\{\frac{1}{2} + \frac{1}{2\omega}, \frac{1}{2} - \frac{1}{2\omega}\right\}.\end{aligned}$$

Fact 2. *Suppose that $\mathbb{P}^{(3)}_{\{0,1\}} = \{\frac{1}{2} + \frac{1}{2\omega}, \frac{1}{2} - \frac{1}{2\omega}\} = \{\Pr(0), \Pr(1)\}$, then*

$\mathbb{P}_{\{0,1,2\}} := \{\overset{\prime}{\Pr}(0), \overset{\prime}{\Pr}(1), \overset{\prime}{\Pr}(2)\} := \mathbb{P}^{(3)}_{\{0,1\}} \uplus \mathbb{P}^{(3)}_{\{0,1\}}$. *Here,*

$$\begin{aligned}\overset{\prime}{\Pr}(0) &= \Pr(0)\Pr(0) = \left(\frac{1}{2} + \frac{1}{2\omega}\right)^2 = \frac{1}{3} + \frac{1}{2\omega} + \frac{1}{4\omega^2} - \frac{1}{12}, \\ \overset{\prime}{\Pr}(1) &= \Pr(0)\Pr(1) + \Pr(1)\Pr(0) = 2\left(\frac{1}{2} + \frac{1}{2\omega}\right)\left(\frac{1}{2} - \frac{1}{2\omega}\right) = \frac{1}{3} + \frac{1}{6} - \frac{1}{2\omega^2}, \\ \overset{\prime}{\Pr}(2) &= \Pr(1)\Pr(1) = \left(\frac{1}{2} - \frac{1}{2\omega}\right)^2 = \frac{1}{3} - \frac{1}{2\omega} + \frac{1}{4\omega^2} - \frac{1}{12}.\end{aligned}$$

Furthermore, there is also

$$\mathbb{P}'_{\{0,1,2\}} := \{\overset{''}{\Pr}(0), \overset{''}{\Pr}(1), \overset{''}{\Pr}(2)\} := \mathbb{P}_{\{0,1,2\}} \uplus \mathbb{P}_{\{0,1,2\}} \bmod 3. \textit{ Here,}$$

$$\begin{aligned} \overset{''}{\Pr}(0) &= \overset{'}{\Pr}(0)\overset{'}{\Pr}(0) + \overset{'}{\Pr}(1)\overset{'}{\Pr}(2) + \overset{'}{\Pr}(2)\overset{'}{\Pr}(1), \\ \overset{''}{\Pr}(1) &= \overset{'}{\Pr}(0)\overset{'}{\Pr}(1) + \overset{'}{\Pr}(1)\overset{'}{\Pr}(0) + \overset{'}{\Pr}(2)\overset{'}{\Pr}(2), \\ \overset{''}{\Pr}(2) &= \overset{'}{\Pr}(0)\overset{'}{\Pr}(2) + \overset{'}{\Pr}(1)\overset{'}{\Pr}(1) + \overset{'}{\Pr}(2)\overset{'}{\Pr}(0). \end{aligned} \tag{3.1}$$

The equation (3.1) *can be rewritten as*

$$\mathbb{P}'_{\{0,1,2\}} = \begin{pmatrix} \Pr'(0) & \Pr'(2) & \Pr'(1) \\ \Pr'(1) & \Pr'(0) & \Pr'(2) \\ \Pr'(2) & \Pr'(1) & \Pr'(0) \end{pmatrix} \begin{pmatrix} \Pr'(0) \\ \Pr'(1) \\ \Pr'(2) \end{pmatrix} = M_{\mathbb{P}_{\{0,1,2\}}} \mathbb{P}_{\{0,1,2\}}.$$

Fact 3. *For the sequence* $\mathbb{P}^{(n)}_{\{0,1,2\}} = M_{\mathbb{P}^{(n-1)}_{\{0,1,2\}}} \mathbb{P}^{(n-1)}_{\{0,1,2\}}$*, define*

$$\mathbb{P}^{(n)}_{\{0,1,2\}} = (a_n^{(0)}, a_n^{(1)}, a_n^{(2)}) = \left(\frac{1}{3} + \Delta_n^{(0)}, \frac{1}{3} + \Delta_n^{(1)}, \frac{1}{3} + \Delta_n^{(2)}\right).$$

Claim 1. *The sequence* $\mathbb{P}^{(n)}_{\{0,1,2\}} = M_{\mathbb{P}^{(n-1)}_{\{0,1,2\}}} \mathbb{P}^{(n-1)}_{\{0,1,2\}}$ *is a Cauchy sequence.*

Proof. To prove that $\mathbb{P}^{(n)}_{\{0,1,2\}} = M_{\mathbb{P}^{(n-1)}_{\{0,1,2\}}} \mathbb{P}^{(n-1)}_{\{0,1,2\}}$ forms a Cauchy sequence, it suffices to show that for any $\delta > 0$, there exists an$N > 0$ such that for any $n > N$,

$$\left\| \mathbb{P}^{(n)}_{\{0,1,2\}} - \mathbb{P}^{(n-1)}_{\{0,1,2\}} \right\| \le \delta.$$

Because

$$\mathbb{P}^{(n)}_{\{0,1,2\}} - \mathbb{P}^{(n-1)}_{\{0,1,2\}} = \begin{pmatrix} (\Delta_0^{(n-1)})^2 + 2\Delta_1^{(n-1)}\Delta_2^{(n-1)} \\ (\Delta_2^{(n-1)})^2 + 2\Delta_0^{(n-1)}\Delta_1^{(n-1)} \\ (\Delta_0^{(n-1)})^2 + 2\Delta_0^{(n-1)}\Delta_2^{(n-1)} \end{pmatrix} = \begin{pmatrix} \Delta_0^{(n-1)} & \Delta_2^{(n-1)} & \Delta_1^{(n-1)} \\ \Delta_1^{(n-1)} & \Delta_0^{(n-1)} & \Delta_2^{(n-1)} \\ \Delta_2^{(n-1)} & \Delta_1^{(n-1)} & \Delta_0^{(n-1)} \end{pmatrix} \begin{pmatrix} \Delta_0^{(n-1)} \\ \Delta_1^{(n-1)} \\ \Delta_2^{(n-1)} \end{pmatrix}.$$

And

$$\left\| \begin{pmatrix} \Delta_0^{(n-1)} & \Delta_2^{(n-1)} & \Delta_1^{(n-1)} \\ \Delta_1^{(n-1)} & \Delta_0^{(n-1)} & \Delta_2^{(n-1)} \\ \Delta_2^{(n-1)} & \Delta_1^{(n-1)} & \Delta_0^{(n-1)} \end{pmatrix} \right\| \le \left\| \begin{pmatrix} \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} \\ \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} \\ \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} & \frac{1}{6} - \frac{1}{2\omega^2} \end{pmatrix} \right\| = \frac{1}{2} - \frac{3}{2\omega^2}.$$

