An Effective Security Comparison Protocol in Cloud Computing

Yuling Chen
Guizhou University
Junhong Tao
Guizhou University
Tao Li (litao2019@qfnu.edu.cn)
Guizhou University
Jiangyuan Cai
POWERCHINA Guizhou Electric Power Engineering Co.,Ltd
Xiaojun Ren
Weifang University of Science and Technology

Research Article

Keywords: secure comparison protocols, zero-knowledge proof, homomorphic encryption, cloud computing

Posted Date: September 16th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2056175/v1

Additional Declarations: No competing interests reported.

RESEARCH

An Effective Security Comparison Protocol in Cloud Computing

Yuling Chen[1,3], Junhong Tao[1], Tao Li[1*], Jiangyuan Cai[4] and Xiaojun Ren[2]

[*]Correspondence:
litao2019@qfnu.edu.cn
[1]State Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang, China
Full list of author information is available at the end of the article

Abstract

Secure comparison protocol is an important branch of secure multi-party computation(SMPC), which compares the size of input data without disclosing any information between participants. The development of cloud computing provides an application platform for SMPC, but it also brings new challenges. In cloud computing with SMPC, clients need to process their own data and submit the processed data to a cloud server, which then performs the computation. In this process, not only the clients need to maintain an honest state at all times, but sensitive data on the cloud server side may also be exposed. In this paper, zero-knowledge proof and homomorphic encryption techniques are used to improve Damgård-Geisler-KrØigaard(DGK) comparison protocol. The improved secure comparison protocol can not only safely calculate private data, but also be applicable to malicious participant model. Finally, the security analysis shows that the proposed scheme not only ensures the privacy security of participants, but also ensures the data fairness of comparison protocols.

Keywords: secure comparison protocols; zero-knowledge proof; homomorphic encryption; cloud computing

Introduction

Nowadays, due to the continuous progress of network technology, many security fields need cooperative computing to solve some security problems [1]. Including edge computing, secure multi-party computation, deep learning, cloud computing, federated learning, Internet of Things, etc [2] [3]. At the same time, participants need to provide sensitive data for cooperative calculation, which also brings security challenges to data privacy. Not only are they vulnerable to attacks by malicious adversaries during data transmission [4], but also sensitive data is easy to leak during data fusion [5]. For example, in a multi-party cloud computing scenario, participants need to truthfully submit their private data to the cloud server, so private data is vulnerable to malicious attacks or interception during transmission, and the cloud server cannot fully guarantee data privacy security. This paper proposed a comparison protocol is based on partially homomorphic encryption can solve this problem, that is, each client uses homomorphic encryption algorithm to encrypt its own secret input, and then sends the resulting ciphertext to the cloud server. Finally, the cloud server performs operations on the ciphertext and sends the comparison results to each client. In the whole process, the cloud server knows nothing about the data information.

In the scheme proposed in this paper, partially homomorphic encryption ensures that data is operational, while zero-knowledge proof [6] is used to prevent participants from submitting false or incorrect data, 1-out-of-n Oblivious Transfer (OT) technology [7] is used to ensure that the remote cloud server cannot obtain any private data input by any participant, and ultimately all parties can only get the result of the comparison agreement.

Related Work

With the development of 5G technology [8] and the popularity of intelligent mobile devices, although it provides convenience for people's life, the huge amount of data also brings severe challenges to various fields [9] [10]. Terminal devices cannot process such huge and sensitive data, such as artificial intelligence, Internet of Things(IoT), wireless sensors, machine learning and so on [11] [12]. The emergence of cloud computing effectively solves this problem. Cloud computing is a technology that can conduct data processing and storage on the remote cloud server, and at the same time, it allows any device connected to the Internet to access data [13]. Cloud computing is not only widely used in computer terminals, but also can be applied to mobile terminals. For example, in the study [14], the authors used a GPU acceleration method to greatly improve the data processing capability on mobile devices, and in [15] proposed a Web API recommendation method for mobile APP development based on correlation graphs.

In a cloud computing environment, users' data and computing are migrated to an external, virtualized "cloud server" [16], and the cloud server performs data operations and storage. Although this method can reduce the maintenance of computer hardware and software [17], it also brings many privacy security challenges [18]. For example, the load of big data unloading on the network and the leakage of private data and other security issues [19]. In the study of [20], the author designs a deep reinforcement learning (DRL)-based dynamic task offloading(DDTO) algorithm to solve this problem. The DDTO algorithm can not only adapt to the dynamic complex environment, but also can maximize the task completion under the allowed delay. In the aspect of data security privacy protection, secure multi-party computation based on homomorphic encryption has become more and more mainstream to protect data privacy [21]. Mostapha Derfouf et al. proposed a security protocol based on partial homomorphic encryption [22], and compared the protocol with the fully homomorphic encryption protocol, and analyzed the advantages and disadvantages of the two schemes. Effectively solving the security challenges in cloud computing is the foundation of establishing the Industrial Internet of Things(IIoT) [23].

Secure multi-party computation can effectively solve the security and privacy problems in cloud computing [24]. In 1982, Yao [25] proposed the millionaire problem, which officially launched the study of secure multi-party computation, in which several participants with secret inputs work together to compute a function and get their own output, but none of the participants get any information about the inputs [26]. Secure multi-party computation can be achieved by garbled circuits, secret sharing, and homomorphic encryption [27]. This paper mainly improves the security comparison protocol proposed by Damgård, Geisler and KrØigaard(DGK) [28], The main idea is to use the additive homomorphism of homomorphic encryption algorithm to calculate each bit, so as to compare the size of input from both sides without compromising privacy.

There is also research on federated learning to protect data privacy in transit. In [29], the authors propose a federated learning (FL) approach for geographic POI recommendation, which protects the privacy of user data communication while providing better POI recommendation. Blockchain is decentralized and immutable, so many studies use blockchain to protect privacy [30]. However, whether it is public chain or private chain, so there is not only the risk of data leakage, but also the data is easy to be maliciously attacked and participate in the evil of nodes [31] [32]. In the study of [33], the authors mainly introduced the advantages and differences of edge computing compared with cloud computing, and analyzed the data security problems therein. In the study of [34], the author designs an access control and computational resource allocation algorithm to obtain the optimal solution and achieve the maximum utility of resource allocation.

Secure multi-party computation based on Full-Homomorphic

In 2009, Gentry [35] first proposed a completely homomorphic encryption scheme, which can carry out arbitrary operations in the form of ciphertext. However, the key storage and generation of fully homomorphic schemes are too heavy on the network, which makes them impractical. In the study of [36], the dynamic unloading algorithm proposed by the author can effectively improve the user utility function and realize the optimal unloading of dynamic data. In the study of [37], Han et al. introduced the multi-key full-homomorphic encryption scheme based on LWE encryption scheme and the attribute-based multi-key full-homomorphic encryption scheme, which can perform secure multi-party computation in an insecure cloud environment. In the study [38], the authors discussed the advantages and disadvantages of different FHE schemes in detail, and finally gave suggestions on the prospect of FHE application in cloud computing.

Secure multi-party computation based on Partially Homomorphic Encryption

In 2018, Zainab Hikmat Mahmood et al. [39] constructed a hybrid homomorphic encryption scheme using the additive homomorphic GM algorithm and the multiplicative homomorphic RSA algorithm, which not only improved the security, but also optimized the computational overhead of the algorithm, but still needs to trust the interaction between clients. In the study of [40], the authors use Wireless Body Area Network (WBAN) to authenticate nodes, which improves the reliability of data. This authentication method can reduce the storage cost and resource loss in the process of data transmission. In the study of [41], The authors propose a secure approach for data sharing between different domains based on edge computing model. The problem of authentication between different domains is solved, which is effective to ensure the trust of nodes. In 2019, Ghanem Sahar M et al. [42] extended homomorphic Encryption Library (HElib) to support Secure Multiparty Computation is detailed. The performance of the scheme is also analyzed. In 2020, Becher Kilian et al. [43] proposed a multi-Party Computation Protocol Based on a novel approach to random index distribution Secure. In 2021, Luo et al. [44] constructed a multi-round SMPC by using information entropy, which ensured the security between participants and was more suitable for cloud computing.

