RESEARCH ARTICLE

Secure two-party computation of solid triangle area and tetrahedral volume based on cloud platform

Jing Zhang\textsuperscript{1,2,3*}, Lixiang Li\textsuperscript{2,3}, Yongli Tang\textsuperscript{2}, Shoushan Luo\textsuperscript{3}, Yixian Yang\textsuperscript{1,3}, Yang Xin\textsuperscript{3}

\textsuperscript{1} School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China, \textsuperscript{2} School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan, China, \textsuperscript{3} Information Security Center, National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing, China

* zj_jsj@sina.com

Abstract

With the emergence and widespread application of cloud computing, the use of cloud platforms to solve the problem of secure multi-party computation has emerged as a new research direction. The traditional computation of a solid geometry is performed through mutual interactions between two parties, which is not suitable in an untrusted cloud computing environment. In this paper, we first design a basic protocol for a secure Euclidean distance calculation that is suitable for cloud platforms and can serve as a building block for other protocols on cloud platforms. Using the solution of the Euclidean distance problem as such a building block, we provide a new method that converts the problems of calculating solid triangular areas and solid tetrahedral volumes into the calculation of distances and determinants in three-dimensional space. Then, we discuss solid point-line distance calculations, which extent the idea of the spatial geometry security problem. We present protocols for the above problems and prove that the proposed protocols can resist conspiracy among users and the untrusted cloud platform so that they can effectively ensure the privacy of the users. We also analyze the performances of these solutions. The analysis results show that our scheme is more versatile.

Introduction

Secure multi-party computation is an important cryptographic primitive in the fields of modern cryptography and communication. In 1982, Yao et al. \cite{1} raised the question of a millionaire. After in-depth research, Goldreich et al. theoretically proved that all SMC problems are solvable, and they proposed a universal solution \cite{2} \cite{3}. Subsequently, many researchers focused on the study of SMC and obtained a rich variety of research results \cite{4} \cite{5} \cite{6} \cite{7} \cite{8} \cite{9}. Some schemes have been used to solve problems such as electronic voting, organizing business flows, network data flows, position determination, and medical information \cite{10} \cite{11} \cite{12} \cite{13}.
With the rise of cloud computing applications, increasingly more users want to entrust cloud platforms to perform intersection computing on private information. For example, Bob has two mixtures $\xi_1$ and $\xi_2$, which contain components $M_1$, $M_2$ and $M_3$. Alice needs a new mixture, $\xi_3$, that contains components $M_1$, $M_2$ and $M_3$. With the cloud server, Alice wants to determine if she can produce this mixture from the two mixtures possessed by Bob. However, she does not want to disclose her needs. Similarly, Bob does not want to disclose the contents of his mixtures. This problem can be transformed into a secure two-party triangle area calculation problem based on cloud computing. Specifically, the participant uploads private data after processing to the cloud and then performs the computation required by the users with the cloud server. During the implementation, the cloud server does not obtain any private information of any participant. However, the cloud cannot be completely trusted because an adversary may perform improper actions through the cloud platform such as tampering with the sensitive data of the client or with the calculation results. How to perform computations concerning the allocation of mixtures with an untrusted cloud while protecting the user’s privacy is a challenging problem.

Advantages of the scheme

In this paper, we move the traditional security computational geometry problem to the cloud platform, and we solve some solid geometry security problems. The main contributions of this study are as follows:

(1). Designing a solution for the secure multiparty computation of the square of the Euclidean distance (SMC-SED) with a cloud platform. In this scheme, Boneh encryption and blind factor are used to protect the user’s privacy. The benefit here is that the private user data participate in the subsequent calculation in the form of a ciphertext. Security and experimental analyses prove that our scheme can resist collusion between users and untrusted cloud platforms and is optimal. This solution is a building block and can be regarded as a new cloud computing technology.