So, it is obtained that

$$\left\| \mathbb{P}^{(n)}_{\{0,1,2\}} - \mathbb{P}^{(n-1)}_{\{0,1,2\}} \right\| \le \left(\frac{1}{2} - \frac{3}{2\omega^2}\right)^{n-1} \left\| \mathbb{P}^{(1)}_{\{0,1,2\}} - \mathbb{P}^{(0)}_{\{0,1,2\}} \right\| \le \frac{1}{\sqrt{3}} \left(\frac{1}{2} - \frac{3}{2\omega^2}\right)^n.$$

□

Lemma 2. *For any initial vector* $a_0 = (a_0^{(0)}, a_0^{(1)}, a_0^{(2)})$*, where* $a_0^{(0)}, a_0^{(1)}, a_0^{(2)} \in [(\frac{1}{2} - \frac{1}{2\omega})^2, (\frac{1}{2} + \frac{1}{2\omega})^2]$ *and* $\sum_{i=0}^{2} a_0^{(i)} = 1$*, and* $\omega > 2$*, the matrix* M_{a_0} *is generated as follows:*

$$M_{a_0} = \begin{pmatrix} a_0^{(0)} & a_0^{(2)} & a_0^{(1)} \\ a_0^{(1)} & a_0^{(0)} & a_0^{(2)} \\ a_0^{(2)} & a_0^{(1)} & a_0^{(0)} \end{pmatrix}.$$

Then, let $a_{n+1} := M_{a_n} a_n := T a_n$*, then* $\{a_n\}_{n=0}^{\infty}$ *is a Cauchy sequence and converges to* $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$*.*

Proof. According to Claim 1, we know that $\begin{pmatrix} \Delta_0^{(n-1)} & \Delta_2^{(n-1)} & \Delta_1^{(n-1)} \\ \Delta_1^{(n-1)} & \Delta_0^{(n-1)} & \Delta_2^{(n-1)} \\ \Delta_2^{(n-1)} & \Delta_1^{(n-1)} & \Delta_0^{(n-1)} \end{pmatrix}$ is a contraction operator, and

$$\left\| \begin{pmatrix} \Delta_0^{(n-1)} & \Delta_2^{(n-1)} & \Delta_1^{(n-1)} \\ \Delta_1^{(n-1)} & \Delta_0^{(n-1)} & \Delta_2^{(n-1)} \\ \Delta_2^{(n-1)} & \Delta_1^{(n-1)} & \Delta_0^{(n-1)} \end{pmatrix} \right\| \le \frac{1}{2} - \frac{3}{2\omega^2}.$$

Therefore, the matrix $\begin{pmatrix} \Delta_0^{(n-1)} & \Delta_2^{(n-1)} & \Delta_1^{(n-1)} \\ \Delta_1^{(n-1)} & \Delta_0^{(n-1)} & \Delta_2^{(n-1)} \\ \Delta_2^{(n-1)} & \Delta_1^{(n-1)} & \Delta_0^{(n-1)} \end{pmatrix}$ is contractive, with $(0,0,0)$ being both a convergent point and a fixed point of this matrix sequence. Moreover, since $a_{n+1} := M_{a_n} a_n$ has been proven to be a Cauchy sequence, the sequence $\{a_n\}_{n=0}^{\infty}$ converges, and it converges to the fixed point of $T(\cdot)$. $\square$

Theorem 1. *Given $\{a_j\}_{j=0}^2$ and $\{s_j\}_{j=1}^2$ such that $a_j \in_R \mathbb{Z}_{\{0,1\}}$, $s \in \mathbb{Z}_{\{0,1\}}$, and the probability that the components of s equal 0 is $\frac{1}{\omega}$, while the probability that they equal 1 is $\frac{\omega-1}{\omega}$, where $\omega > 2$. Then for any $i = 0, 1, 2$, we have*

$$\max_{i=0,1,2} \left| \Pr\left(\sum_{j=0}^{2} (a_j s_j) = i \right) - \Pr(u = i) \right| \le \frac{1}{\sqrt{3}} \left(\frac{1}{2} - \frac{3}{2\omega^2} \right)^n.$$

Corollary 1. *For any $A \in \mathbb{Z}_{\{0,1\}}^{m\times n}$, $s \in \mathbb{Z}_{\{0,1\}}^n$ where the probability that the components of s equal 0 is $\frac{1}{\omega}$ and the probability that they equal 1 is $\frac{\omega-1}{\omega}$, with $\omega > 2$, and $u \in \mathbb{Z}_{\{0,1,2\}}^m$, then the indistinguishability probability between $As \bmod 3$ and u is bounded by $\frac{1}{\sqrt{3}}\left(\frac{1}{2} - \frac{3}{2\omega^2}\right)^n$.*

Corollary 2. *For any $A \in \mathbb{Z}_{\{0,1\}}^{m\times n}$, $s \in \mathbb{Z}_{\{0,1\}}^n$ where the probability that the components of s equal 0 is $\frac{1}{\omega}$ and the probability that they equal 1 is $\frac{\omega-1}{\omega}$, $e \in_R \mathbb{Z}_{\{0,1\}}^n$, $\omega > 2$, and $u \in \mathbb{Z}_{\{0,1,2,3\}}^m$, then the indistinguishability probability between $(As) \bmod 3 + e$ and u is bounded by $\frac{1}{\sqrt{3}}\left(\frac{1}{2} - \frac{3}{2\omega^2}\right)^n$.*

4 Analysis of PSI

4.1 The theoretical analysis of PSI

Fact 4. *The PSI scheme has been proven to be secure under inadvertent transmission, but the inadvertent transmission technique does not affect the determinism of matrix D. Therefore, in our simulation, we do not take into account the inadvertent transmission technique, and instead directly consider matrices A and C as matrix D.*

Fact 5. *Assuming $\mathcal{X}$ is the privacy set, for any $x \in \mathcal{X}$, with a sufficiently large sample size, we have*

$$\Pr(x) \approx \Pr(\psi) \approx \Pr\left(H_2(C_1[v[1]]\| \cdots \|C_\omega[v[\omega]])\right).$$

4.2 Instantiate CM12 on Python

```
def H1(A,x,q):
    c = np.dot(A,x)%q
    return c
```

Figure 2: Instantiate hash function H_1

```python
def generate_or_load_matrix(filename, shape):
    if os.path.exists(filename):
        with open(filename, 'rb') as f:
            matrix = pickle.load(f)
    else:
        matrix = np.random.randint(0, q, size=shape)
        with open(filename, 'wb') as f:
            pickle.dump(matrix, f)
    return matrix
```

Figure 3: Fixed storage pseudo-random function parameters

```python
def xcel():
    r = np.random.randint(0,100)
    if (r < 30) or (r == 30):
        x = np.array([2, 1])
    elif (r>30) and (r<55):
        x = np.array([2, 2])
    elif (r>54) and (r<80):
        x = np.array([2, 3])
    else:
        x = np.array([2, 4])
    return x
```

Figure 4: Assign a probability to each privacy element in $\mathcal{X}$.

```python
def calculate_proportions(lst):
    counts = {}
    proportions = {}
    total = len(lst)
    for item in lst:
        if tuple(item) in counts:
            counts[tuple(item)] += 1
        else:
            counts[tuple(item)] = 1
    for item, count in counts.items():
        proportion = (count / total) * 100
        proportions[item] = proportion
    return proportions
```

Figure 5: Analyzing the proportion of encrypted ciphertexts

The code for this article has been presented on [Sha24].