Figure 1 secure multi-party cloud computation

This paper focuses on the security comparison protocol based on partially homomorphic encryption. And uses the concept of secure multi-party cloud computation(SMCC). In this framework(See figure 1), the cloud server performs some calculations on the data and produces an output. In most of the research on multi-party secure computing, collusion or evil among the participants will affect the correct execution of the protocol. Therefore, the scheme in this paper supervises the participants by verifying the zero-knowledge proof and correctly implements the comparison protocol, without disclosing any privacy information of input data during the protocol process, and is expected to be applied to practical practical applications [45]. The main work of this paper is as follows:

Our contribution

- This scheme uses non-interactive zero-knowledge proof to improve the DGK comparison protocol. By means of the cloud server verifying the zero-knowledge proof, a security comparison protocol that participants cannot do evil is constructed without affecting the security of the original protocol. The improved protocol can not only perform ciphertext comparison calculation safely, but also be applicable to malicious models of participants.
- This scheme uses the additive homomorphism of homomorphic encryption algorithm to split the xOR calculation step of the original protocol in four rounds, so that the server can carry out the calculation stage independently. Compared with the original protocol, the improved protocol can prevent the parties from breaking the protocol by communicating with each other during the computing process, and finally it is more suitable for cloud computing scenarios.
- In the scenario of cloud computing, since servers perform computing tasks, there may be a problem of privacy data disclosure by servers. In this scheme, the server obtains the key bits of the ciphertext through the OT technology. This process does not reveal the privacy of the plaintext input of any client, which not only protects the privacy of the input of participants, but also improves the fairness of the protocol.

Preliminaries

AH-ElGamal Encryption Algorithm

AH-ElGamal [46] is a variant design of ElGamal [47] encryption algorithm. It has the property of addition homomorphism, that is, given the ciphertext $E(m_1, r_1)$ and $E(m_2, r_2)$ under the same public key, the private key holder can obtain the ciphertext sum of m_1 and m_2, through homomorphism calculation, that is, $E(m_1, r_1)E(m_2, r_2) = E(m_1 + m_2, r_1 + r_2)$. The algorithm consists of key generation, encryption and decryption.

Key Generation Select two large prime numbers a and b randomly, let $n = ab$, select two generators g and h in Z_n^*, select number x in Z_n randomly, set the public key to (n, g, h, y), calculate $y = h^x mod n$ and set the private key to x.

Encryption Encrypts the message m, selects the number r randomly, obtains the ciphertext $C = E(m) = (c_1, c_2) = (g^m y^r, h^r)$.

Decryption $g^m = c_1(c_2^x)^{-1}$ is first computed and then searched according to the pre-computed index table with respect to g to obtain plaintext m.

Zero-Knowledge Proof

Zero-knowledge proof (ZKP) was first proposed by Golfwasser et al. [48] in 1985. It means that the prover proves to the verifier that an event is true without giving away any relevant information. Zero knowledge proof system has three characteristics: completeness, reliability and zero knowledge. Completeness means that the verifier always accepts a proposition when it is indeed true. Reliability refers to the probability that the proposition will be accepted by any fraudulent prover when the proposition is not standing. Zero knowledge means that the prover does not learn anything new about the proposition from the proof.

Zero knowledge proof is divided into interactive and non-interactive. Interactive zero-knowledge proof means that the prover needs to challenge the prover constantly to verify the "knowledge" of the prover without disclosing the information of the "knowledge". Non-interactive does not require repeated challenges, but rather uses random oracle to challenge. Zero-knowledge proofs in this paper replace all challenges with hash values through fiat-Shamir [49] heuristic algorithm, which can be used to transform interactive zero-knowledge proofs into non-interactive zero-knowledge proofs.

Oblivious Transfer

Oblivious Transfer (OT) is a cryptographic protocol first proposed by Rabin [50] in 1981. In this protocol, the message sender sends a message from the database to the receiver through the Oblivious Transfer technology. However, in the whole transmission process, the sender does not know which message is sent to the receiver, which ensures anonymity in the information query process. Naor and Pinkas proposed a reusable 1-out-of-N OT protocol based on the Diffie-Hellman hypothesis, which was also the first n-choose 1 inadvertence transport protocol. The following uses 1-out-of-2 OT protocol as an example, The Alice inputs a string (X_0, X_1), and the Bob gets the output X_r by inputting a bit r, (in this process, the Alice cannot determine whether the Bob's input is equal to 0 or 1, while the Bob inputs a bit r, and can only get X_r, not X_{1-r}).

DGK Comparison Protocol

DGK comparison protocol is an efficient and secure comparison protocol under semi-honest model proposed by Damgard et al. This protocol uses the additive homomorphism of homomorphic encryption to efficiently solve the millionaire problem. DGK protocol starts xOR calculation from the high level until xOR value is 1. The comparison result of this key bit can determine the size of two binary numbers. Suppose two binary numbers are represented by m_a and m_b respectively, and $1 < i \leq l$, where l is the length of the binary number and i is the ordinal number of the binary

Algorithm 1 1-out-of-2 OT protocol

```
Input: $(X_0, X_1)$, $r$
Output: $X_r$
1: Prover: Select random number $a$, $C \leftarrow Z_q$. To calculate the $g^a, C^a$.
2: Verifier: random number $k$, $PK_r = g^k, PK_{1-r} = \frac{C}{g^k}$, Send $PK_0$ to the prover.
3: Prover: Calculate the $PK_0^a$, $PK_1^a = \frac{C^a}{PK_0^a}$.
   Perform encryption
   $E_0 = (g^a, H((PK_0^a), 0) \bigoplus X_0)$
   $E_1 = (g^a, H((PK_1^a), 1) \bigoplus X_1)$
   Send $E_0$ and $E_1$ to the verifier.
4: Verifier: Calculate the $H((PK_r)^a, r) \leftarrow H((g^a)^k, r)$.
   $X_r = E_r \bigoplus H((PK_r)^a, r)$.
5: end
```

number. Under the explicit text, the following formula is calculated to obtain the comparison result, if $c_i = 0$ then prove that $m_a < m_b$.

$$c_i = 1 + m_{a,i} - m_{b,i} + 3 \sum_{j=i+1}^{l-1} (m_{a,i} \bigoplus m_{b,i}) \tag{1}$$

If the comparison is made in ciphertext, the final comparison result can be obtained by calculating the following formula. For example, $[m_{a,i}]$ indicates the ciphertext that uses the public key of the encryption algorithm to encrypt $m_{a,i}$.

$$\begin{aligned}[c_i] &= [s + m_{a,i} - m_{b,i} + 3 \sum_{j=i+1}^{l-1} (m_{a,i} \bigoplus m_{b,i})] \\ &= [s] \cdot [m_{a,i}] \cdot [m_{b,i}]^{-1} \cdot (\prod_{j=i+1}^{l-1} [m_{a,j} \bigoplus m_{b,j}])^3\end{aligned} \tag{2}$$

After we get the value of $[c_i]$, in order to prevent the disclosure of secrets, we also need to blind the value of $[c_i]$. Finally, decrypt it. If the decryption result is 0, prove $m_1 < m_2$.

Improved DGK Comparison Protocol Based on Zero-Knowledge Proof (ZKP-DGK)

In this scheme, the behavior of clients is constrained by zero-knowledge proof, and the comparison computing task in the original DGK protocol is handed over to the cloud service provider. As long as the cloud service provider is semi-honest, it will not disclose any privacy, and can correctly conduct secure multi-party computation.

Initialization phase

The public-private key pair generation mechanism of the participating subjects uses the key generation algorithm in the AH-ElGamal encryption algorithm. After the initialization setting to generate data, suppose there are clients A, B and the server C. The server randomly selects two large prime numbers a and b, let $n = ab$, and chooses two generating elements g and h from Z_n^*. Each client randomly selects the number x in Z_n as its private key and satisfies y as the public key, and the public-private key pairs of each subject are denoted as (y_1, x_1), (y_2, x_2) and (y_3, x_3), the encryption form is $C = E(m) = (c_1, c_2) = (g^m y^r, h^r)$, the decryption form is $g^m = c_1(c_2^x)^{-1}$, and m can be decrypted according to the exponential table. The protocol scheme is broadly carried out in four phases, and each client needs to submit the corresponding zero-knowledge proofs in each phase.

Exponential multiplication phase

Clients A and B use server C's public key to encrypt their plaintext information $m_{a,i}$ and $m_{b,i}$ respectively, and the encryption format is AH-ElGmal encryption algorithm mentioned above. Similarly, the encrypted ciphertext of client A is $C_{a,i} = E(m_{a,i}) = (c_{1,i}, c_{2,i}) = (g^{m_{a,i}} y_3^{r_{a,i}}, h^{r_{a,i}})$, and the encrypted ciphertext of client B is $C_{b,i} = E(m_{b,i}) = (c'_{1,i}, c'_{2,i}) = (g^{m_{b,i}} y_3^{r_{b,i}}, h^{r_{b,i}})$. $r_{a,i}$ and $r_{b,i}$ are random numbers selected by clients A and B during the encryption, which are visible only to clients. $m_{a,i} and m_{b,i}$ are the comparative plaintext of A and B. $m_{a,i}$ is composed of l bits of binary numbers, where i represents the number of bits in binary. For example, $m_{a,i}$ represents the ith bit of m_a, where $0 < i \leq l$.