(2). Focusing on the problem of triangular areas and that of tetrahedral volumes in three-dimensional space, we propose corresponding solutions, the Secure multiparty computation of triangle area calculation (SMC-TA) protocol and the Secure multiparty computation of tetrahedral volume calculation (SMC-TV) protocol. These new schemes convert the solid triangular area problem and the solid tetrahedral volume problem into the calculation of distances and determinants based on (1). Unlike traditional solutions, the cloud server is introduced as a third party to complete the core computing of the protocols. Even if an adversary were to break into the cloud server or compromise one of the users, he would not be able to obtain Bob’s private information. We prove the privacy-preserving property of these solutions using the security proof model, and we compare their performance with different schemes. The results show that our scheme is more versatile.

(3). Based on the secure multiparty computation solution for the triangle area computation, we solve the secure multiparty problem of the relationship between a point and a line and propose corresponding solutions. The security of this scheme is based on SMC-TA.

Related work

The general SMC based on cloud services. Maheshwari et al. [14] proposed SMC solution techniques that could be embedded while designing a cloud computing architecture,
especially when multiple cloud users jointly compute some function on their private data inputs. Kamara et al. [15] proposed the formalized definition of server-assisted secure multiparty computation, which required that there was no collusion between an ordinary participant and the server. On this basis, Kamara et al. [16] presented the concept of the secure function calculation of cloud auxiliary, and they constructed a protocol for the secure function calculation of a single server auxiliary. Carter et al. [17] proposed an implementation mechanism for private data protection in a cloud environment based on the outsourcing technology of Oblivious Transfer. This mechanism effectively mitigated dishonest behavior by cloud service providers. Compared to Yao’s chaotic circuit technology, the structure of SMC based on a cloud service is more secure given the use of homomorphic encryption techniques. Asharov et al. [18] constructed a protocol for SMC based on cloud services with a threshold homomorphic encryption scheme. In this protocol, participants are only required to execute calculations associated with the protocol, and the secure computation was performed by the cloud server. Lopez-alt et al. [19] proposed a complete multi-key homomorphic encryption scheme, and based on this scheme, they constructed the On-the-Fly Multi-party Computation (OFMC) protocol, which was secure under malicious adversaries.

The special SMC based on cloud services. Kerschbaum et al. [20] presented the non-interactive encrypted computation of the set intersection operation using an untrusted service provider. This service provider computed the intersection result after the users had submitted their encrypted sets to the service. The server could not obtain any information about the computation process. Kamara et al. [21] designed private set intersection (PSI) protocols in the server-aided setting, where the parties had access to a single untrusted server that made its computational resources available as a service. These protocols are secure in several adversarial models, and they address a range of security and privacy concerns such as fairness and leakage of the intersection size. Abadi et al. [22] designed a PSI on outsourced datasets based on a novel point-value polynomial representation, which ensured that intersections could only be calculated with the permission of all clients and that datasets and results remained completely confidential from the server. Veugen et al. [23] provided a generic framework that allowed an arbitrary number of users to securely outsource a computation to two non-colluding external servers with the help of a pre-processing phase that was independent of the inputs of the users. This approach was shown to be provably secure in an adversarial model. To address the inefficiency of previous schemes, Chen et al. [24] first transformed the original problem into a one-time evaluation problem for polynomials, and then, they designed four efficient and concise cloud computing environments to outsource the user set computing protocol. The analysis and comparison showed that these protocols were more efficient and concise than previously developed protocols. Chen et al. [25] transformed this traditional pattern into a cloud computation protocol that allowed an untrusted third party to be involved in the calculation process. They also designed a protocol for scalar product calculations applicable to cloud computing. On this basis, Chen the designed five solutions for spatial location relations. Although many researchers have begun to focus on SMC based on cloud services, fewer results on specific computing problems based on cloud services, especially for multi-party geometric computation, have been obtained. Research on such issues is attractive.

This paper focuses on some solid geometry problems based on cloud computing and their applications. We design the solution of secure multiparty computation of the square of the Euclidean distance using Boneh encryption. Using this solution as a building block, we solve the problem of triangular areas and that of tetrahedral volumes in three-dimensional
space. On this basis, we give the protocol of the secure multiparty problem of the relationship between a point and a line.

**Preliminaries**

We briefly review the groups underlying our scheme.