4.3 Simulate and analyze CM20

The tools used in the subsection are Python 3.8, the programs are performed on Vostro Dell PC Desktop 11th Gen Intel(R)Core(TM) i5-11400@2.60GHz 2.59GHz, RAM 8.00GB.

4.3.1 Simulated Attack Results Based on Actual Ratio: 30:24:25:21

Table 1: Actual Ratio: 30:24:25:21, Collecting 10 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(6,6) 40.00% (3,6) 30.00% (2,6) 30.00%	(3, 6) 50.00% (5, 6) 30.00% (6, 6) 20.00%	(5, 6) 40.00% (3, 6) 10.00% (2, 6) 20.00% (6, 6) 30.00%
Success or Failure	[2 1] Failure [2 2] Failure [2 3] Ambiguous [2 4] Failure	[2 1] Success [2 2] Ambiguous [2 3] Failure [2 4] Failure	[2 1] Failure [2 2] Failure [2 3] Failure [2 4] Failure

Table 2: Actual Ratio: 30:24:25:21, Collecting 100 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(2,6) 26.00% (5,6) 13.00% (3,6) 34.00% (6,6) 27.00%	(3,6) 31.00% (2,6) 27.00% (5,6) 27.00% (6,6) 15.00%	(6,6) 22.00% (2,6) 28.00% (3,6) 37.00% (5,6) 13.00%
Success or Failure	[2 1] Success [2 2] Success [2 3] Success [2 4] Success	[2 1] Success [2 2] Failure [2 3] Ambiguous [2 4] Failure	[2 1] Success [2 2] Success [2 3] Success [2 4] Success

Table 3: Actual Ratio: 30:24:25:21, Collecting 1000 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(2,6) 25.00% (5,6) 18.80% (6,6) 26.50% (3,6) 29.70%	(2, 6) 23.30% (3, 6) 34.90% (6, 6) 22.80% (5, 6) 19.00%	(3, 6) 27.80% (6, 6) 22.90% (2, 6) 25.00% (5, 6) 24.30%
Success or Failure	[2 1] Success [2 2] Success [2 3] Success [2 4] Success	[2 1] Success [2 2] Success [2 3] Success [2 4] Success	[2 1] Success [2 2] Failure [2 3] Failure [2 4] Failure

4.3.2 Simulated Attack Results Based on Actual Ratio: 32:28:22:18

Table 4: Actual Ratio: 32:28:22:18, Collecting 10 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(3,6) 60.00%	(6, 6) 30.00%	(3, 6) 50.00%
	(2,6) 20.00%	(5, 6) 10.00%	(2, 6) 20.00%
	(5,6) 20.00%	(3, 6) 20.00%	(5, 6) 20.00%
		(2, 6) 40.00%	(6, 6) 10.00%
Success or Failure	[2 1] Success	[2 1] Failure	[2 1] Success
	[2 2] Failure	[2 2] Failure	[2 2] Failure
	[2 3] Failure	[2 3] Failure	[2 3] Ambiguous
	[2 4] Failure	[2 4] Failure	[2 4] Ambiguous

Table 5: Actual Ratio: 32:28:22:18, Collecting 100 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(2,6) 26.00%	(3, 6) 37.00%	(2, 6) 30.00%
	(3,6) 29.00 %	(6, 6) 31.00%	(5, 6) 16.00%
	(6,6) 27.00 %	(5, 6) 11.00%	(3, 6) 29.00%
	(5,6) 18.00 %	(2, 6) 21.00%	(6, 6) 25.00%
Success or Failure	[2 1] Success	[2 1] Success	[2 1] Failure
	[2 2] Failure	[2 2] Success	[2 2] Failure
	[2 3] Failure	[2 3] Success	[2 3] Failure
	[2 4] Success	[2 4] Success	[2 4] Success

Table 6: Actual Ratio: 32:28:22:18, Collecting 1000 Samples

	First Group	Second Group	Third Group
Collected Ciphertext Ratio	(2,6) 21.20%	(3, 6) 32.90%	(5, 6) 16.00%
	(3,6) 34.90%	(6, 6) 28.40%	(6, 6) 31.10%
	(6,6) 27.20%	(2, 6) 20.00%	(3, 6) 32.50%
	(5,6) 16.70%	(5, 6) 18.70%	(2, 6) 20.40%
Success or Failure	[2 1] Success	[2 1] Success	[2 1] Success
	[2 2] Success	[2 2] Success	[2 2] Failure
	[2 3] Success	[2 3] Success	[2 3] Ambiguous
	[2 4] Success	[2 4] Success	[2 4] Failure

Conclusion 1. *Assuming the existence of such an adversary who can intercept ciphertext and possess probabilistic knowledge, including the ability to analyze the probability distribution of discarded plaintext privacy, if the probability distribution of discarded plaintext privacy is nearly consistent with the current one, then this adversary can exploit probabilistic attacks to decrypt the CM20 protocol with a significant advantage that cannot be ignored.*

5 Modification of the PSI Protocol

5.1 Construction and Security Proof of PRG with Perturbation

Definition 3 (PRG with perturbation). *A pseudorandom generator with perturbation, denoted as $G_\gamma(\cdot)$, is defined such that for $x_1, x_2 \in \mathcal{X}$, there exists γ satisfying the following conditions:*

1. *When $x_1 = x_2$, $\Pr(G_\gamma(x_1) = G_\gamma(x_2)) \leq \exp(-\Omega(n))$,*
2. *When $x_1 = x_2$, such that $\|G_\gamma(x_1) - G_\gamma(x_2)\| < \gamma$, there exists N such that $\|G_\gamma(x_1) - G_\gamma(x_2)\| \geq \gamma \cdot N$, where clearly $N = 1$ is optimal.*

Setup. Let $A \in \mathbb{Z}_{\{0,1\}}^{m\times n}$, $x \in \mathbb{Z}_{\{0,1\}}^{n}$, $e \in \mathbb{Z}_{\{0,1\}}^{m}$.