Figure 2 Exponential multiplication phase

- Clients A and B each encrypt each bit to obtain $\{C_{a,l}, C_{a,l-1}...C_{a,1}\}$and $\{C_{b,l}, C_{b,l-1}...C_{b,1}\}$. Client B sends all ciphertexts to A for encryption.
- After client A gets the ciphertext of B, client A uses its plaintext value $m_{a,i}$ to perform exponential operation on the ciphertext of client B and adds a random number r_0 to hide the plaintext of $m_{a,i}$. In order to ensure the decryptibility of the encryption algorithm, the second part of the ciphertext is processed in the same way, and finally the ciphertext $C_{a*b} = (c^*_{1,i}, c^*_{2,i}) = (g^{m_{a,i} m_{b,i}} y_3^{m_{a,i} r_{b,i} + r_0}, h^{m_{a,i} r_b + r_0})$ is obtained, and the obtained ciphertext is sent to server C, and the discrete logarithm encryption proof is submitted.
- The server C can get the value of $m_{a,i} m_{b,i}$ after decrypting the ciphertext. Then, find the serial number $m_{a,i} m_{b,i} = 0$ in the result and mark it as k_n.

Additive homomorphism phase

Figure 3 Additive homomorphism phase

- Client A multiplies B's ciphertext k_n bit with his own ciphertext to get $C_{a,i} C_{b,i} = (c_{1,i} \times c'_{1,i}, c_{2,i} \times c'_{2,i}) = (g^{m_{a,i} + m_{b,i}} y_3^{r_{a,i} + r_{b,i}}, h^{r_{a,i} + r_{b,i}})$. and sends it to server C.
- The server C decrypts each of the ciphertexts. Get $m_{1,i} + m_{2,i}$ from $g^{m_{a,i} + m_{b,i}} = c_{1,i} \times c'_{1,i} (c_{2,i} \times c'_{2,i})^{-x_3} = g^{m_{a,i} + m_{b,i}} y_3^{r_{ai} + r_{b,i}} (h^{r_{a,i} + r_{b,i}})^{-x_3}$
- The server C calculates xOR value $c_\oplus$ by the following formula:

$$c_\oplus = m_{1,i} + m_{2,i} - 2 m_{a,i} m_{b,i} \tag{3}$$

- The server C finds the first sequence number i from $c_\oplus$ that is 0, denoted as the key comparison bit t.

Figure 4 The OT phase

The OT phase

- Before this phase, clients A and B need to combine their public keys to generate the joint public key $y' = y_1 * y_2$. The joint public key here adopts the idea of threshold decryption for the final result to be decrypted and verified jointly by A and B, so as to avoid the evil of one party.
- Clients A and B use the joint public key y' to generate ciphertext $C'_{a,i} = E'(m_{a,i}) = (c_{a,i}, c'_{a,i}) = (g^{m_{a,i}} y'^{r_{a,i}}, h^{r_{a,i}})$ and $C'_{b,i} = E'(m_{b,i}) = (c_{b,i}, c'_{b,i}) = (g^{m_{b,i}} y'^{r_{b,i}}, h^{r_{b,i}})$, respectively.
- 1-out-of-N OT: The server C enters the number t. Clients A and B enter all $C'_{a,i}$ and $C'_{b,i}$, which are calculated by the OT protocol. The server C finally gets the ciphertext of $C'_{a,t}$ and $C'_{b,t}$ two key bits.
- The server C calculates C^* to get the final comparison result:

$$c_1^* = g \times c_{a,t} \times c_{b,t}^{-1} = g^{1+m_{a,t}-m_{b,t}} y'^{r_{a,t} r_{b,t}} \tag{4}$$

$$c_2^* = c'_{a,t} \times c'^{-1}_{b,t} = h^{r_{a,t}-r_{b,t}} \tag{5}$$

Send $C^* = (c_1^*, c_2^*)$ to clients A and B.

Verification phase

Figure 5 Verification phase

- After obtaining the ciphertext comparison result C^*, clients A and B jointly decrypt the result to obtain $g^{1+m_{a,t}-m_{b,t}}$. If $g^{1+m_{a,t}-m_{b,t}}$ is 1 then $m_a < m_b$ is proved.

Zero knowledge proof process

Binary format proof

The zero-knowledge proof in this paper uses Fiat-Shamir heuristic algorithm to replace all challenges with hash values, and this method can be used to transform interactive zero-knowledge proof into non-interactive zero-knowledge proof. Algorithm 1 is described as follows: There is ciphertex $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{ra,i}, h^{r_{a,i}})$. The prover proves to the verifier that $m_{a,i}$ in c_1 is the correct binary format [6].

Proof of ciphertext format correctness

Algorithms 2 and 3 are described as follows: The participants need to prove that and in $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{ra,i}, h^{r_{a,i}})$ are encrypted in accordance with the AH-ElGamal encryption form. The proof is divided into two parts: representation proof and discrete logarithm proof.

Algorithm 2 Binary Format Proof

```
Input: $Proof_{1,0}^{C_{a,i}}(c_1, m_{a,i}, r_{r,i})$, random numbers $\lambda_1$,$\lambda_2$,$\lambda_3$, Challenge $C$ , total number $l$
Output: $Verify_{1,0}^{C_{a,i}}(Proof_{1,0}^{C_{a,i}}, c_1)$
 1: Prover:
 2: for $i = 1, i \leqslant l$ do
 3:     $t_1 = g^{\lambda_1} y^{\lambda_2}$
 4:     $t_2 = g^{\lambda_1 m_{a,i}} y^{\lambda_3}$
 5:   $H : \{0,1\}^* \to Z_p^*$ is a secure one-way hash function.The Prover computes Challenge $C$ through
    a random oracle: $C = H(g, y, t_1, t_2)$
 6:     $s_1 = m_{a,i}C + \lambda_1$
 7:     $s_2 = r_{a,i}C + \lambda_2$
 8:     $s_3 = r_{a,i}(C - s_1) + \lambda_3$
 9: endfor
 1: Verifier:
 2: if $c_1^C t_1 = g^{s_1} y^{s_2}, c_1^{C-s_1} t_2 = g^0 y^{s_3}$ then
      $Verify_{1,0}^{C_{a,i}} = 1$
 3: else
      $Verify_{1,0}^{C_{a,i}} = 0$
```

Algorithm 3 Proof of Part c_1

```
Input: $Proof^{c_1}(c_1, m_{a,i}, r_{r,i})$, random numbers $\beta_1$,$\beta_2$, Challenge $C$ , total number $l$
Output: $Verify^{c_1}(Proof^{c_1}, c_1)$
 1: Prover:
 2: for $i = 1, i \leqslant l$ do
 3:     $t = g^{\beta_1} y^{\beta_2}$
 4:   $H : \{0,1\}^* \to Z_p^*$ is a secure one-way hash function.The Prover computes Challenge $C$ through
    a random oracle: $C = H(g, y, c_1, t)$
 5:     $s_1 = m_{a,i}C + \beta_1$
 6:     $s_2 = r_{a,i}C + \beta_2$
 7: endfor
 1: Verifier:
 2: if $t \times c_1^C = g^{s_1} \times y^{s_2}$ then
      $Verify^{c_1} = 1$
 3: else
      $Verify^{c_1} = 0$
```

- Part c_1: The ciphertext is $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{ra,i}, h^{r_{a,i}}), 1 < i \leq l$. Prover knows that $witness(m_{a,i}, r_{a,i})$ and proves that $c_1 = g^{m_{a,i}} y^{ra,i}$ holds.
- Part c_2: The ciphertext is $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{ra,i}, h^{r_{a,i}}), 1 < i \leq l$. Prover knows that $witness(r_{a,i})$ and proves that $c_2 = h^{r_{a,i}}$ holds.

Algorithm 4 Proof of Part c_2

```
Input: $Proof^{c_2}(c_2, r_{r,i})$, random numbers $\beta_3$, Challenge $C$ , total number $l$
Output: $Verify^{c_2}(Proof^{c_2}, c_2)$
 1: Prover:
 2: for $i = 1, i \leqslant l$ do
 3:     $t = h^{\beta_3}$
 4:   $H : \{0,1\}^* \to Z_p^*$ is a secure one-way hash function.The Prover computes Challenge $C$ through
    a random oracle: $C = H(h, c_2, t)$
 5:     $s = \beta_3 - Cr_{a,i}$
 6: endfor
 1: Verifier:
 2: if $t \times c_2^C \times h_s = t$ then
      $Verify^{c_2} = 1$
 3: else
      $Verify^{c_2} = 0$
```

Random Number Consistency Proof

The encryption format used in this scheme is $(c_1, c_2) = (g^{m_{a,i}} y^{r_{a,i}}, h^{r_{a,i}})$. Since there are two ciphertexts c_1 and c_2, c_2 is used for decryption. If c_1 and c_2 are inconsistent during the encryption, the decryption operation cannot be carried out normally. Therefore, it is necessary to prove that $r_{a,i}$ of c_1 and c_2 are consistent.

