ChinaRxiv

AI translation · View original & related papers at
chinarxiv.org/items/chinaxiv-202304.00899

Secure Two-Party Comparison Against Untrusted Participants

Authors: Zhao Bowen, Zhu Yao, Xiao Yang, Pei Qingqi, Li Xiaoguo, Liu Ximen, Xiao Yang

Date: 2023-04-08T00:00:00+00:00

Abstract

Secure two-party comparison is widely used in constructing various secure computation protocols, such as secure machine learning training and inference. Existing secure two-party comparison protocols typically have one party learn the comparison result first and then inform the other party, making it difficult to prevent the party that learns the result first from tampering with the comparison result. To address this problem, this paper first proposes a new paradigm for secure two-party comparison that is resistant to untrusted participants. Subsequently, this paper designs a secure two-party comparison protocol satisfying the new paradigm using a threshold Paillier cryptosystem. This protocol allows both parties involved in the comparison to obtain a consistent comparison result without revealing their respective data, and the protocol guarantees that no participant can tamper with the comparison result. Rigorous theoretical analysis and proofs demonstrate that the proposed protocol is correct and secure. Experimental results show that the proposed protocol outperforms existing secure two-party comparison methods in terms of computational efficiency and functionality. Compared with existing secure two-party comparison methods, the computational efficiency of the protocol in this paper is improved by 50 times.

Full Text

A Novel Two-Party Comparison Protocol Against Untrusted Parties

Bo-Wen Zhao[1], Yao Zhu[1], Yang Xiao[2], Qing-Qi Pei[3], Xiao-Guo Li[4], Xi-Meng Liu[5]

[1](Guangzhou Institute of Technology, Xidian University, Guangzhou 510555, Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen, 518060)

[2](State Key Lab of Integrated Service Networks, School of Cyber Engineering, Xidian University, Xi'an 710126)
[3](State Key Laboratory of Integrated Services Networks and Shaanxi Key Laboratory of Blockchain and Secure Computing, Xidian University, Xi'an 710071)
[4](School of Computing and Information Systems, Singapore Management University, Singapore 178902)
[5](College of Computer Science and Data Science, Fuzhou University, Fuzhou 350108)

Abstract

Secure two-party comparison is widely used to build various secure computing protocols (e.g., secure training, secure inference). In existing secure two-party comparison protocols, there is always one party that obtains a comparison result first, and then notifies the other party of this result. Consequently, they cannot prevent the party that obtains the result first from tampering with it. To address this problem, this paper first proposes a new paradigm for secure two-party comparison against untrusted parties. Then, we design a secure two-party comparison protocol (TOMS) satisfying this new paradigm based on the threshold Paillier cryptosystem. Each party in TOMS obtains the same comparison result without revealing their private data. Moreover, TOMS prevents any party from tampering with the comparison results. Strict theoretical analyses demonstrate the security and correctness of TOMS. Finally, experimental results show that TOMS outperforms existing secure two-party comparison methods in terms of computational efficiency and functionality, achieving a 50× speedup over previous approaches.

Keywords: cryptography; trusted computing; secure comparison; secure two-party computation; homomorphic encryption; threshold cryptography

This work was supported by the National Key Research and Development Program of China (No. 2022YFB3102700) and the National Natural Science Foundation of China (Nos. 62202358, 62102295, 62072109, U1804263, 61632013).

Bo-Wen Zhao (male, Ph.D., associate professor, CCF member) researches privacy-preserving computation and its applications. E-mail: bwinzhao@gmail.com.
Yao Zhu (co-first author, male, M.S. candidate) researches privacy-preserving computation and its applications.
Yang Xiao (corresponding author, male, Ph.D., lecturer, CCF member) researches intelligent security and privacy protection. E-mail: yangtomas7@gmail.com.
Qing-Qi Pei (male, Ph.D., professor, CCF senior member) researches digital asset protection and network security.
Xiao-Guo Li (male, Ph.D., researcher) researches trusted computing and privacy-preserving computation.
Xi-Meng Liu (male, Ph.D., professor, CCF senior member) researches secure

computation and big data security.

1 Introduction

Secure two-party comparison enables two participants to jointly execute a comparison function $f(x, y)$ and obtain results u_a and u_b without revealing their respective input data x and y to each other, where $(u_a, u_b) \leftarrow f(x, y)$. This problem originated from Yao's Millionaires' Problem proposed in 1982, which has become a key technology in secure multi-party computation. Yao's original solution required exponential numbers of decryption and verification operations, resulting in enormous time and space complexity that limited practical applicability. To improve practicality, Yao later proposed a secure two-party computation scheme based on garbled circuits in 1986.

Since then, researchers have focused on designing more efficient solutions. Ioannidis et al. addressed secure two-party comparison by parallelizing n rounds of 1-out-of-2 oblivious transfer, where n is the bit length of the input data, achieving better efficiency than exponential operations. Li et al. solved the problem using set intersection combined with symmetric encryption, which improved efficiency but performed poorly with large inputs. Damgård implemented an integer comparison protocol using a specialized DGK homomorphic encryption scheme that offered superior efficiency, though initialization took approximately 150 seconds due to security considerations. Recent work also addresses security: Li et al. used ElGamal encryption for semi-honest models and extended to malicious models using zero-knowledge proofs and cut-and-choose protocols.