Enc. Compute

$$G_\gamma(x) = (Ax) \bmod 3 + e.$$

Figure 6: Pseudorandom generator with perturbation $G_\gamma(\cdot)$

Theorem 2. *Assume the construction of the PRG $G_\gamma(\cdot)$ as depicted in Figure 6, then $G_\gamma(\cdot)$ is indistinguishable from $u \in \mathbb{Z}_{\{0,1,2,3\}}^{m}$.*

Proof. It is straightforward to prove the correctness of Theorem 2 from Corollary 2. □

Theorem 3. *Assume the construction of the PRG $G_\gamma(\cdot)$ as depicted in Figure 6, then $G_\gamma(\cdot)$ satisfies Definition 3.*

Proof. We prove each statement separately. First, when $x_1 = x_2$, we have

$$\Pr(G_\gamma(x_1) = G_\gamma(x_2)) = \Pr(e_1 = e_2) = \frac{1}{2^n}.$$

Additionally, set $\gamma = \sqrt{n+1}$, so

$$\|(Ax_1 + e_1) - (Ax_2 + e_2)\| = \|e_1 - e_2\| < \gamma.$$

When $x_1 \neq x_2$, set $v_1 = G_\gamma(x_1)$, $v_2 = G_\gamma(x_2)$, and know that

$$\Pr(\|v_1 - v_2\| \leq \sqrt{n}) = \sum_{k=0}^{n} C_n^k \left(\frac{1}{3}\right)^k \left(\frac{1}{2}\right)^{n-k} + \sum_{k=0}^{n/2} C_n^k \left(\frac{1}{3}\right)^k \left(\frac{1}{6}\right)^k \left(\frac{1}{2}\right)^{n-2k}.$$

Because

$$\sum_{k=0}^{n} C_n^k \left(\frac{1}{3}\right)^k \left(\frac{1}{2}\right)^{n-k} = \frac{1}{2^n}\left(\frac{2}{3} + \left(\frac{2}{3}\right)^2 + \cdots + \left(\frac{2}{3}\right)^n\right) = \frac{3}{2^n}\left(1 - \left(\frac{2}{3}\right)^n\right),$$

and

$$\sum_{k=0}^{n/2} C_n^k \left(\frac{1}{3}\right)^k \left(\frac{1}{6}\right)^k \left(\frac{1}{2}\right)^{n-2k} \leq \frac{3\cdot 6}{17}\frac{1}{2^{n-\frac{n}{2}}}\left(1 - \left(\frac{1}{3\cdot 6}\right)^{\frac{n}{2}}\right).$$

Therefore

$$\Pr(\|v_1 - v_2\| \leq \sqrt{n} < \sqrt{n+1}) \leq \frac{1}{2^n}.$$

Thus, there is a very high probability that $\|v_1 - v_2\| \geq \sqrt{n+1}$, and $N = 1$. □

5.2 New PSI Protocol with PRG

0. P_1 and P_2 agree on security parameters λ, σ, protocol parameters $m, \omega, \ell_1, \ell_2$, a hash function $H_1 : \mathbb{Z}_3^* \to \{0,1\}^{\ell_1}$ and a $G_\gamma : \{0,1\}^\omega \to \mathbb{Z}_3^{\ell_2}$, a pseudorandom function $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$.
1. **Precomputation** Same as [CM20].
2. **Oblivious Transfer** Same as [CM20].
3. **OPRF Evaluation**
   (a) P_2 sends the PRF key k to P_1.
   (b) $\forall x \in \mathcal{X}$, P_1 computes $v = F_k(H_1(x))$ and its OPRF value $\psi = G_\gamma(C_1[v[1]]\| \cdots \|C_\omega[v[\omega]])$ and sends ψ to P_2.
   (c) Let Ψ be the set of OPRF values received from P_1. $\forall y \in \mathcal{Y}$, P_2 computes $v = F_k(y)$ and its OPRF value $\|\psi - G_\gamma(A_1[v[1]]\| \cdots \|A_\omega[v[\omega]])\| < \sqrt{\omega}\gamma$ and outputs y iff $\psi \in \Psi$.

Figure 7: Modified PSI OPRF Evaluation

5.3 Security proof

Lemma 3. *Assuming $f(y) \approx_C u_1$ and $g(u_1) \approx_C u_2$, then $(g \circ f)(y) \approx_C u_2$.*

Lemma 4. *Find a suitable pseudorandom function $\widetilde{F}_* : \{0,1\}^\lambda \times \mathbb{Z}_3^* \to [m]^\omega$. Assuming that the pseudo-random function $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$ and the hash function $H_1 : \mathbb{Z}_3^* \to \{0,1\}^{\ell_1}$ are indistinguishable, we have*

$$\widetilde{F}_*(y) \approx_C F(H_1(y)).$$

Proof. On one hand, because the pseudorandom $\widetilde{F}_* : \{0,1\}^\lambda \times \mathbb{Z}_3^* \to [m]^\omega$, for any k, $y \in \mathcal{Y} \subset \mathbb{Z}_3^k$, we have

$$\widetilde{F}_k(y) \approx_C u_\omega \in [m]^\omega.$$

On the other hand, due to the pseudorandom function $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$, for $u_{\ell_1} \in \{0,1\}^{\ell_1}$, we have

$$F_k(u_{\ell_1}) \approx_C u_\omega.$$

According to the property of the hash function, we have

$$H_1(y) \approx_C u_{\ell_1}.$$

Combining with Lemma 3, we then have

$$F_k(H_1(y)) \approx_C u_\omega.$$

Consequently,

$$\widetilde{F}_*(y) \approx_C F(H_1(y)).$$

□

Theorem 4. *If F is s PRF, H_1 is a collision resistant hash function, then the protocol in Fig.7 securely realizes $\mathcal{F}_{\mathsf{PSI}}$ in the semi-honest model when parameters $m, \omega, \ell_1, \ell_2$ are chosen as described in CM20。*

Proof. Perspective from P_1.

Hyb$_0$ P_1's view and P_2's output in the real protocol.

Hyb$_1$ Same as Hyb$_0$ except that on P_2's side, for each $i \in [\omega]$, if $s[i] = 0$,then sample $A_i \leftarrow \{0,1\}^m$ and compute $B_i = A_i \oplus D_i$; otherwise sample $B_i \leftarrow \{0,1\}^m$ and compute $A_i = B_i \oplus D_i$. This hybrid is identical to Hyb$_0$.

Hyb$_2$ Initialize an $m \times w$ binary matrix D to all 1's. Denote its column vectors by $D_1, \ldots, D_\omega$. Then $D_1 = \ldots = D_\omega = 1^m$. For $y \in \mathcal{Y}$, randomly select $v \leftarrow [m]^\omega$, and set $D_i[v[i]] = 0$ for all $i \in [\omega]$.

Hyb$_3$ Find a suitable pseudorandom function $\widetilde{F}_* : \{0,1\}^\lambda \times \mathbb{Z}_3^* \to [m]^\omega$. For $y \in \mathcal{Y}$, compute $v = \widetilde{F}_*(y)$, and set $D_i[v[i]] = 0$ for all $i \in [\omega]$.

Hyb$_4$ Let there be a pseudorandom function $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$ and a hash function $H_1 : \mathbb{Z}_3^* \to \{0,1\}^{\ell_1}$. For $y \in \mathcal{Y}$, compute $v = F(H_1(y))$, and set $D_i[v[i]] = 0$ for all $i \in [\omega]$.

Given that $\text{Hyb}_0 \approx_C \text{Hyb}_1 \approx_C \text{Hyb}_2 \approx_C \text{Hyb}_3$, and according to Lemma 4, we know that $\text{Hyb}_3 \approx_C \text{Hyb}_4$. Therefore, we have $\text{Hyb}_0 \approx_C \text{Hyb}_4$.

Perspective from P_2.

Hyb$_0$ P_2's view in the real protocol.

Hyb$_1$ $\psi \leftarrow \mathbb{Z}_3^{\ell_2}$, all other aspects are consistent with the real protocol.