Algorithm 5 is described as follows: There is ciphertext $C_{a,i} = (C_{a,l}, C_{a,l-1}, C_{a,l-2}, C_{a,l-3}...C_{a,1})$, where $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{r_{a,i}}, h^{r_{b,i}})$, $0 < i \leq l$. The Prover knows witness $(m_{a,i}, r_{a,i})$ and shows that $r_{a,i} = r_{b,i}$ in $(g^{m_{a,i}} y^{r_{a,i}}, h^{r_{b,i}})$.

Algorithm 5 Random Number Consistency Proof

```
Input: $Proof_r^{C_{a,i}}(c_1, m_{a,i}, r_{r,i})$, random numbers $\gamma_1$,$\gamma_2$,$\gamma_3$, Challenge $C$ , total number $l$
Output: $Verify_r^{C_{a,i}}(Proof_r^{C_{a,i}}, c_1, c_2)$
1: Prover:
2: for $i = 1, i \leqslant l$ do
3:     $t = g^{\gamma_1} y^{\gamma_2} h^{\gamma_3}$
4:   $H : \{0,1\}^* \rightarrow Z_p^*$ is a secure one-way hash function.The Prover computes Challenge $C$ through
   a random oracle: $C = H(g, y, t_1, t_2)$
5:     $s_1 = \gamma_1 - m_{a,i}C$
6:     $s_2 = \gamma_2 - r_{a,i}C$
7:     $s_3 = \gamma_3 - r_{b,i}C$
8: endfor
1: Verifier:
2:     $z = c_1 c_2 = g^{m_{a,i}} y^{r_{a,i}} h^{r_{b,i}}$
3: if $g_1^s y_2^s h_3^s z^C = t$ then
     $Verify_r^{C_{a,i}} = 1$
4: else
     $Verify_r^{C_{a,i}} = 0$
```

Ciphertext Consistency Proof

Since this scheme uses multiple rounds of comparison, it is necessary to ensure that the plaintext information contained in the ciphertext submitted by participants in each round is consistent.

Algorithm 6 is described as follows:The encryption format is $C_{a,i} = (C_{a,l}, C_{a,l-1}, C_{a,l-2}, C_{a,l-3}...C_{a,1})$, where $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}} y^{r_{a,i}}, h^{r_{b,i}})$, $0 < i \leq l$. The ciphertext $c_1 = g^{m_{a,i}} y^{r_{a,i}}$ in the first round and $c_1' = g^{m'_{a,i}} y^{r_{a,i}}$ in the second round are set. We need to prove that $m_{a,i}$ is consistent with $m'_{a,i}$.

Algorithm 6 Ciphertext Consistency Proof

```
Input: $Proof^{m_{a,i}}(C_{a,i}, c'_{a,i})$, total number $l$
Output: $Verify^{m_{a,i}}(Proof^{m_{a,i}}, c_1, c_2)$
1: Prover:
2: for $i = 1, i \leqslant l$ do
3:     $z = c_1 \times c_1'^{-1} = g^{m_{a,i} - m'_{a,i}} y_3^{r_{a,i} - r'_{a,i}}$
4: endfor
1: Vierfier:
2: if $z = y_3^{r_{a,i} - r'_{a,i}}$ then
     $Verify_r^{C_{a,i}} = 1$
3: else
     $Verify_r^{C_{a,i}} = 0$
```

Security and Fairness analysis

Security analysis

Since the objective of this scheme is to apply under the malicious participant model, a security analysis of the zero-knowledge proofs submitted in the scheme is required.

Binary format proof

Suppose there exists a malicious adversary A' that is able to construct a new challenge C' by imitating the challenge C of the random oracle and constructs a response s_1, s_2, s_3, while submitting a false proof $Proof_{1,0}^{C_{a,i}}(C_{a,i}, m_{a,i}, r_{r,i})$. Then A can get $c_1^{C-C'} = g^{s_1-s'_1}y^{s_2-s'_2}$ and $c_1^{C-s_1+C'-s'_1} = g^0y^{s_3-s'_3}$ by combining equations, and then the $m_{a,i}$ and $r_{a,i}$ values of the ciphertext can be obtained by equations $(s_1 - s'_1)(C_1 - C'_1)^{-1}1$ and $(s_2 - s'_2)(C_1 - C'_1)^{-1}$, so the plaintext is completely compromised. However, in the case that the security parameter is large enough, the malicious adversary A' has to guess the same C' from the random prediction machine under the satisfied condition, which is considered infeasible, and after guessing C' and to get the value of t_1 at the same time, but the value of t_1 needs to be found by the discrete logarithm values of λ_1 and λ_2. The difficulty is to solve the discrete logarithm problem on G_p in polynomial time, which is also considered infeasible, so the assumption is not valid.

Proof of ciphertext format correctness

Since the ciphertext encryption format of the participants takes AH-ElGamal encryption algorithm, the protocol participants need to ensure that the ciphertext is encrypted in the form of $C_{a,i} = E(m_{a,i}) = (c_1, c_2) = (g^{m_{a,i}}y^{ra,i}, h^{r_{a,i}})$. Therefore, this scheme requires the participants to submit two proofs of and respectively. The following is an example of part c1.

- **Part** c_1: Suppose that the malicious client A' can construct a new challenge C' by imitating the challenge C of the random oracle. then the value of the ciphertext $m_{a,i}$ can be obtained by combining the equation $(s_1 - s'_1)(C_1 - C'_1)^{-1}$, and then the value of the random number $r_{a,i}$ can be obtained by $(s_2 - s'_2)(C_1 - C'_1)^{-1}$. Thus, the ciphertext c_1 is breached, resulting in the leakage of the plaintext. In the same way as the binary format proof analysis, the cheating client A' has to guess the same C' from the random prophecy machine under the satisfied condition, which is considered infeasible with sufficiently large security parameters.
- **Part** c_2: It is assumed that the cheating participant A' can forge an identical response s', and then the r value of the ciphertext can be obtained by combining the equation $\frac{s-s'}{C-C'}$. In this way, the ciphertext c_2 is broken, leading to the disclosure of the random number $r_{a,i}$. In the same way as the above analysis, cheating participant A' would have to guess the same C' from the random predictor if the conditions were satisfied, which would be considered infeasible if the safety parameters were large enough.

The analysis proves that it is not feasible for the malicious adversary A' to forge a false proof in all stages of the scheme. And neither client A nor B under the malicious model is involved in the comparison calculation, so as long as the referee follows the protocol correctly, then clients A and B under the malicious model will not affect the security of the scheme, and the security of the remaining phases can be referred to the security proof of the DGK protocol.

Exponential multiplication phase

In the exponential multiplication phase of the ZKP-DGK scheme, client A needs to exponentiate the ciphertext of client B with its own plaintext, and then add a random number r_0 to construct the ciphertext form of $C_{b,i}^{m_{a,i}} y_3^{r_0} = g^{m_{a,i} m_{b,i}} y_3^{m_{a,i} r_{b,i} + r_0}$. This proof can be analyzed by the discrete logarithm proof, and by the same token, it is infeasible for the client A to try to forge a false proof. And in additive homomorphism phase, since the ciphertext is bound using the Pedersen commitment, it is also secure enough for A and B.

In the third stage, the clients A and B need to encrypt with the joint public key y' and submit a ciphertext correctness proof, similar to the first stage. Since the server C in the ZKP-DGK scheme is a semi-honest model, and neither client A nor B under the malicious model is involved in the comparison calculation, as long as the server C follows the protocol correctly, then even if clients A and B are malicious adversaries, the security of the scheme will not be affected, and the security of the remaining phases can be proved by referring to the security of the DGK protocol.

Fairness analysis

Since this scheme assumes that the clients are malicious models and a semi-honest server C is introduced, it is necessary to consider whether there is information leakage in each stage of data interaction and ciphertext computation in this scheme.

Exponential multiplication phase

For malicious clients A and B, they both want to maximize their gains, so this subsection first analyzes whether the clients A and B can access each other's secret input. In the first phase, client A exponentiates the secret message received from B with its own plaintext to get $C_{b,i}^{m_{a,i}}$. To avoid the leakage of A's plaintext and add a random number r_0 to hide $m_{a,i}$ one step deeper to get $g^{m_{a,i} m_{b,i}} y_3^{m_{a,i} r_{b,i} + r_0}$ and submit a proof of ciphertext format correctness. In this process, for client A, it will not know any information about $m_{b,i}$ because A only performs the exponential operation on $C_{b,i}$. For client B, the value of the plaintext $m_{a,i}$ is not queried through the exponential table because of the random number r_0. For the server C, the server C can only decrypt the value of $m_{a,i}$ and $m_{b,i}$ multiplied from $g^{m_{a,i} m_{b,i}} y_3^{m_{a,i} r_{b,i} + r_0}$, but does not get the specific values of $m_{a,i}$ and $m_{b,i}$.