Secure two-party comparison has found applications in privacy-preserving auctions, machine learning, outsourced computation, and interval computation. Damle et al. proposed a solution for combinatorial auctions (TPACAS) that protects auction-related privacy. They later developed a verifiable solution using semi-trusted third-party proxies and Ethereum. Abspoel et al. applied Legendre symbol-based comparison to secure neural network classifiers, achieving 5× efficiency improvements over MPyC's built-in protocol on MNIST. Liu et al. designed a floating-point outsourcing framework (POCF) with a sub-protocol SLT for comparing ciphertexts. Liu et al. also transformed interval computation into secure comparison using zero-knowledge proofs and Goldwasser-Micali encryption. Guo et al. combined Paillier cryptosystems with geometric theory for efficient rational interval computation. Zhao et al. constructed secure sorting protocols based on ciphertext comparison to address privacy leaks in particle swarm optimization.

Despite these efforts to improve performance and apply secure two-party comparison, existing protocols overlook a critical vulnerability: untrusted participants can tamper with comparison results, undermining protocol reliability. Typically, one party (e.g., Alice) obtains the result u_a first, then notifies Bob of u_b. If Alice is trusted, she sets $u_b = u_a$ and both parties receive identical results. However, if Alice is untrusted, she can set $u_b \neq u_a$, causing Bob to

receive an incorrect result.

Problem Scenario: Two millionaires, Alice and Bob, wish to purchase a property, agreeing that only the wealthier one may buy it, yet neither wants to reveal their total assets. While existing protocols seem applicable, the millionaire who learns the result first can easily deceive the other, making it impossible for both to confidently determine who has purchasing rights.

To solve this, we propose methods for secure two-party comparison against untrusted participants. Technically, our approach ensures that Alice and Bob, with private inputs x and y, always obtain identical comparison results ($u_a = u_b$) after executing protocol $(u_a, u_b) \leftarrow f(x, y)$, with neither party able to tamper with the result. Our contributions are threefold:

1. We propose a novel paradigm for secure two-party comparison against untrusted parties that mandates identical results for both participants and prevents result tampering.
2. We design a concrete protocol (TOMS) using threshold Paillier cryptosystem key splitting and secure two-party computation principles to prevent untrusted participant attacks.
3. The protocol is provably secure and efficient. Rigorous analysis demonstrates that result tampering is computationally infeasible, and experiments show 50× speedup over comparable methods.

The remainder is organized as follows: Section 2 surveys related work. Section 3 presents our paradigm. Section 4 details the threshold Paillier cryptosystem, system model, threat model, and protocol design. Section 5 proves CPA security and compliance with our paradigm. Section 6 evaluates efficiency and feasibility. Section 7 concludes.

2 Related Work

We review three categories of secure two-party comparison protocols: garbled circuit-based, homomorphic encryption-based, and secret sharing-based approaches.

Garbled Circuit-Based Methods. These transform comparison problems into garbled circuits. ObliVM compiles ObliVM-lang (a Java-like language) to implement efficient ORAM schemes for semi-honest models. ABY is a C++ framework that enables fine-grained efficiency control through protocol conversion mechanisms. Both provide generic solutions (optimized with free-XOR) that can handle comparison. For malicious participants, cut-and-choose methods prevent attacks by having the sender construct multiple circuits and the receiver randomly verify subsets. Canetti et al. used this approach for correctness guarantees, but constructing numerous circuits yields high computational and space overhead, limiting practicality.

Homomorphic Encryption-Based Methods. Homomorphic encryption's ability to map ciphertext operations to plaintext operations makes it ideal for

privacy-preserving comparison. Lin et al. combined ElGamal with set intersection methods, comparing 0/1 encodings while protecting 1-encodings. Liu et al. represented wealth values as vectors and used Paillier's homomorphic properties. Li et al. proposed max/min comparison protocols for semi-honest and malicious models using ElGamal, vectorization, and zero-knowledge proofs. Liu et al. developed POCR for natural numbers using threshold Paillier, with sub-protocol SLT for ciphertext comparison. Zhao et al. improved this work with SOCI, whose SCMP sub-protocol supports integer comparison and resists chosen-plaintext attacks. Our TOMS protocol builds upon SCMP.

Secret Sharing-Based Methods. Nishide proposed an efficient bit-decomposition protocol using bitwise secret sharing. Liu et al. vectorized wealth values and used secret sharing instead of Paillier to reduce complexity for multi-party comparison. Damgård et al. transformed polynomial shares of secrets into bit shares, enabling efficient comparison via multiplication protocols.

Table 1 compares our protocol with representative schemes. SemiSMC uses homomorphic encryption, semiSSC uses secret sharing, while ObliVM and ABY use garbled circuits. Neither ObliVM nor ABY can prevent untrusted participants from tampering with results. Our TOMS protocol achieves this protection.

3 Paradigm for Secure Two-Party Comparison Against Untrusted Parties

As shown in Figure 1 [Figure 1: see original paper], in an ideal secure two-party comparison, Alice and Bob with private data x and y jointly execute function $(u_a, u_b) \leftarrow f(x, y)$ through a trusted third party to determine their relative magnitudes, without revealing inputs and with both obtaining correct, identical results $u_a = u_b$.