Hyb$_2$ Introduce G_γ, let the initial matrices be $C_1 = \cdots = C_\omega = 1^m$, randomly select $v \in [m]^\omega$, set $C_i[v[i]] = 0$ for all $i \in [\omega]$. Compute $G_\gamma(C_1[v[1]] \| \cdots \| C_\omega[v[\omega]])$.

Hyb$_3$ Let the initial matrices be $C_1 = \cdots = C_\omega = 1^m$, find an appropriate pseudorandom function $\widetilde{F}_* : \{0,1\}^\lambda \times \mathbb{Z}_3^* \to [m]^\omega$. For $y \in \mathcal{Y}$, compute $v = \widetilde{F}_*(y)$, set $C_i[v[i]] = 0$ for all $i \in [\omega]$. Compute $G_\gamma(C_1[v[1]] \| \cdots \| C_\omega[v[\omega]])$.

Hyb$_4$ Let the initial matrices be $C_1 = \cdots = C_\omega = 1^m$, set a pseudorandom function $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$ and a hash function $H_1 : \mathbb{Z}_{\ell_2+1}^* \to \{0,1\}^{\ell_1}$. For $y \in \mathcal{Y}$, compute $v = F(H_1(y))$, set $C_i[v[i]] = 0$ for all $i \in [\omega]$. Compute $G_\gamma(C_1[v[1]] \| \cdots \| C_\omega[v[\omega]])$.

Similarly, it can be proven that $\text{Hyb}_0 \approx_C \text{Hyb}_4$. □

Definition 4 (CPA security model of the protocol in Fig.7)**.** *Assume there exists a perturbed pseudorandom oracle machine PrOM_γ (where γ is the upper bound on the norm of the perturbation in PrOM_γ), such that for an input x, it outputs two values: one is a random value y_0, and the other is a pseudorandom value y_1 with x as its input.*

Setup *The simulator $\mathcal{B}$ generates the necessary parameters for the algorithms. The adversary $\mathcal{A}$ chooses s and sends it to the simulator $\mathcal{S}$ using OT.*

Hash Queries, PRF Queries and PRG Queries *The adversary $\mathcal{A}$ sequentially performs hash function queries, pseudorandom function queries, and pseudorandom synthesizer queries.*

Challenge *The adversary $\mathcal{A}$ selects a private message m and sends it to the simulator $\mathcal{B}$. The simulator queries the hash function, pseudorandom function, and oblivious transfer values of the real scheme, inputs these results into the pseudorandom oracle machine PrOM_γ, obtains two ciphertexts c_0 and c_1, and sends them to the adversary $\mathcal{A}$.*

Guessing *After receiving the two ciphertexts c_0 and c_1, $\mathcal{A}$ guesses which ciphertext corresponds to the encryption of m and sends the guess back to the simulator $\mathcal{B}$.*

The advantage of the adversary $\mathcal{A}$ is defined as the advantage of the simulator $\mathcal{B}$ in distinguishing the outputs of PrOM_γ.

Note 1. *The PrOM mentioned in this paper differs from [JLLW23]. In [JLLW23], PrOM refers to a pseudorandom oracle machine that outputs random values when the adversary does not know the pseudorandom function key, and outputs pseudorandom function values based on the key known to the adversary when the key is known. This is a single-value output. However, the PrOM required in this paper outputs both of these values simultaneously, making it a multi-value output.*

Theorem 5. *If F is s PRF, H_1 is a collision resistant hash function, then the protocol in Fig.7 securely realizes $\mathcal{F}_{\mathsf{PSI}}$ in the definition 4.*

Proof. Suppose the adversary $\mathcal{A}_{P_1}$ can break the scheme with non-negligible advantage. Now, the simulator $\mathcal{S}$ simulates the scheme. Suppose there exists a black-box $G_\gamma^{black-box}$ such that

$$\begin{tikzcd} & y_0 = G_\gamma(x) \in \mathbb{Z}_3^{\ell_2}, \\ G_\gamma^{black-box}(x) \to (y_0, y_1) \arrow[ur] \arrow[dr] & \\ & y_1 \in_R \mathbb{Z}_3^{\ell_2}. \end{tikzcd}$$

Setup The simulator $\mathcal{S}$ generates some necessary parameters for the algorithms and selects an appropriate hash function $H_1 : \mathbb{Z}_3^* \to \{0,1\}^{\ell_1}$ and a $G_\gamma : \{0,1\}^\omega \to \mathbb{Z}_3^{\ell_2}$, as well as a PRF $F : \{0,1\}^\lambda \times \{0,1\}^{\ell_1} \to [m]^\omega$ with key $k \in \{0,1\}^\lambda$. The adversary $\mathcal{A}_{P_1}$ selects s and transmits s to the simulator $\mathcal{S}$ using OT.

H-Query, PRF-Query and PRG-Query The adversary $\mathcal{A}_{P_1}$ makes queries about the hash function, pseudorandom function, oblivious transfer values, and pseudorandom generator. The simulator $\mathcal{S}$ pre-establishes lists for handling H-Query, PRF-Query, and PRG-Query respectively.

H_1-Query For the i^{th} query $x_i \in \mathbb{Z}_3^*$ corresponding to the value of H_1, the simulator $\mathcal{S}$ selects from the hash value list if available, otherwise selects a random $X_i \in \{0,1\}^{\ell_1}$. Set $X_i = H_1(x_i)$ and update the list accordingly.

F-Query For the i^{th} query $y_i \in \{0,1\}^{\ell_1}$ corresponding to the value of F, the simulator $\mathcal{S}$ selects from the pseudorandom function value list if available, otherwise selects a random $Y_i \in \{0,1\}^\omega$. Set $Y_i = F(y_i, k)$ and update the list accordingly.

G_γ-Query For the i^{th} query $w_i \in \{0,1\}^\omega$ corresponding to the value of G'_γ, the simulator $\mathcal{S}$ selects from the pseudorandom generator value list if available, otherwise selects a random $W_i \in \mathbb{Z}_3^{\ell_2}$. Set $W_i = G'_\gamma(w_i)$ and update the list accordingly. Note that G'_γ is not $G_\gamma^{\text{black-box}}$.

Challenge $\mathcal{A}_{P_1}$ selects $m \in \mathcal{X}/\mathcal{Y}$ and sends it to $\mathcal{S}$. $\mathcal{S}$ using the corresponding hash function queries and pseudorandom function queries, inputs the queried values into the black-box G'_γ, obtaining ψ_0 and ψ_1, and then sends ψ_0, ψ_1 to $\mathcal{A}_{P_1}$.

Guess Based on the received ψ_0 and ψ_1, $\mathcal{A}_{P_1}$ guesses whether ψ_0 or ψ_1 is the ciphertext of the encrypted message m.

According to the assumption, if the adversary $\mathcal{A}_{P_1}$ can break the scheme with a non-negligible advantage, then the simulator $\mathcal{S}$ can also break the black-box G'_γ with a non-negligible advantage. This contradicts the assumption that G'_γ is secure. □

6 Efficiency Analysis for PSI

6.1 Efficiency Analysis on PC

The tools used in the subsection are Python 3.8, the programs are performed on Vostro Dell PC Desktop 10th Gen Intel(R)Core(TM) i5-11400@2.60GHz 2.59GHz, RAM 8.00GB. The hash function H_2

of CM20 is the built-in hash function in Python.