Additive homomorphism phase

At this stage client A needs to multiply its own ciphertext $C_{a,i}$ with client B's ciphertext $C_{b,i}$ to get $C_{a,i} C_{b,i} = (g^{m_{a,i} + m_{b,i}} y_3^{r_{a,i} + r_{b,i}}, h^{r_{a,i} + r_{b,i}})$. This stage is the same as the previous stage of the proof, where clients A and B cannot get any explicit information about $m_{a,i}$ and $m_{b,i}$. For server C, who has information about $m_{a,i} m_{b,i}$ as well as $m_{a,i} + m_{b,i}$ at this point, it is fair for all parties involved since server C does not have the ability to determine the specific explicit value of $m_{a,i}$ or $m_{b,i}$ based on $m_{a,i} m_{b,i}$ and $m_{a,i} + m_{b,i}$.

The OT phase

This phase of oblivious transfer (OT) refers to the 1-out-of-N model in a black box environment. That is, the server C inputs the sequence number t of the key comparison bit, and the corresponding ciphertexts $C'_{a,t}$ and $C'_{b,t}$ are obtained by the OT protocol. in this technique, clients A and B cannot know which ciphertext data server C has obtained, and at the same time server C can obtain other ciphertexts besides $C'_{a,t}$ and $C'_{b,t}$.

Verification phase

At this phase, clients A and B need to decrypt jointly to get the final comparison result, thus ensuring that A and B cannot decrypt privately. Since clients A and B did not participate in the calculation process of the comparison results, and server C needed to verify the submitted zero-knowledge proof, the clients could not do evil. As for the semi-honest server C, as long as the server correctly carries out the agreement, the fairness of the agreement will not be affected.

Analysis shows that, throughout the course of the scheme, neither clients A and B, nor server C, have access to data about private inputs. Therefore, this scheme is fair to the clients. The whole scheme not only protects the user's private data, but also limits the malicious behavior between data parties, which provides the basis for some real-world applications, such as electronic voting, privacy auction, big data fusion, and so on [51]. With the implementation of practical applications, the data will become larger and larger, which will bring a serious burden to the network. A large number of scholars have conducted research on this aspect [52], so the network overhead will be the next step for this solution.

Conclusion

The security comparison protocol based on homomorphic encryption provides positive significance for the development of cloud computing, but the distrust between data owners and the expensive computing overhead of full homomorphism still bring severe challenges to cloud computing. This paper combines zero-knowledge proof and partially homomorphic encryption, improves the DGK comparison protocol, and uses the cloud server in cloud computing to replace the third party, and proposes a security comparison protocol suitable for malicious parties. In this scheme, zero-knowledge proofs are used to curb malicious behavior of clients. Compared with fully homomorphic encryption, partially-homomorphic encryption not only guarantees the privacy of ciphertext, but also has a smaller computational cost. During the whole process, private information will not be leaked to any party, including trusted servers. The final analysis shows that the scheme can achieve the security and fairness of ciphertext comparison. In future work, we will continue to study zero-knowledge proof, homomorphic encryption, edge computing, etc. and finally construct a security comparison protocol under a completely malicious model.

Acknowledgements
We are thankful to State Key Laboratory of Public Big Data of Guizhou University for providing an environment for editing manuscripts and experiments.

Funding
This work is financially supported by the National Natural Science Foundation of China under Grant No. (62202118. 61962009). And in part by Top Technology Talent Project from Guizhou Education Department (Qian jiao ji [2022]073). And in part by Foundation of Guangxi Key Laboratory of Cryptography and Information Security(GCIS202118).

Author Contributions
Yuling constructs the ZKP-DGK protocol in detail under the malicious model. Junhong Tao analyzes the completeness of the scheme and draws the flow chart. Tao Li analyzes the security of the scheme and puts forward some suggestions on the construction of the algorithm. Ren has made useful improvements to the language and structure of the overall article. Yun Luo finally verified the correctness of the scheme.

Ethics approval and consent to participate
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Availability of data and materials
The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Author details
[1]State Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang, China. [2]Blockchain Laboratory of Agricultural Vegetables, Weifang University of Science and Technology, Weifang, China. [3]Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, Guilin, China. [4]POWERCHINA Guizhou Electric Power Engineering Co.,Ltd., Guiyang, Guizhou, China.

References
1. J. Dong, W. Wu, Y. Gao, X. Wang, and P. Si, "Deep reinforcement learning based worker selection for distributed machine learning enhanced edge intelligence in internet of vehicles," *Intelligent and Converged Networks*, vol. 1, no. 3, pp. 234–242, 2020, doi: 10.23919/ICN.2020.0015.
2. K. El Makkaoui, A. Beni-Hssane, and A. Ezzati, "Cloud-elgamal: An efficient homomorphic encryption scheme," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2016, pp. 63–66, doi: 10.1109/WINCOM.2016.7777192.
3. J. Xu, D. Li, W. Gu, and Y. Chen, "Uav-assisted task offloading for iot in smart buildings and environment via deep reinforcement learning," *Building and Environment*, vol. 222, p. 109218, 2022, doi: https://doi.org/10.1016/j.buildenv.2022.109218. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360132322004541
4. T. Li, Y. Chen, Y. Wang, Y. Wang, M. Zhao, H. Zhu, Y. Tian, X. Yu, Y. Yang, and X. Xu, "Rational protocols and attacks in blockchain system," *Sec. and Commun. Netw.*, vol. 2020, jan 2020, doi: 10.1155/2020/8839047. [Online]. Available: https://doi.org/10.1155/2020/8839047
5. L. Qi, C. Hu, X. Zhang, M. R. Khosravi, S. Sharma, S. Pang, and T. Wang, "Privacy-aware data fusion and prediction with spatial-temporal context for smart city industrial environment," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4159–4167, 2021, doi: 10.1109/TII.2020.3012157.
6. J. Groth and M. Kohlweiss, "One-out-of-many proofs: Or how to leak a secret and spend a coin," in *Advances in Cryptology - EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 253–280, doi: 10.1007/978-3-662-46803-6_9.
7. M. Naor and B. Pinkas, "Efficient oblivious transfer protocols," in *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '01. USA: Society for Industrial and Applied Mathematics, 2001, p. 448–457.
8. Y. Zhang, H. Zhang, J. Cosmas, N. Jawad, K. Ali, B. Meunier, A. Kapovits, L.-K. Huang, W. Li, L. Shi, X. Zhang, J. Wang, I. Koffman, M. Robert, and C. C. Zarakovitis, "Internet of radio and light: 5g building network radio and edge architecture," *Intelligent and Converged Networks*, vol. 1, no. 1, pp. 37–57, 2020, doi: 10.23919/ICN.2020.0002.
9. Y. Chen, H. Xing, Z. Ma, and et al., "Cost-efficient edge caching for noma-enabled iot services," *China Communications*, 2022.
10. Y. Chen, X. Yang, T. Li, Y. Ren, and Y. Long, "A blockchain-empowered authentication scheme for worm detection in wireless sensor network," *Digital Communications and Networks*, 2022, doi: 10.1016/j.dcan.2022.04.007.
11. Y. Wang, T. Li, M. Liu, C. Li, and H. Wang, "Stsiiml: Study on token shuffling under incomplete information based on machine learning," *International Journal of Intelligent Systems*, pp. 1–23, 2022, doi: 10.1002/int.23033. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/int.23033
12. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/COMST.2015.2444095.
13. W. Zhang, X. Chen, and J. Jiang, "A multi-objective optimization method of initial virtual machine fault-tolerant placement for star topological data centers of cloud systems," *Tsinghua Science and Technology*, vol. 26, no. 1, pp. 95–111, 2021, doi: 10.26599/TST.2019.9010044.
14. M. Ayad, M. Taher, and A. Salem, "Mobile gpu cloud computing with real time application," in *5th International Conference on Energy Aware Computing Systems & Applications*, 2015, pp. 1–4, doi: 10.1109/ICEAC.2015.7352209.