Definition 1. Let Alice and Bob execute two-party secure comparison protocol $f(x, y)$ with private inputs x and y, obtaining results u_a and u_b. A paradigm for secure two-party comparison against untrusted parties satisfies:

- **Confidentiality**: After protocol execution, Alice's input x does not leak to Bob, and vice versa.
- **Correctness**: After execution, Alice and Bob obtain results u_a and u_b where $u_a = u_b$ always holds. Both parties consistently receive identical comparison results.
- **Reliability**: During execution, the party that obtains the result first (e.g., Alice) cannot tamper with u_b to make $u_b \neq u_a$, and the later party (e.g., Bob) can easily detect any tampering.

4 Secure Two-Party Comparison Protocol Against Untrusted Parties

We first introduce the threshold Paillier cryptosystem, then describe TOMS's system model, threat model, detailed design, and correctness analysis.

4.1 Threshold Paillier Cryptosystem

We describe a $(2,2)$ threshold Paillier cryptosystem comprising key generation, encryption, decryption, key splitting, partial decryption, and threshold decryption. Unlike standard Paillier, the private key is split into two partial keys.

Key Generation (KeyGen): Let k be the security parameter. Choose large primes p, q with $|p| = |q| = k$. Compute $N = p \cdot q$, $\lambda = (p-1)(q-1)/2$, and $\mu = \lambda^{-1} \bmod N$. Define $L(x) = \frac{x-1}{N}$ and select generator $g = N+1$. The public key is $pk = (N, g)$ and private key is $sk = \lambda$.

Encryption (Enc): For plaintext $m \in \mathbb{Z}_N$, output ciphertext $[\![m]\!] = \mathrm{Enc}(pk, m) = g^m r^N \bmod N^2$, where r is randomly chosen from $\mathbb{Z}_N^*$.

Decryption (Dec): For ciphertext $[\![m]\!]$, output plaintext $m = \mathrm{Dec}(sk, [\![m]\!]) = \frac{L([\![m]\!]^\lambda \bmod N^2)}{L(g^\lambda \bmod N^2)} \bmod N$.

Key Splitting (KeyS): Input private key $sk = \lambda$, output split keys $sk_1 = \lambda_1$ and $sk_2 = \lambda_2$ satisfying $\lambda_1 + \lambda_2 \equiv 0 \bmod \lambda$ and $\lambda_1 + \lambda_2 \equiv 1 \bmod N$. By the Chinese Remainder Theorem, we compute $\delta = \lambda_1 + \lambda_2 = \lambda \cdot \mu \bmod (\lambda \cdot N)$. Choose λ_1 as a σ-bit random number and set $\lambda_2 = \lambda \cdot \mu + \eta \cdot \lambda N - \lambda_1$, where η is a non-zero integer.

Partial Decryption (PDec): Input ciphertext $[\![m]\!]$ and partial key sk_i ($i \in \{1,2\}$), output partial decryption result $M_i = \mathrm{PDec}([\![m]\!], sk_i) = [\![m]\!]^{\lambda_i} \bmod N^2$.

Threshold Decryption (TDec): Input partial results M_1, M_2, output plaintext $m = \mathrm{TDec}(M_1, M_2) = L(M_1 \cdot M_2 \bmod N^2) \cdot \mu \bmod N$.

The threshold Paillier cryptosystem provides additive and scalar multiplication homomorphisms: - **Additive**: $\mathrm{Dec}(sk, [\![m_1]\!] \cdot [\![m_2]\!] \bmod N^2) = m_1 + m_2 \bmod N$ - **Scalar Multiplication**: $\mathrm{Dec}(sk, [\![m]\!]^c \bmod N^2) = c \cdot m \bmod N$, where $c \in \mathbb{Z}_N$.

4.2 System Model

As shown in Figure 2 [Figure 2: see original paper], TOMS involves two participants, Alice with private data x and Bob with private data y. To protect privacy, Alice uses a $(2,2)$ threshold Paillier system while Bob uses standard Paillier. Specifically:

- **Alice**: Generates key pair (pk_a, sk_a) where $pk_a = (N_a, g_a)$. She splits sk_a into $(\lambda_1, \lambda_2) \leftarrow \mathrm{KeyS}(sk_a)$ and distributes λ_2 and pk_a to Bob.
- **Bob**: Generates Paillier key pair (pk_b, sk_b) where $pk_b = (N_b, g_b)$ and distributes pk_b to Alice.

4.3 Threat Model

TOMS involves two entities: Alice and Bob, both assumed **semi-honest**. They follow the protocol faithfully but may attempt to infer the other's input from intermediate values. Their goals are to: (1) learn the other's input, or (2) tamper with the other's final result. Specific adversarial behaviors include:

1. Semi-honest Alice inferring Bob's input y from messages received.
2. Semi-honest Bob inferring Alice's input x from messages received.
3. Semi-honest Alice (who obtains results first) tampering with Bob's result u_b (making $u_b \neq u_a$).

4.4 Detailed Design

Assume $x, y \in [-2^\ell, 2^\ell]$ where ℓ is an integer satisfying $2^\ell \ll N$ (e.g., $\ell = 32$). The TOMS protocol, shown in Figure 3 [Figure 3: see original paper], takes private inputs x and y and outputs comparison results u_a and u_b, formally $(u_a, u_b) \leftarrow F(x, y)$ where F satisfies our paradigm. The five-step process is:

Step 1: Bob encrypts y as $[\![y]\!]_{pk_b}$ and sends it to Alice.