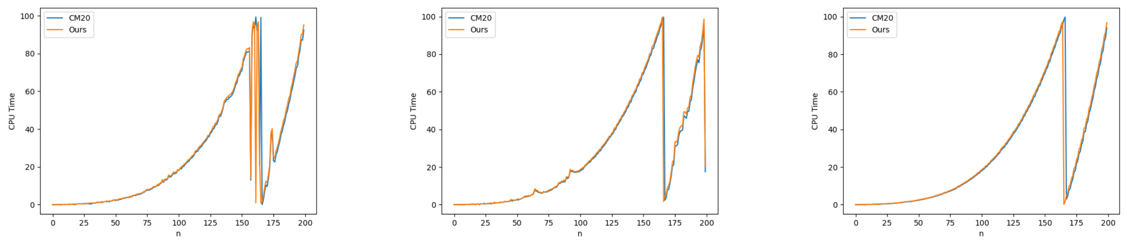

Figure 8: Parallel comparison of encryption on PC, where n represents the security parameter, unit is 10^4 microseconds

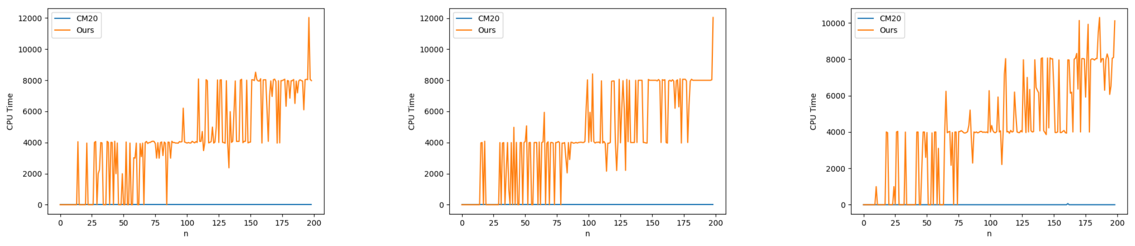

Figure 9: Parallel comparison of decryption on PC, where n represents the number of elements in P_2's private set, with time measured in microseconds

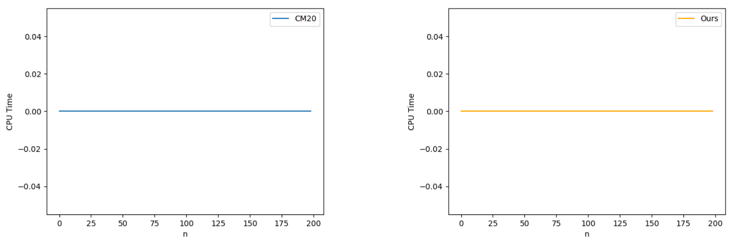

Figure 10: Parallel comparison of decryption on PC, where n represents the number of elements in P_2's private set, with time measured in seconds

6.2 Analysis of Efficiency on Mobile Pads

The tools used in the subsection are Pydriod 3, the programs are performed on Xiaomi Pad 6 Pro File Explorer 1th Qualcomm(R)AI Engine(TM) Xiaolong 8+ mobile platform@3.2GHz, RAM 8.00+3.00GB.

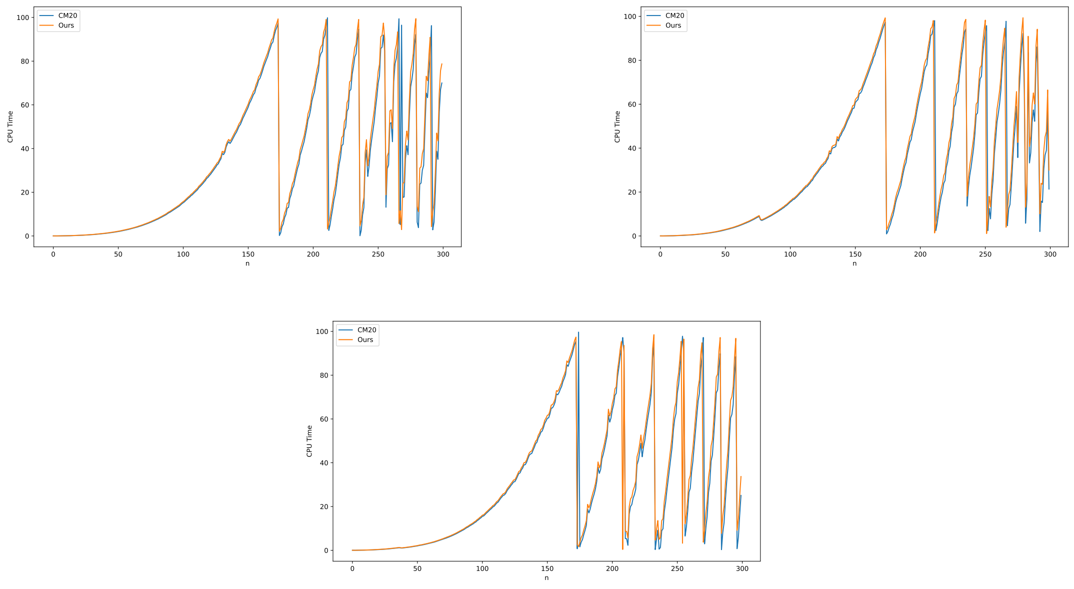

Figure 11: Parallel comparison of encryption on mobile pads, where n represents the security parameter, unit is 10^4 microseconds

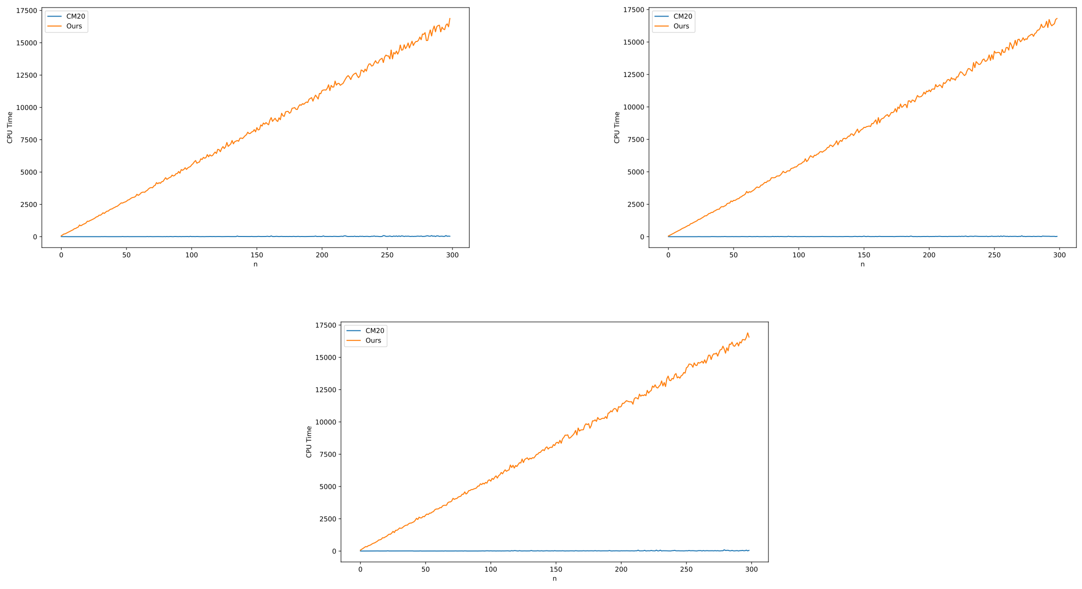

Figure 12: Parallel comparison of decryption on mobile pads, where n represents the number of elements in P_2's private set, with time measured in microseconds