15. L. Qi, W. Lin, X. Zhang, W. Dou, X. Xu, and J. Chen, "A correlation graph based approach for personalized and compatible web apis recommendation in mobile app development," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2022, doi: 10.1109/TKDE.2022.3168611.
16. Z. Salman and W. M. Elmedany, "A trustworthy cloud environment using homomorphic encryption: a review," in *4th Smart Cities Symposium (SCS 2021)*, vol. 2021, 2021, pp. 31–36, doi: 10.1049/icp.2022.0308.
17. S. M. Ghanem and I. A. Moursy, "Secure multiparty computation via homomorphic encryption library," in *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2019, pp. 227–232, doi: 10.1109/ICICIS46948.2019.9014698.
18. A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," *Big Data Mining and Analytics*, vol. 5, no. 1, pp. 32–40, 2022, doi: 10.26599/BDMA.2021.9020016.
19. Y. Chen, J. Sun, Y. Yang, T. Li, X. Niu, and H. Zhou, "Psspr: a source location privacy protection scheme based on sector phantom routing in wsns," *International Journal of Intelligent Systems*, vol. 37, no. 2, pp. 1204–1221, 2022.
20. Y. Chen, W. Gu, and K. Li, "Dynamic task offloading for internet of things in mobile edge computing via deep reinforcement learning," *International Journal of Communication Systems*, doi: 10.1002/dac.5154.
21. A. chouhan, A. kumari, and M. Saiyad, "Secure multiparty computation and privacy preserving scheme using homomorphic elliptic curve cryptography," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, 2019, pp. 776–780, doi: 10.1109/ICCS45141.2019.9065645.
22. M. Derfouf and M. Eleuldj, "Cloud secured protocol based on partial homomorphic encryptions," in *2018 4th International Conference on Cloud Computing Technologies and Applications (Cloudtech)*, 2018, pp. 1–6, doi: 10.1109/CloudTech.2018.8713353.
23. Y. Chen, Z. Liu, Y. Zhang, Y. Wu, X. Chen, and L. Zhao, "Deep reinforcement learning-based dynamic resource management for mobile edge computing in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4925–4934, 2021, doi: 10.1109/TII.2020.3028963.
24. T. Oladunni and S. Sharma, "Homomorphic encryption and data security in the cloud," in *Proceedings of 28th International Conference on Software Engineering and Data Engineering*, ser. EPiC Series in Computing, F. Harris, S. Dascalu, S. Sharma, and R. Wu, Eds., vol. 64. EasyChair, 2019, pp. 129–138, doi: 10.29007/drnc. [Online]. Available: https://easychair.org/publications/paper/HWfT
25. A. C. Yao, "Protocols for secure computations," in *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, 1982, pp. 160–164, doi: 10.1109/SFCS.1982.38.
26. Y. Chen, S. Dong, T. Li, Y. Wang, and H. Zhou, "Dynamic multi-key fhe in asymmetric key setting from lwe," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 5239–5249, 2021, doi: 10.1109/TIFS.2021.3127023.
27. K. Patel, "Secure multiparty computation using secret sharing," in *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, 2016, pp. 863–866, doi: 10.1109/SCOPES.2016.7955564.
28. I. Damgard, M. Geisler, and M. Kroigaard, "Homomorphic encryption and secure comparison," *International Journal of Applied Cryptography*, vol. 1, no. 1, pp. 22–31, 2008, doi: 10.1504/IJACT.2008.017048.
29. J. Huang, Z. Tong, and Z. Feng, "Geographical poi recommendation for internet of things: A federated learning approach using matrix factorization," *International Journal of Communication Systems*, doi: 10.1002/dac.5161.
30. K. Lu and C. Zhang, "Blockchain-based multiparty computation system," in *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 28–31, doi: 10.1109/ICSESS49938.2020.9237698.
31. T. Li, Z. Wang, G. Yang, Y. Cui, Y. Chen, and X. Yu, "Semi-selfish mining based on hidden markov decision process," *International Journal of Intelligent Systems*, vol. 36, no. 7, pp. 3596–3612, 2021, doi: 10.1002/int.22428.
32. T. Li, Z. Wang, Y. Chen, C. Li, Y. Jia, and Y. Yang, "Is semi-selfish mining available without being detected?" *International Journal of Intelligent Systems*, doi: https://doi.org/10.1002/int.22656. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/int.22656
33. M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, "Edge computing perspectives: Architectures, technologies, and open security issues," in *2019 IEEE International Conference on Edge Computing (EDGE)*, 2019, pp. 116–123, doi: 10.1109/EDGE.2019.00035.
34. J. Huang, B. Lv, Y. Wu, Y. Chen, and X. Shen, "Dynamic admission control and resource allocation for mobile edge computing enabled small cell network," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1964–1973, 2022, doi: 10.1109/TVT.2021.3133696.
35. C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 169–178, doi: 10.1145/1536414.1536440.
36. Y. Chen, F. Zhao, Y. Lu, and X. Chen, "Dynamic task offloading for mobile edge computing with hybrid energy supply," *Tsinghua Science and Technology*, 2021, doi: 10.26599/TST.2021.9010050.
37. J.-L. Han, Z.-L. Wang, Y.-Q. Shi, M.-J. Wang, and H. Dong, "Secure multiparty computation via fully homomorphic encryption scheme," in *2018 Eighth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*, pp. 250–253, doi: 10.1109/IMCCC.2018.00060.
38. S. Mittal, K. Ramkumar, and A. Kaur, "Preserving privacy in clouds using fully homomorphic encryption," in *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, 2021, pp. 1–7, doi: 10.1109/SMARTGENCON51891.2021.9645822.
39. N. Sudhakar Reddy, V. L. Padmalatha, and A. V. L. N. Sujith, "A novel hybrid quantum protocol to enhance secured dual party computation over cloud networks," in *2018 IEEE 8th International Advance Computing Conference (IACC)*, 2018, pp. 142–149, doi: 10.1109/IADCC.2018.8692128.
40. X. Tan, J. Zhang, Y. Zhang, Z. Qin, Y. Ding, and X. Wang, "A puf-based and cloud-assisted lightweight authentication for multi-hop body area network," *Tsinghua Science and Technology*, vol. 26, no. 1, pp. 36–47,

2021, doi: 10.26599/TST.2019.9010048.

41. K. Fan, Q. Pan, J. Wang, T. Liu, H. Li, and Y. Yang, "Cross-domain based data sharing scheme in cooperative edge computing," in *2018 IEEE International Conference on Edge Computing (EDGE)*, 2018, pp. 87–92, doi: 10.1109/EDGE.2018.00019.
42. S. M. Ghanem and I. A. Moursy, "Secure multiparty computation via homomorphic encryption library," in *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2019, pp. 227–232, doi: 10.1109/ICICIS46948.2019.9014698.
43. K. Becher and T. Strufe, "Efficient cloud-based secret shuffling via homomorphic encryption," in *2020 IEEE Symposium on Computers and Communications (ISCC)*, 2020, pp. 1–7, doi: 10.1109/ISCC50000.2020.9219588.
44. Y. Luo, Y. Chen, T. Li, Y. Wang, Y. Yang, and X. Yu., "An entropy-view secure multiparty computation protocol based on semi-honest model," *Journal of Organizational and End User Computing (JOEUC)*, vol. 34, no. 10, pp. 1–17, 2022, doi: http://doi.org/10.4018/JOEUC.306752.
45. F. Yuan, S. Chen, K. Liang, and L. Xu, "Research on the coordination mechanism of traditional chinese medicine medical record data standardization and characteristic protection under big data environment," H. Changqing, Ed. Shandong: Shandong People's Publishing House, 2021.
46. Z. Chen, R. Zhang, Y. Yang, and Z. Li, "A homomorphic elgamal variant based on bgn's method," in *2013 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2013, pp. 1–5, doi: 10.1109/CyberC.2013.10.
47. T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985, doi: 10.1109/TIT.1985.1057074.
48. S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems," in *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, ser. STOC '85. New York, NY, USA: Association for Computing Machinery, 1985, p. 291–304, doi: 10.1145/22145.22178. [Online]. Available: https://doi.org/10.1145/22145.22178
49. M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ser. CCS '93. New York, NY, USA: Association for Computing Machinery, 1993, p. 62–73, doi: 10.1145/168588.168596. [Online]. Available: https://doi.org/10.1145/168588.168596
50. M. O. Rabin, "How to exchange secrets by oblivious transfer," *Technical Memo TR-81:Harvard University: Aiken Computation Laboratory*, 1981.
51. J. Wang, Y. Shen, and B. Wang, "Sealed-bid auction scheme based on blockchain and secure multi-party computation," in *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*, vol. 5, 2021, pp. 407–412, doi: 10.1109/ITNEC52019.2021.9587239.
52. Y. Chen, F. Zhao, X. Chen, and Y. Wu, "Efficient multi-vehicle task offloading for mobile edge computing in 6g networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 4584–4595, 2022, doi: 10.1109/TVT.2021.3133586.