Step 2: Alice encrypts x as $[\![x]\!]_{pk_b}$. She randomly selects $s \in \{0, 1\}$ and computes:

$$D = \begin{cases} [\![r_1(x - y + 1) + r_2]\!]_{pk_b} & \text{if } s = 0 \\ [\![r_1(y - x) + r_2]\!]_{pk_b} & \text{if } s = 1 \end{cases}$$

where $r_1 \in \{0, 1\}^\sigma \quad \{0\}$ and $r_2 \in \{0, 1\}^\kappa \quad \{0\}$ are random numbers (σ, κ are security parameters with $\ell < \sigma < \kappa < |N_b|/2$) satisfying $r_2 \leq N_b$ and $r_1 + r_2 > N_b$.

Step 3: Bob receives D and decrypts it to get $d = \mathrm{Dec}(sk_b, D)$. He sets $u_1 = 0$ if $d > N_b/2$, otherwise $u_1 = 1$. He encrypts u_1 as $[\![u_1]\!]_{pk_a}$ and sends it to Alice. Alice encrypts s as $[\![s]\!]_{pk_a}$, partially decrypts it to get $M_1 = \mathrm{PDec}([\![s]\!]_{pk_a}, \lambda_1)$, and sends $D, [\![s]\!]_{pk_a}, M_1$ to Bob.

Step 4: Alice decrypts $[\![u_1]\!]_{pk_a}$ to obtain u_1, computes $u_a = s - u_1$, and sends λ_2 to Bob.

Step 5: Bob receives λ_2, partially decrypts $[\![s]\!]_{pk_a}$ to get $M_2 = \mathrm{PDec}([\![s]\!]_{pk_a}, \lambda_2)$, then computes $s = \mathrm{TDec}(M_1, M_2)$. Finally, Bob computes $u_b = s - u_1$.

4.5 Correctness Analysis

Alice obtains correct results. In Step 2, with $x, y \in [-2^\ell, 2^\ell]$, random $r_1 \in \{0, 1\}^\sigma \quad \{0\}$ and $r_2 \in \{0, 1\}^\kappa \quad \{0\}$ satisfy $r_2 \leq N_b$ and $r_1 + r_2 > N_b/2$. We analyze cases based on s:

- **If** $s = 0$: $x - y + 1 \in [-2^{\ell+1} + 1, 2^{\ell+1} + 1]$, so $d = r_1(x - y + 1) + r_2 \in (N_b/2 + 2 - 2^{\sigma+\ell+1} - 2^\sigma, N_b/2 + 2^{\sigma+\ell+1} + 2^\sigma)$. Since $N_b/2 \gg 2^{\sigma+\ell+1}$ and

$0 < d < N_b$, Alice obtains u_1 in Step 4 (if $d \geq N_b/2$, $u_1 = 0$; otherwise $u_1 = 1$).

  - When $x \geq y$: $d \geq N_b/2$, so $u_1 = 0$ and $u_a = s - u_1 = 0$. TOMS outputs $u_a = 0$ representing $x \geq y$.
  - When $x < y$: $d < N_b/2$, so $u_1 = 1$ and $u_a = 0 - 1 = -1 \equiv N_b - 1 \pmod{N_b}$. Since $u_a > N_b/2$, we compute $u_a = N_b - u_a = 1$, representing $x < y$.

- **If** $s = 1$: $y - x \in [-2^{\ell+1}, 2^{\ell+1}]$, so $d = r_1(y - x) + r_2 \in (N_b/2 + 2 - 2^{\sigma+\ell+1} - 2^{\sigma}, N_b/2 + 2^{\sigma+\ell+1} + 2^{\sigma})$. Similar analysis shows Alice always obtains correct results.

Bob obtains correct results. Bob sends $[\![y]\!]_{pk_b}$ in Step 1. In Step 3, he decrypts D to get d, derives u_1 from d versus $N_b/2$, and obtains encrypted $[\![u_1]\!]_{pk_a}$. In Step 5, after receiving λ_2, he partially decrypts $[\![s]\!]_{pk_a}$ to get M_2, then fully decrypts to obtain $s = \mathrm{TDec}(M_1, M_2)$. Since Alice correctly obtains u_a by Step 4 and both parties use identical s and u_1, Bob correctly computes $u_b = s - u_1$.

Since $u_a = s - u_1$ and $u_b = s - u_1$, we have $u_a = u_b$ always holds.

5 Security Analysis

We first define CPA security and prove that the encryption scheme $r_1(x - y + 1) + r_2$ (or $r_1(y - x) + r_2$) used in Step 2 is CPA-secure. We then define semi-honest security and prove TOMS's security via simulation paradigm. Finally, we verify TOMS satisfies our paradigm.

5.1 CPA Security Definition

The experiment $\mathrm{PubK}^{\mathrm{CPA}}_{A,r_{1m}+r_2}(\sigma,\kappa)$ involves adversary A and challenger C:

1. A selects messages m_0, m_1 and sends them to C.
2. C randomly chooses $b \in \{0,1\}$, generates random r_1, r_2, and sends $r_{1m}b + r_2$ to A.
3. A outputs guess $b' \in \{0,1\}$.
4. If $b = b'$, the experiment returns 1 (adversary succeeds); otherwise 0.