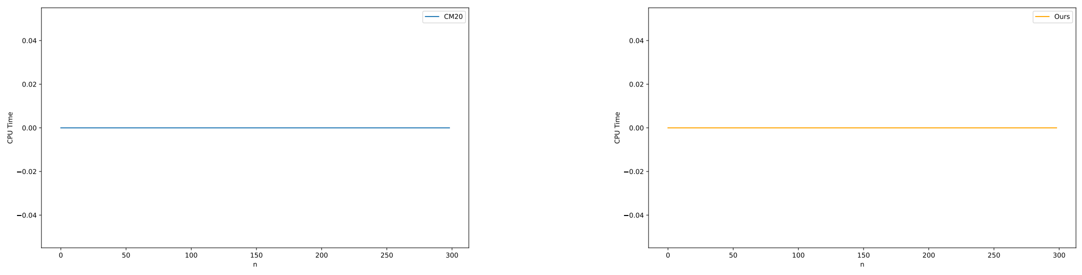

Figure 13: Parallel comparison of decryption on mobile pads, where n represents the number of elements in P_2's private set, with time measured in seconds

6.3 Analysis of Efficiency on Mobile Phones

The tools used in the subsection are Pydriod 3, the programs are performed on Redmi K30 File Explorer 4th Qualcomm(R)AI Engine(TM) Qualcomm Xiaolong 730G 8+ mobile platform@2.2GHz, RAM 6.00GB.

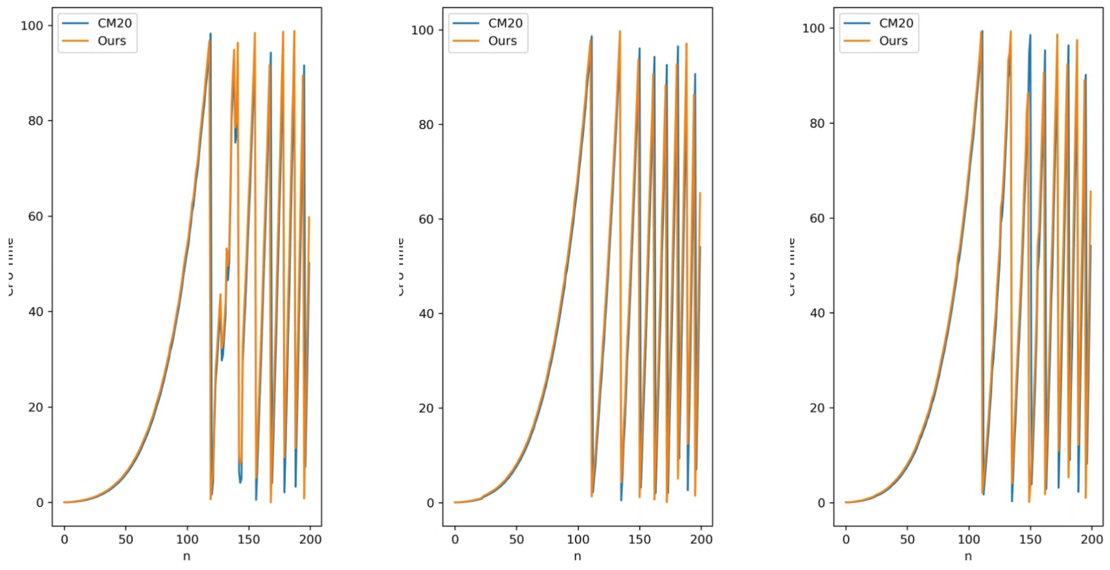

Figure 14: Parallel comparison of encryption on mobile phones, where n represents the security parameter, unit is 10^4 microseconds

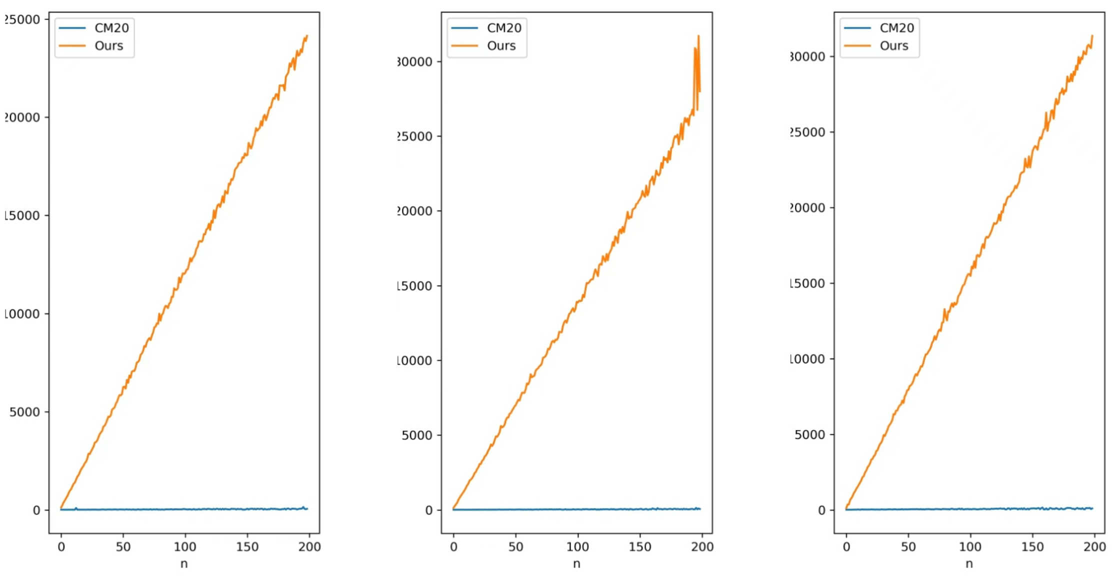

Figure 15: Parallel comparison of decryption on mobile phones, where n represents the number of elements in P_2's private set, with time measured in microseconds