Figures

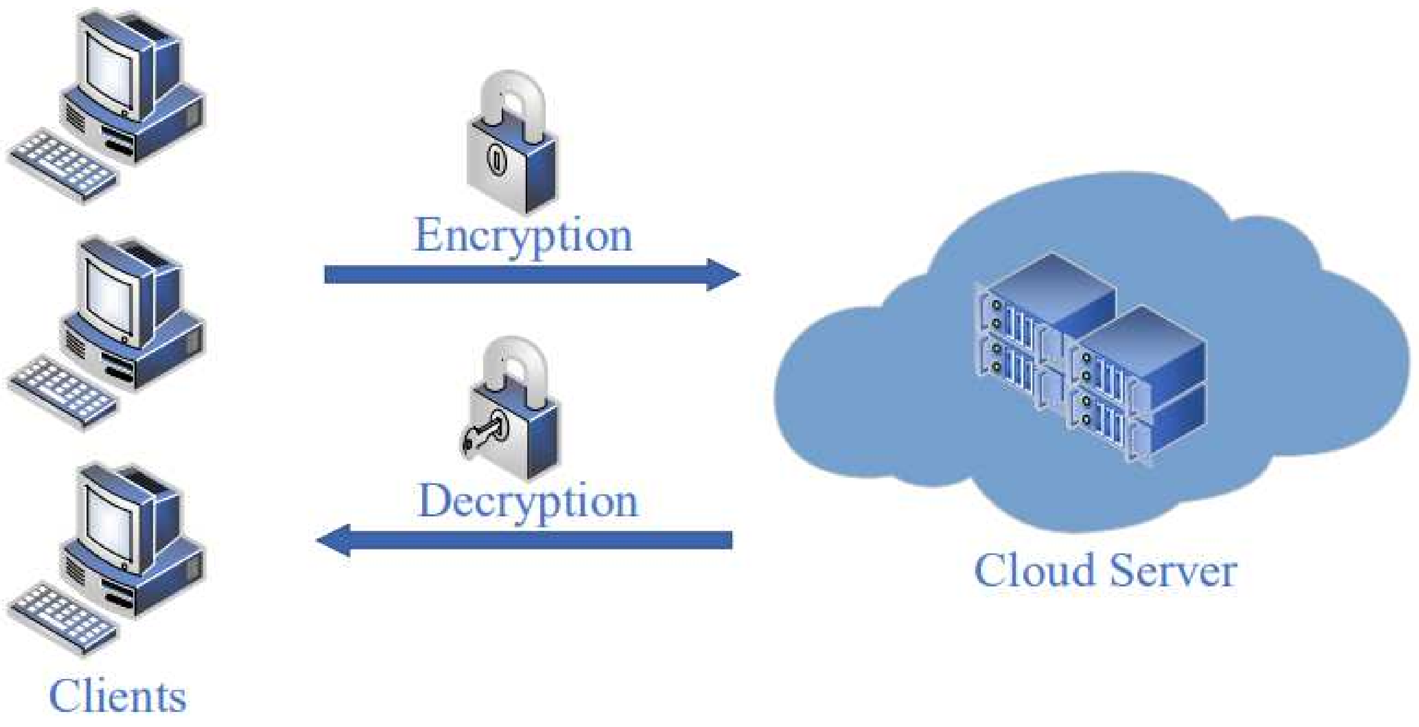


Figure 1

secure multi-party cloud computation

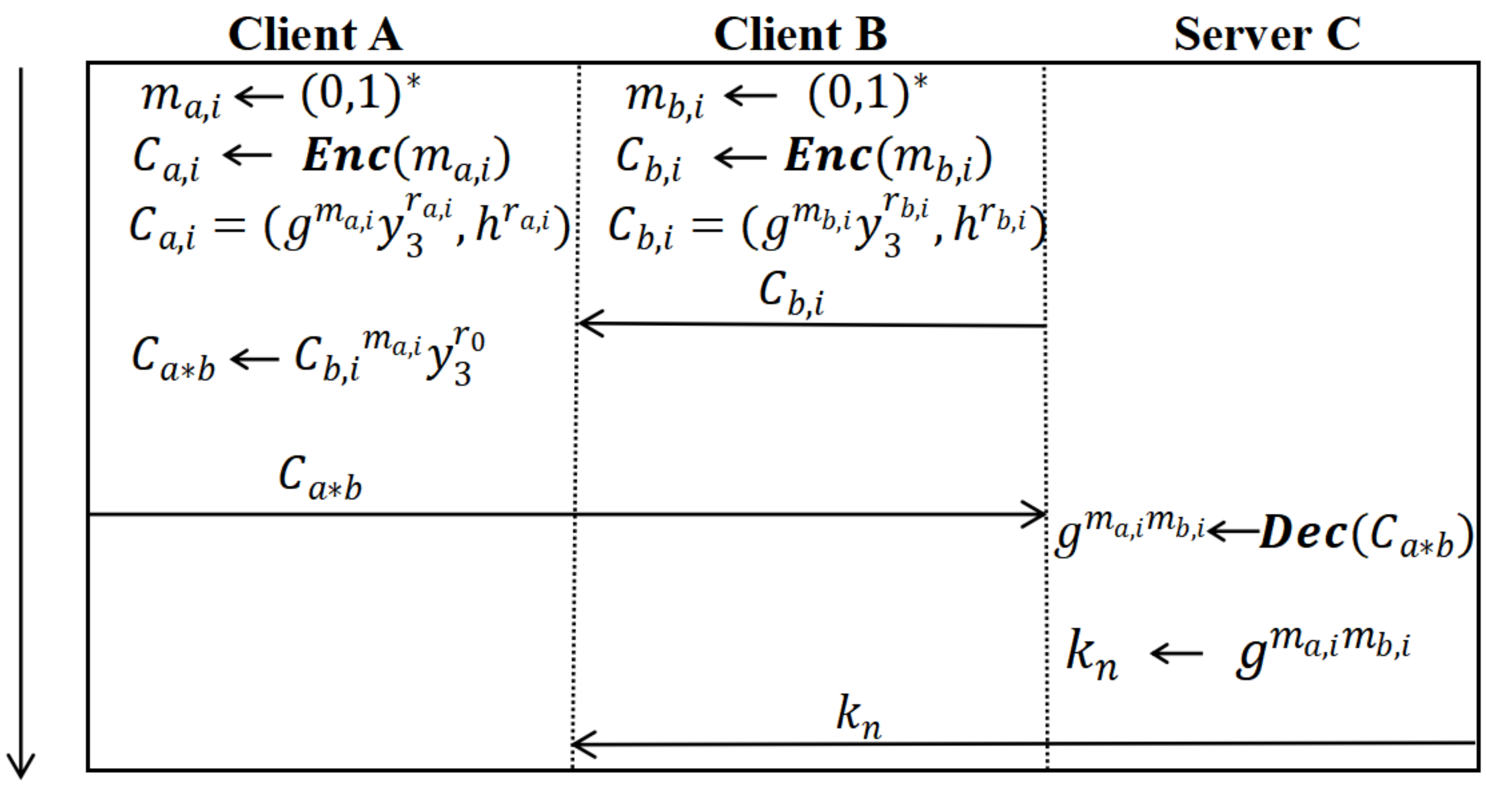
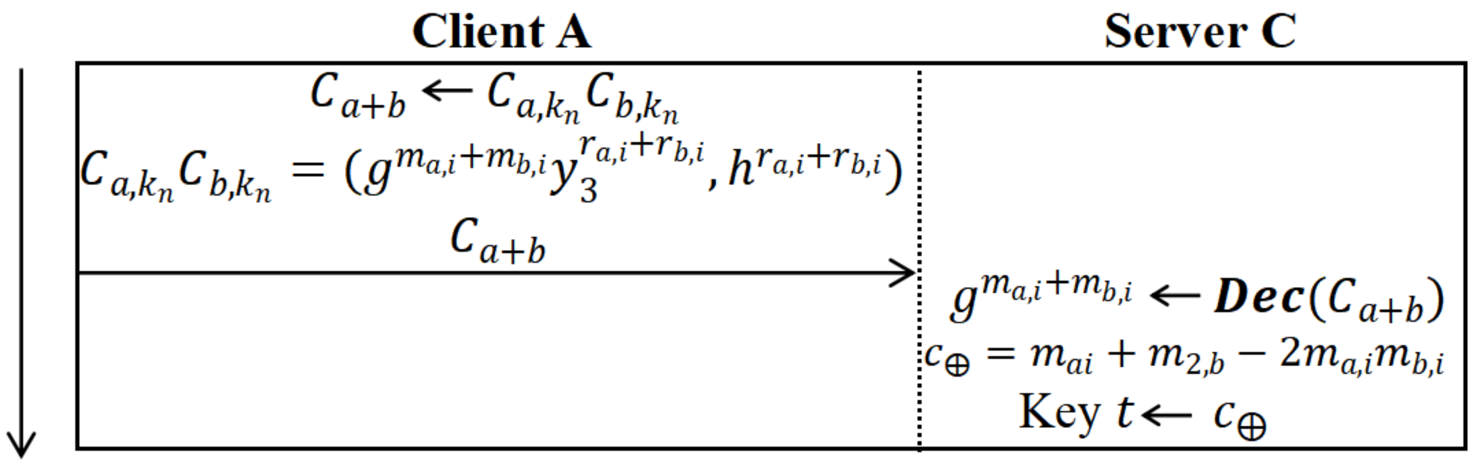