Definition 2. The encryption scheme $r_{1m} + r_2$ is CPA-secure if for any probabilistic polynomial-time adversary A, there exists a negligible function $\mathrm{negl}(\sigma,\kappa)$ such that:

$$\Pr[\mathrm{PubK}^{\mathrm{CPA}}_{A,r_{1m}+r_2}(\sigma,\kappa) = 1] \leq \frac{1}{2} + \mathrm{negl}(\sigma,\kappa)$$

Theorem 1. For $m_0, m_1 \in [-2^{\ell}, 2^{\ell}]$, $r_{1,0}, r_{1,1} \in [2^{\sigma-1}, 2^{\sigma} - 1]$ and $r_{2,0}, r_{2,1} \in [2^{\kappa-1}, 2^{\kappa} - 1]$, the values $r_{1,0}m_0 + r_{2,0}$ and $r_{1,1}m_1 + r_{2,1}$ are computationally indistinguishable. Formally, challenger C randomly selects $m_b \in [-2^{\ell}, 2^{\ell}]$, $r_1 \in$

$[2^{\sigma-1}, 2^{\sigma}-1]$, $r_2 \in [2^{\kappa-1}, 2^{\kappa}-1]$ and computes $r_{1m}b + r_2$. For adversary A inferring $b' = b$ from $r_{1m}b + r_2$, we have $\Pr[b' = b \mid r_{1m}b + r_2] \leq \frac{1}{2} + \text{negl}(\kappa)$.

Proof. Since r_1, r_2 are randomly selected, $r_{1m}b + r_2$ is uniformly distributed in $[2^{\kappa-1} - 2^{\ell+\sigma} + 2^{\ell}, 2^{\kappa} + 2^{\ell+\sigma} - 2^{\ell} - 1]$. The adversary's success probability is maximized when $m_0 = -2^{\ell}$, $m_1 = 2^{\ell}$:

- If $b = 0$ and $r_{1m}b + r_2 \in [2^{\kappa-1} - 2^{\ell+\sigma} + 2^{\ell}, 2^{\kappa-1} + 2^{\ell+\sigma-1} - 1]$, A guesses correctly with probability 1.
- If $b = 1$ and $r_{1m}b + r_2 \in [2^{\kappa} - 2^{\ell+\sigma-1}, 2^{\kappa} + 2^{\ell+\sigma} - 2^{\ell} - 1]$, A guesses correctly with probability 1.

The maximum success probability is:

$$\Pr[b' = b \mid r_{1m}b + r_2] \leq \frac{3 \cdot 2^{\ell+\sigma} - 2^{\ell+1}}{2^{\kappa}}$$

Given $\ell < \sigma < \kappa$, we have $2^{\kappa} \gg 3 \cdot 2^{\ell+\sigma} - 2^{\ell+1}$, so there exists negligible function $\text{negl}(\kappa) = \frac{3 \cdot 2^{\ell+\sigma} - 2^{\ell+1}}{2^{\kappa}}$ satisfying the CPA security bound.

5.2 Semi-Honest Security Definition

Ideal Model: Let $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$ be a probabilistic polynomial-time function representing the secure computation, where $f(x, y) = (f_1(x, y), f_2(x, y))$. In the ideal model, Alice and Bob send inputs x and y to a Trusted Third Party (TTP), which computes $f(x, y)$ and returns $f_1(x, y)$ to Alice and $f_2(x, y)$ to Bob. The TTP reveals nothing beyond these outputs, making the ideal model maximally secure.

Simulation Paradigm: A real protocol π is secure if it leaks no more information than the ideal model. Specifically, a participant's view during protocol execution should be simulatable from only its input and output.

Let $\text{view}_1^{\pi}(x, y) = (x, r, m_1, \ldots, m_t)$ and $\text{view}_2^{\pi}(x, y) = (y, r', m_1', \ldots, m_t')$ denote Alice's and Bob's views during protocol π, where r, r' are random coin tosses and m_i, m_i' are received messages. Their outputs are $\text{output}_1^{\pi}(x, y)$ and $\text{output}_2^{\pi}(x, y)$.

Definition 3. Protocol π securely computes f if there exist probabilistic polynomial-time simulators S_1 and S_2 such that:

$$\{(S_1(x, f_1(x, y)), f_2(x, y))\}_{x,y} \equiv \{(\text{view}_1^{\pi}(x, y), \text{output}_2^{\pi}(x, y))\}_{x,y}$$
$$\{(S_2(y, f_2(x, y)), f_1(x, y))\}_{x,y} \equiv \{(\text{view}_2^{\pi}(x, y), \text{output}_1^{\pi}(x, y))\}_{x,y}$$

where $\equiv$ denotes computational indistinguishability.

5.3 Security Proof

Theorem 2. The secure two-party comparison protocol against untrusted parties is secure under semi-honest conditions.

Proof. We construct simulators S_1 and S_2 per Definition 3.