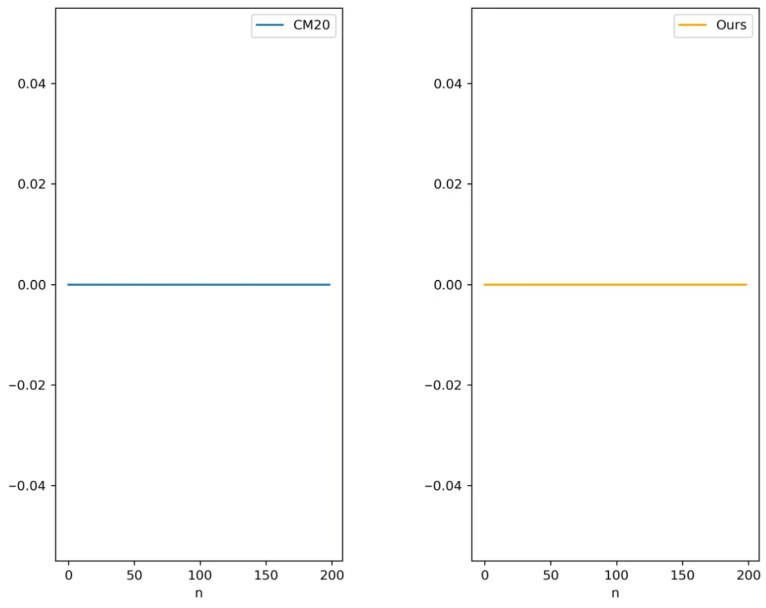

Figure 16: Parallel comparison of decryption on mobile phones, where n represents the number of elements in P_2's private set, with time measured in seconds

6.4 Summary of Data Comparison

Enc The encryption algorithm used in this paper shows similar efficiency to CM20, whether on PC or mobile devices. Among them, the CPU Times frequency of mobile device phone is significantly faster than that of PC and mobile device pad. The CPU times frequency of PC is slightly faster than that of mobile device pad. However, whether it is PC, mobile pad, or mobile phone, it seems that they have an upper limit of 100×10^4 milliseconds. Whether this boundary is suitable for the parameters depends on specific cases, which remains unknown.

Dec The decryption algorithm used in this paper is significantly higher than CM20. The computational overhead on PC shows a step-like pattern, while on mobile pad and phone, it roughly follows a linear trend. In comparison, the CPU times of CM20 can be almost disregarded.

References

[ARKA$^+$20] Ismaeel Al Ridhawi, Yehia Kotb, Moayad Aloqaily, Yaser Jararweh, and Thar Baker. A profitable and energy-efficient cooperative fog solution for iot services. *IEEE Transactions on Industrial Informatics*, 16(5):3578–3586, 2020.

[BATB20] Ouns Bouachir, Moayad Aloqaily, Lewis Tseng, and Azzedine Boukerche. Blockchain and fog computing for cyberphysical systems: The case of smart industry. *Computer*, 53(9):36–45, 2020.

[BDP21] Pedro Branco, Nico Döttling, and Sihang Pu. Multiparty cardinality testing for threshold private intersection. In *Public-Key Cryptography – PKC 2021*, pages 32–60, Cham, 2021. Springer International Publishing.

[BENOPC22] Aner Ben-Efraim, Olga Nissenbaum, Eran Omri, and Anat Paskin-Cherniavsky. Psimple: Practical multiparty maliciously-secure private set intersection. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, page 1098 – 1112, New York, NY, USA, 2022. Association for Computing Machinery.

[BKM$^+$20] Prasad Buddhavarapu, Andrew Knox, Payman Mohassel, Shubho Sengupta, Erik Taubeneck, and Vlad Vlaskin. Private matching for compute, 2020.

[BMRR21] Saikrishna Badrinarayanan, Peihan Miao, Srinivasan Raghuraman, and Peter Rindal. Multi-party threshold private set intersection with sublinear communication. In *Public-Key Cryptography – PKC 2021*, pages 349–379, Cham, 2021. Springer International Publishing.

[Ceg12] Andrzej Cegielski. *Iterative Methods for Fixed Point Problems in Hilbert Spaces.* 2012.

[CM20] Melissa Chase and Peihan Miao. Private set intersection in the internet setting from lightweight oblivious prf. In *Advances in Cryptology – CRYPTO 2020*, pages 34–63. Springer International Publishing, 2020.

[DCW13] Changyu Dong, Liqun Chen, and Zikai Wen. When private set intersection meets big data: An efficient and scalable protocol. pages 789–800. Association for Computing Machinery, 2013.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of ACM*, 33(4):792 – 807, 1986.

[GN19] Satrajit Ghosh and Tobias Nilges. An algebraic approach to maliciously secure private set intersection. In *Advances in Cryptology – EUROCRYPT 2019*, pages 154–185. Springer International Publishing, 2019.

[GS19] Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. In *Advances in Cryptology – CRYPTO 2019*, pages 3–29. Springer International Publishing, 2019.

[GSM18] Fuchun Guo, Willy Susilo, and Yi Mu. *Digital Signatures with Random Oracles.* Springer, 2018.

[HDWD23] Yupu Hu, Siyue Dong, Baocang Wang, and Xingting Dong. On the invalidity of lv16/lin17 obfuscation schemes revisited. Cryptology ePrint Archive, Paper 2023/1291, 2023. `https://eprint.iacr.org/2023/1291`.

[HFH99] Bernardo A. Huberman, Matthew K. Franklin, and Tad Hogg. Enhancing privacy and trust in electronic communities. In *ACM Conference on Economics and Computation*, 1999.

[HV17] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In *Public-Key Cryptography – PKC 2017*, pages 175–203. Springer Berlin Heidelberg, 2017.

[IOP18] Roi Inbar, Eran Omri, and Benny Pinkas. Efficient scalable multiparty private set-intersection via garbled bloom filters. In *Security and Cryptography for Networks*, pages 135–252. Springer International Publishing, 2018.

[JLLW23] Aayush Jain, Huijia Lin, Ji Luo, and Daniel Wichs. The pseudorandom oracle model and ideal obfuscation. In *Advances in Cryptology – CRYPTO 2023*, pages 233–262, Cham, 2023. Springer Nature Switzerland.

[KMP+17] Vladimir Kolesnikov, Naor Matania, Benny Pinkas, Mike Rosulek, and Ni Trieu. Practical multi-party private set intersection from symmetric-key techniques. pages 1257–1272. Association for Computing Machinery, 2017.

[Rab05] Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Paper 2005/187, 2005. `https://eprint.iacr.org/2005/187`.

[RR16] Peter Rindal and Mike Rosulek. Faster malicious 2-party secure computation with online/offline dual execution. pages 297–314. USENIX Association, 2016.

[Sha24] Zhuang Shan. Analysis and modify of cm20 on python, 2024.

[SSH+18] Emiliano Sisinni, Abusayeed Saifullah, Song Han, Ulf Jennehag, and Mikael Gidlund. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*, 14(11):4724–4734, 2018.

[WLI+19] Jiafu Wan, Jiapeng Li, Muhammad Imran, Di Li, and Fazal e Amin. A blockchain-based solution for enhancing security and privacy in smart factory. *IEEE Transactions on Industrial Informatics*, 15(6):3652–3660, 2019.

[YCP+22] Jason H. M. Ying, Shuwei Cao, Geong Sen Poh, Jia Xu, and Hoon Wei Lim. Psi-stats: Private set intersection protocols supporting secure statistical functions. In *Applied Cryptography and Network Security: 20th International Conference, ACNS 2022, Rome, Italy, June 20 – 23, 2022, Proceedings*, page 585 – 604, Berlin, Heidelberg, 2022. Springer-Verlag.

[ZC18] Yongjun Zhao and Sherman S. M. Chow. Can you find the one for me? 2018.