**Boneh encryption algorithm**

Suppose that $E$ is an encryption algorithm and that we are given the encryptions $C_1, C_2 \in G$ of messages $m_1$ and $m_2$, respectively, where $C_1 = E(m_1, r_1)$ and $C_2 = E(m_2, r_2)$, in which $r_1, r_2$ are random numbers. We describe Boneh’s encryption algorithm and its homomorphic [26] as follows.

KeyGen $(r)$: Let $N = pq$, where $p, q$ are two random primes. Generate a bilinear map $e$: $(G_1 \times G_1 \rightarrow G), G_1, G$ are groups with order $N = p \cdot q$. Pick two random generators $g, u \in G$ and set $h = u^r$. Then, $h$ is a random generator of the subgroup of $G$ with order $p$. The public key is $PK = (G_1, G, e, g, h, N)$, and the private key $SK = q$.

Encrypt $(PK, m)$: Assume that the message space consists of integers in the set $\{0, 1, \ldots, n - 1\}$ with $T < q$. To encrypt a message $m (m < p)$ using the public key $PK$, pick a random number $r \in Z_N$, compute $c = g^m h^r \in G$ and output $c$ as the ciphertext.

Decrypt $(SK, c)$: To decrypt a ciphertext $c$ using the private key $SK = q$, observe that $c^q = (g^m h^r)^q = (g^m)^q = m$. Let $g' = g^q$. To recover $m$, it suffices to compute the discrete logarithm of $c^q$ base $g'$.

The Boneh encryption algorithm is clearly additively homomorphic. Suppose that $m_1$ and $m_2$ are messages. We have
\[
E(m_1) \cdot E(m_2) = g^m h^{r_1} \cdot g^{m_2} h^{r_2} = g^{m_1 + m_2} h^{r_1 + r_2}
\]
\[
= g^{m_1 + m_2} h^r = E(m_1 + m_2, r).
\]

Anyone can create a uniformly distributed encryption of $m_1 + m_2$ by the above formula for a random $r$.

More importantly, anyone can multiply two encrypted messages once using the bilinear map. Set $e(g, g) = g, e(g, h) = h_1$, write $h = g^{\alpha r}$ for some (unknown) $\alpha$, and pick a random $r$; then, we obtain
\[
e(E(m_1), E(m_2))
\]
\[
= e(g^{m_1} \cdot h^{r_1}, g^{m_2} \cdot h^{r_2}) \cdot h_1^r
\]
\[
= e(g, g)^{m_1 m_2} \cdot e(g, h)^{m_1 r_1 + m_2 r_2 + \alpha pr_1 r_2} \cdot h_1^r
\]
\[
= g^{m_1 m_2} h_1^r
\]
where $r' = m_2 r_1 + m_1 r_2 + \alpha p r_1 r_2 + r$ is distributed uniformly.

**Cayley-Menger determinant**

The Cayley-Menger determinant is often used to address the Euclidean distance problem of an invariant space. This plays a fundamental role in the so-called “distance geometry” [27]. The Cayley-Menger bideterminant of two sequences with $n$ vectors $(p_1, p_2, \ldots, p_n)$ and $(q_1, q_2, \ldots, q_n)$
$D(p_1, p_2, \ldots p_n; q_1, q_2, \ldots q_n) \in \mathbb{R}^n$ is defined as

$$D(p_1, p_2, \ldots p_n; q_1, q_2, \ldots q_n) = 2 \left( -\frac{1}{2} \right)^n \begin{vmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & D(p_1, q_1) & D(p_1, q_2) & \cdots & D(p_1, q_n) \\ 1 & D(p_2, q_1) & D(p_2, q_2) & \cdots & D(p_2, q_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & D(p_n, q_1) & D(p_n, q_2) & \cdots & D(p_n, q_n) \end{vmatrix}$$

where $D(p_i, q_i)$ denotes the squared distance between $p_i$ and $q_i$. In many cases, these two sequences are the same. Hence, it will be convenient to abbreviate $D(p_1, p_2, \ldots p_n; q_1, q_2, \ldots q_n)$ by $D(p_1, p_2, \ldots p_n)$, which is simply called a Cayley-Menger determinant.