Constructing S_1: 1. Randomly selects $y' \in [-2^\ell, 2^\ell]$ such that $f_1(x,y) = f_1(x,y') = u_a$. Encrypts y' as $[\![y']\!]_{pk_b}$. 2. Randomly selects $r_1 \in \{0,1\}^\sigma \setminus \{0\}$, $r_2 \in \{0,1\}^\kappa \setminus \{0\}$, and $s \in \{0,1\}$. Encrypts s as $[\![s]\!]_{pk_a}$ and partially decrypts to get $M_1 = \text{PDec}([\![s]\!]_{pk_a}, \lambda_1)$. Computes $D' = [\![r_1(x - y' + 1) + r_2]\!]_{pk_b}$ if $s = 0$, else $D' = [\![r_1(y' - x) + r_2]\!]_{pk_b}$. 3. Decrypts D' to get $d' = \text{Dec}(sk_b, D')$. Sets $u'_1 = 0$ if $d' > N_b/2$, else $u'_1 = 1$. Encrypts u'_1 as $[\![u'_1]\!]_{pk_a}$. 4. Decrypts $[\![u'_1]\!]_{pk_a}$ to get u'_1, computes $u'_a = s - u'_1$. 5. Partially decrypts $[\![s]\!]_{pk_a}$ to get $M_2 = \text{PDec}([\![s]\!]_{pk_a}, \lambda_2)$, fully decrypts to get $s = \text{TDec}(M_1, M_2)$, and computes $u'_b = s - u'_1$.

The simulated view is $S_1(x, f_1(x,y)) = (x, pk_a, pk_b, [\![y']\!]_{pk_b}, s, [\![s]\!]_{pk_a}, M_1, M_2, D', d', u'_1, f_1(x,y'))$. Since $f_1(x,y) = f_1(x,y')$, we have $u_a = u'_a$, implying $u_1 = u'_1$, $d = d'$, and $D \equiv D'$. By Paillier's semantic security, $[\![y]\!]_{pk_b} \equiv [\![y']\!]_{pk_b}$ and $[\![s]\!]_{pk_a} \equiv [\![s]\!]_{pk_a}$. Thus:

$$\{(S_1(x, f_1(x,y)), f_2(x,y))\}_{x,y} \equiv \{(\text{view}_1^\pi(x,y), \text{output}_2^\pi(x,y))\}_{x,y}$$

Constructing S_2: 1. Randomly selects $\bar{x} \in [-2^\ell, 2^\ell]$ such that $f_2(x,y) = f_2(\bar{x},y) = u_b$. Encrypts y as $[\![y]\!]_{pk_b}$. 2. Randomly selects $\bar{r}_1 \in \{0,1\}^\sigma \setminus \{0\}$, $\bar{r}_2 \in \{0,1\}^\kappa \setminus \{0\}$, and $\bar{s} \in \{0,1\}$. Encrypts $\bar{s}$ as $[\![\bar{s}]\!]_{pk_a}$, partially decrypts to $\bar{M}_1 = \text{PDec}([\![\bar{s}]\!]_{pk_a}, \lambda_1)$. Computes $\bar{D} = [\![\bar{r}_1(\bar{x} - y + 1) + \bar{r}_2]\!]_{pk_b}$ if $\bar{s} = 0$, else $\bar{D} = [\![\bar{r}_1(y - \bar{x}) + \bar{r}_2]\!]_{pk_b}$. 3. Decrypts $\bar{D}$ to $\bar{d} = \text{Dec}(sk_b, \bar{D})$. Sets $\bar{u}_1 = 0$ if $\bar{d} > N_b/2$, else $\bar{u}_1 = 1$. Encrypts $\bar{u}_1$ as $[\![\bar{u}_1]\!]_{pk_a}$. 4. Decrypts $[\![\bar{u}_1]\!]_{pk_a}$ to $\bar{u}_1$, computes $\bar{u}_a = \bar{s} - \bar{u}_1$. 5. Partially decrypts $[\![\bar{s}]\!]_{pk_a}$ to $\bar{M}_2 = \text{PDec}([\![\bar{s}]\!]_{pk_a}, \lambda_2)$, fully decrypts to $\bar{s} = \text{TDec}(\bar{M}_1, \bar{M}_2)$, computes $\bar{u}_b = \bar{s} - \bar{u}_1$.

The simulated view is $S_2(y, f_2(x,y)) = (y, pk_a, pk_b, [\![\bar{s}]\!]_{pk_a}, \bar{M}_1, \bar{M}_2, \bar{D}, \bar{d}, \bar{u}_1, f_1(\bar{x},y))$. Since $f_2(x,y) = f_2(\bar{x},y)$, we have $u_b = \bar{u}_b$, implying $\bar{s} \equiv s$, $\bar{u}_1 \equiv u_1$, and by Theorem 1, $d \equiv \bar{d}$ and $D \equiv \bar{D}$. By Paillier's semantic security, $M_1 \equiv \bar{M}_1$ and $M_2 \equiv \bar{M}_2$. Thus:

$$\{(S_2(y, f_2(x,y)), f_1(x,y))\}_{x,y} \equiv \{(\text{view}_2^\pi(x,y), \text{output}_1^\pi(x,y))\}_{x,y}$$

Reliability: Since $d = r_1(y - x) + r_2$ (or $d = r_1(x - y + 1) + r_2$), adversary Alice cannot change Bob's final result u_b by selecting a specific d'. By Theorem 1, d and d' are computationally indistinguishable, so Alice cannot tamper with u_b. Therefore, TOMS satisfies the reliability requirement of our paradigm.

6 Experimental Evaluation

We verify that TOMS satisfies our paradigm (confidentiality, correctness, reliability) and compare its performance with ObliVM, ABY, and semiSMC.

Confidentiality: Neither Alice's nor Bob's inputs leak. Alice's x is protected because Bob receives only $D = [\![r_1(x - y + 1) + r_2]\!]_{pk_b}$ or $[\![r_1(y - x) + r_2]\!]_{pk_b}$,

which is CPA-secure per Theorem 1. Bob's y is protected because Alice only receives $[\![y]\!]_{pk_b}$ without sk_b.