Client A
Client B
Server C
$m_{a,i} \leftarrow (0,1)^*$
$C_{a,i} \leftarrow \boldsymbol{Enc}(m_{a,i})$
$C_{a,i} = (g^{m_{a,i}} y_3^{r_{a,i}}, h^{r_{a,i}})$
$m_{b,i} \leftarrow (0,1)^*$
$C_{b,i} \leftarrow \boldsymbol{Enc}(m_{b,i})$
$C_{b,i} = (g^{m_{b,i}} y_3^{r_{b,i}}, h^{r_{b,i}})$
$C_{b,i}$
$C_{a*b} \leftarrow {C_{b,i}}^{m_{a,i}} y_3^{r_0}$
C_{a*b}
$g^{m_{a,i} m_{b,i}} \leftarrow \boldsymbol{Dec}(C_{a*b})$
$k_n \leftarrow g^{m_{a,i} m_{b,i}}$
k_n

Figure 2

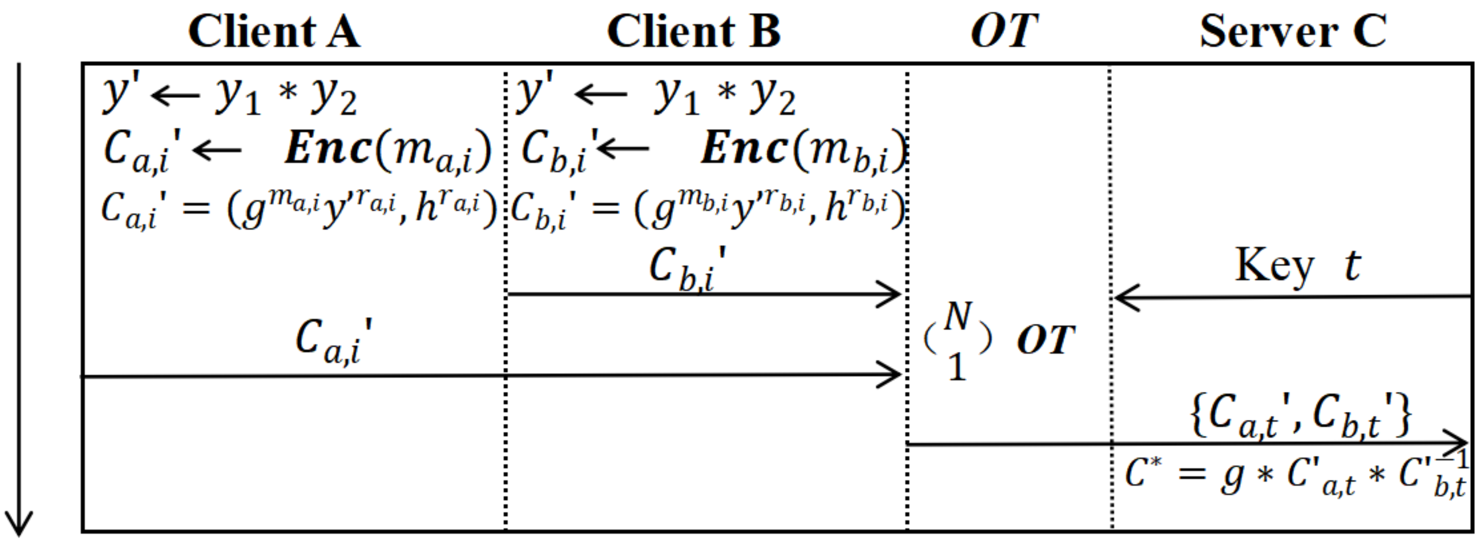
Exponential multiplication phase



Client A
Server C
$C_{a+b} \leftarrow C_{a,k_n} C_{b,k_n}$
$C_{a,k_n} C_{b,k_n} = (g^{m_{a,i}+m_{b,i}} y_3^{r_{a,i}+r_{b,i}}, h^{r_{a,i}+r_{b,i}})$
C_{a+b}
$g^{m_{a,i}+m_{b,i}} \leftarrow \boldsymbol{Dec}(C_{a+b})$
$c_{\oplus} = m_{ai} + m_{2,b} - 2m_{a,i}m_{b,i}$
Key $t \leftarrow c_{\oplus}$

Figure 3

Additive homomorphism phase



Client A
Client B
OT
Server C
$y' \leftarrow y_1 * y_2$
$C_{a,i}' \leftarrow \boldsymbol{Enc}(m_{a,i})$
$C_{a,i}' = (g^{m_{a,i}} y'^{r_{a,i}}, h^{r_{a,i}})$
$y' \leftarrow y_1 * y_2$
$C_{b,i}' \leftarrow \boldsymbol{Enc}(m_{b,i})$
$C_{b,i}' = (g^{m_{b,i}} y'^{r_{b,i}}, h^{r_{b,i}})$
$C_{b,i}'$
$C_{a,i}'$
$\binom{N}{1}$ OT
Key t
$\{C_{a,t}', C_{b,t}'\}$
$C^* = g * C'_{a,t} * C'^{-1}_{b,t}$

Figure 4

The OT phase

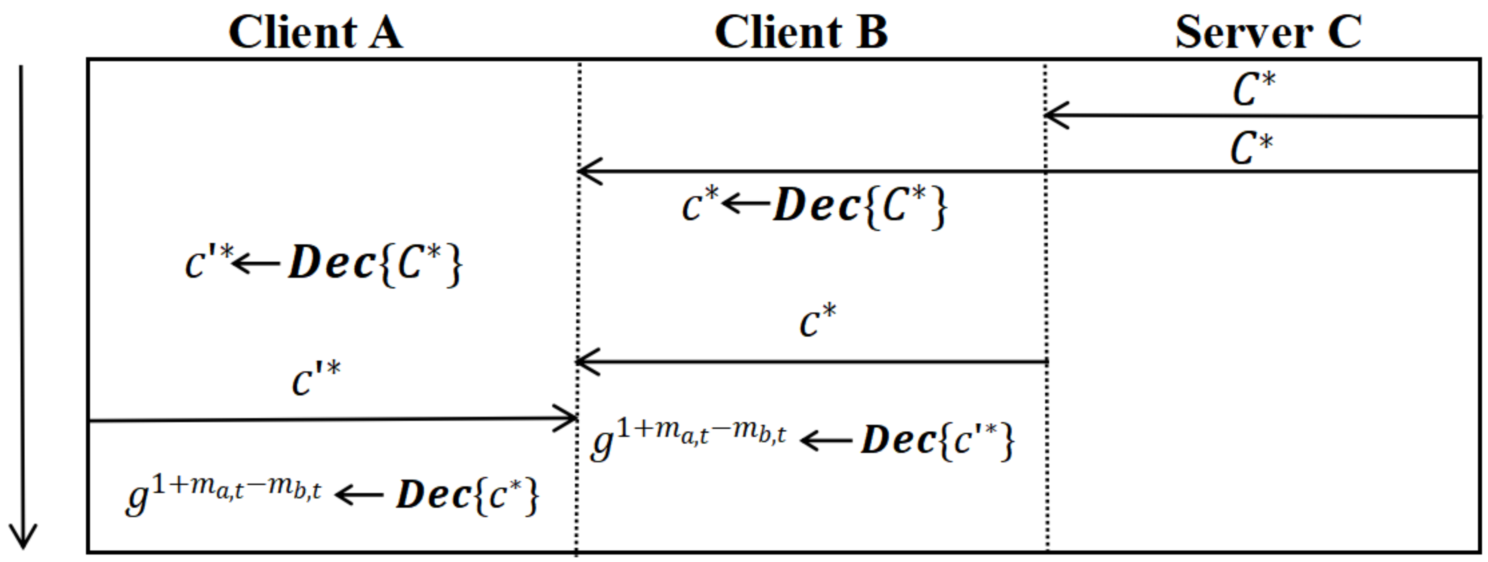

Client A
Client B
Server C
C^*
C^*
$c^* \leftarrow \mathbf{Dec}\{C^*\}$
$c'^* \leftarrow \mathbf{Dec}\{C^*\}$
c^*
c'^*
$g^{1+m_{a,t}-m_{b,t}} \leftarrow \mathbf{Dec}\{c'^*\}$
$g^{1+m_{a,t}-m_{b,t}} \leftarrow \mathbf{Dec}\{c^*\}$

Figure 5

Verification phase