Correctness: Proven in Section 4.5.

Reliability: Alice cannot tamper with Bob's result $u_b = s - u_1$ because d and d' are computationally indistinguishable (Theorem 1), making it infeasible for Alice to manipulate u_1 by choosing a specific d'.

6.1 Efficiency Analysis

We measure efficiency via computational and communication complexity. Since modular exponentiation dominates, we count exponentiations while ignoring lighter operations. Pre-computation phases are excluded.

Computational Complexity: For TOMS with $N_a = N_b = N$: 1. Bob encrypts y: $1.5|N|$ modular multiplications. 2. Alice encrypts x and s: $3|N|$ multiplications; generates D: $1.5|N| + 3\sigma$; partial decryption: 1.5σ. 3. Bob decrypts D: $1.5|N|$; encrypts u_1: $1.5|N|$. 4. Alice decrypts u_1: $1.5|N|$. 5. Bob partial decryption: 1.5σ.

Total: $10.5|N| + 6\sigma$ modular multiplications.

For semiSMC (configured for two parties), encryption requires $6v|N|$ multiplications and decryption averages $3v|N|$, totaling $9v|N|$, where v is the input domain size. Since $v \gg 2$ and $|N| \gg \sigma$, TOMS is more efficient.

ObliVM and ABY use garbled circuits; their complexity depends on AND gate count (denoted as $\perp$ in Table 2), making direct comparison difficult. Experiments confirm TOMS's superior efficiency.

Communication Complexity: ObliVM and ABY require 4 rounds (garbled table, OT, result). semiSMC needs $n(n-1)$ rounds for n parties. TOMS uses 4 rounds. Communication overhead: TOMS (1.25 KB) « semiSMC (49.9 KB) < ABY (10.29 KB) < ObliVM (14.8 KB).

6.2 Experimental Tests

Experiments run on Intel Core i5-8300H CPU @ 2.30GHz, 16GB RAM, Windows 11 64-bit, using C++ and GMP 6.2. Pre-computation optimizes TOMS by pre-calculating values like $[\![s]\!]_{pk_a}$ before execution.

Test 1: Varying security parameter N (512–1024 bits) with $\ell = 32$ and $\ell = 64$ bits. As shown in Figures 4 [Figure 4: see original paper] and 5 [Figure 5: see original paper], ObliVM and ABY's runtime is independent of N (200.8 ms and 13.9 ms for $\ell = 32$; 340.1 ms and 17.5 ms for $\ell = 64$). TOMS and semiSMC runtime increase with N. At $N = 1024$ bits, TOMS outperforms ObliVM by 189 ms (32-bit) and 327.6 ms (64-bit), ABY by 2.1 ms (32-bit) and 5 ms (64-bit), and semiSMC by 592 ms (51.2× speedup for 32-bit) and 598 ms (48.8× speedup for 64-bit).

Test 2: Varying plaintext length ℓ (8–64 bits) with fixed $\sigma = 128$ and $N \in \{512, 1024, 2048, 4096\}$. As shown in Figure 6 [Figure 6: see original paper], TOMS runtime is independent of ℓ for a given N (e.g., ~12.8 ms at $N = 1024$ across all ℓ). This is because Paillier's random factor r makes ciphertext size independent of plaintext length.

Results confirm TOMS achieves both our paradigm's security guarantees and superior efficiency.

7 Conclusion

To address the limitation that existing secure two-party comparison protocols cannot resist untrusted participants, this paper proposes a novel protocol paradigm and a concrete instantiation, TOMS, based on threshold Paillier. Compared to existing protocols, TOMS ensures that all participants obtain identical, tamper-proof comparison results while significantly improving efficiency. Future work will extend this paradigm to general secure multi-party computation against untrusted participants.

References

[1] YAO A C. Protocols for secure computations//23rd annual symposium on foundations of computer science (sfcs 1982), 1982: 160-164.
[2] YAO A C. How to generate and exchange secrets//27th Annual Symposium on Foundations of Computer Science (sfcs 1986), 1986: 162-167.
[3] Ioannidis I, GRAMA A. An efficient protocol for yao's millionaires' problem//36th Annual Hawaii International Conference on System Sciences, 2003: 6-pp.
[4] Li S D, DAOSHUN W, YIQI D, et al. Symmetric cryptographic solution to yao's millionaires' problem and an evaluation of secure multiparty computations. Information Sciences, 2008, 178(1): 244-255.
[5] DAMGARD I, GEISLER M, KROIGARD M. Homomorphic encryption and secure comparison. International Journal of Applied Cryptography, 2008, 1(1): 22-31.
[6] Li S D, Wang W L, Du R M. Protocol for millionaires' problem in malicious model. Scientia Sinica Informationis, 2021, 51(01): 75-88.
[7] DAMLE S, FALTINGS B, GUJAR S. A practical solution to yao's millionaires' problem and its application in designing secure combinatorial auction. arXiv preprint arXiv:1906.06567, 2019.
[8] DAMLE S, FALTINGS B, GUJAR S. Blockchain-based practical multi-agent secure comparison and its application in auctions//International Conference on Web Intelligence and Intelligent Agent Technology, 2021: 430-437.
[9] ABSPOEL M, BOUMAN N J, SCHOENMAKERS B, et al. Fast secure comparison for medium-sized integers and its application in binarized neural networks//Cryptographers' Track at the RSA Conference, 2019: 453-472.
[10] LIU X, DENG R H, DING W, et al. Privacy-preserving outsourced

calculation on floating point numbers. Transactions on Information Forensics and Security, 2016, 11(11): 2513-2527.
[11] LIU X, ZHANG R, XU G, et al. Confidentially judging the relationship between an integer and an interval against malicious adversaries and its applications. Computer Communications, 2021, 180: 115-125.
[12] Guo Y M, Zhou S F, Dou J W, et al. Efficient privacy-preserving interval computation and its applications. Chinese Journal of Computers, 2016 (39): 1-17.
[13] Zhao B, Liu X, Song A, et al. PRIMPSO: A Privacy-Preserving Multiagent Particle Swarm Optimization Algorithm. IEEE Transactions on Cybernetics, 2022.
[14] LIU C, WANG X S, NAYAK K, et al. Oblivm: A programming framework for secure computation//2015 IEEE Symposium on Security and Privacy, 2015: 359-376.
[15] DEMMLER D, SCHNEIDER T, ZOHNER M. Aby-a framework for efficient mixed-protocol secure two-party computation.//NDSS, 2015.
[16] KOLESNIKOV V, SCHNEIDER T. Improved garbled circuit: Free xor gates and applications//International Colloquium on Automata, Languages, and Programming, 2008: 486-498.
[17] RAN POBURINNAYA Equivocating yao: VENKITASUBRAMANIAM M. constant-round adaptively secure multiparty computation in the plain model//the 49th Annual ACM SIGACT Symposium, 2017.
[18] LIN H Y, TZENG W G. An efficient solution to the millionaires' problem based on homomorphic encryption//International Conference on Applied Cryptography and Network Security, 2005: 456-466.
[19] LIU X, LI S, CHEN X, et al. Efficient solutions to two-party and multiparty millionaires' problem. Security and Communication Networks, 2017, 2017.
[20] Li S D, Xu W T, Wang W L, et al. Secure Maximum (Minimum) Computation in Malicious Model. Chinese Journal of Computers, 2021, 44(10): 2076-2089.
[21] LIU X, CHOO K K R, DENG R H, et al. Efficient and privacy-preserving outsourced calculation of rational numbers. IEEE Transactions on Dependable and Secure Computing, 2016, 15(1): 27-39.
[22] ZHAO B, YUAN J, LIU X, et al. SOCI: A toolkit for secure outsourced computation on integers. Transactions on Information Forensics and Security, 2022, 17: 3637-3648.
[23] NISHIDE T, OHTA K. Multiparty computation for interval, equality, and comparison without bit-decomposition protocol//International Workshop on Public Key Cryptography, 2007: 343-360.
[24] DAMGÅRD I, FITZI M, KILTZ E, et al. Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation//Theory of Cryptography Conference, 2006: 285-304.
[25] PAILLIER P. Public-key cryptosystems based on composite degree residuosity//International conference on the theory and applications of cryptographic techniques, 1999: 223-238.
[26] PEI D, SALOMAA A, DING C. Chinese remainder theorem: applications

in computing, coding, cryptography. World Scientific, 1996.
[27] KATZ J, LINDELL Y. Introduction to modern cryptography. CRC press, 2020.
[28] ODED G. Foundations of cryptography: volume 2, basic applications. Cambridge university press, 2009.

Bo-Wen Zhao received his Ph.D. degree in cyberspace security from South China University of Technology, China, in 2020. He is currently an associate professor at Guangzhou Institute of Technology, Xidian University. His research interests include privacy-preserving computation. E-mail: bwinzhao@gmail.com

Yao Zhu received his B.S. degree from Hubei University of Technology and is currently pursuing his M.S. degree at Guangzhou Institute of Technology, Xidian University. His research interests include privacy-preserving computation and its applications.

Yang Xiao received his B.S. and Ph.D. degrees in communication engineering from Xidian University in 2013 and 2020. From 2017-2019, he was a visiting Ph.D. student at the University of New South Wales supported by the China Scholarship Council. He is currently a Lecturer at the State Key Laboratory of Integrated Services Networks, School of Cyber Engineering, Xidian University. His research interests include social networks, joint recommendations, graph neural networks, trust evaluation, and blockchain. E-mail: yangtomas7@gmail.com

Qing-Qi Pei received his Ph.D. in Computer Science and Cryptography from Xidian University in 2008. He is a Professor at the State Key Laboratory of Integrated Services Networks, Senior Member of CCF and Chinese Institute of Electronics. His research focuses on digital content protection and wireless network security.

Xiao-Guo Li received his Ph.D. in computer science from Chongqing University in 2019. He was a Postdoctoral Research Fellow at Hong Kong Baptist University (2019-2021) and is currently a Research Fellow at Singapore Management University. His research interests include trusted computing and privacy-preserving computation.

Xi-Meng Liu received his Ph.D. in Cryptography from Xidian University in 2015. He is a full professor at the College of Computer Science and Data Science, Fuzhou University, and a research fellow at Peng Cheng Laboratory, Shenzhen. He received the ACM SIGSAC China Rising Star Award (2018), "Minjiang Scholars" Distinguished Professor, and "Qishan Scholars" awards. His research interests include secure computation, applied cryptography, and big data security.

Note: Figure translations are in progress. See original paper for figures.

Source: ChinaXiv — Machine translation. Verify with original.