In $m$-dimensional Euclidean space, the rank of the matrix of Cayley-Menger determinants is no greater than $m + 1$. Specifically, $D(p_1, p_2, \ldots p_n) = 0$ when $n \geq m + 1$. Let us show the geometric interpretation of Cayley-Menger determinants when $n = 2, 3, 4$ and $m = 3$.

For $n = 2$,

$$D(p_1, p_2) = d(p_1, p_2)^2$$

where $D(p_1, p_2)$ is the Euclidean distance between $p_1$ and $p_2$. Observe that the squared distance between $p_1$ and $p_2$ is consistent with the result of the Cayley-Menger determinant.

For $n = 3$,

$$D(p_1, p_2, p_3) = 4A^2 = \|p_2 - p_1\| \times \|p_3 - p_1\|^2 = 2 \cdot \left( -\frac{1}{2} \right)^3 \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & L_{12}^2 & L_{13}^2 \\ 1 & L_{12}^2 & 0 & L_{23}^2 \\ 1 & L_{13}^2 & L_{23}^2 & 0 \end{vmatrix}$$

where $A$ is the area of the triangle spanned by $p_1, p_2, p_3$, and $\|\|$ is the length of vector. $L_{ij}$ is the distance between $p_i$ and $p_j$, $i, j = 1, 2, 3$.

For $n = 4$,

$$D(p_1, p_2, p_3, p_4) = 36V^2 = 2 \cdot \left( -\frac{1}{2} \right)^4 \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & L_{12}^2 & L_{13}^2 & L_{14}^2 \\ 1 & L_{12}^2 & 0 & L_{23}^2 & L_{24}^2 \\ 1 & L_{13}^2 & L_{23}^2 & 0 & L_{34}^2 \\ 1 & L_{14}^2 & L_{24}^2 & L_{34}^2 & 0 \end{vmatrix}$$

where $V$ is the volume of the tetrahedron spanned by $p_1, p_2, p_3, p_4$, and $L_{ij}$ is the distance...
between $p_i$ and $p_{ij}$, $i, j = 1, 2, 3, 4$.

Security proof in the semi-honest model
In the semi-honest model, the parties abide by the protocol. However, they keep a record of all the intermediate computations and expect to deduce the private inputs of other parties from the record. The security of secure two-party computations in the semi-honest model can be described as follows. There is a probability polynomial-time algorithm (referred to as a simulator). Using this simulator, any semi-honest participant can simulate the execution process of the protocol alone and obtain all the intermediate information using his own inputs and the final result from the protocol.

**Definition 1** (The security of secure two-party computations in the semi-honest model) Let $f : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^*$ be a functionality, where $f(x, y)$, $i = 1, 2$ is the $i$th element of $f(x, y)$ and $\Pi$ is a two-party protocol for computing $f$ (denoted as $\hat{f}$). The view of the $i$th party during an execution of $\Pi$ on $(x, y)$, denoted as $\text{VIEW}^i_{\Pi}(x, y)$, is $(x, r, m_1, \ldots, m_n)$, where $r$ represents the outcome of the $i$th party’s internal coin tosses, and $m_j$ represents the $j$th message that the $i$th party has received. Let $\text{OUTPUT}^i_{\Pi}(x, y)$ be the $i$th output result. If there exists a probabilistic polynomial-time algorithm, denoted as $S_1$ and $S_2$, making (1) and (2) workable, we say that $\Pi$ privately computes $f$.

\[
\{S_i(x, f_1(x, y))\}_{x, y \in \{0, 1\}^*, r, l, [x] = [y]} \equiv \{\text{VIEW}^i_{\Pi}(x, y)\}_{x, y \in \{0, 1\}^*, r, l, [x] = [y]} \tag{1}
\]

\[
\{S_2(y, f_2(x, y))\}_{x, y \in \{0, 1\}^*, r, l, [x] = [y]} \equiv \{\text{VIEW}^2_{\Pi}(x, y)\}_{x, y \in \{0, 1\}^*, r, l, [x] = [y]} \tag{2}
\]

Problems
This paper studies the following problems.

Problem 1: Secure multiparty computation of the square of the Euclidean distance (SMC-SED). Alice has a private point $P_A = (x_1, x_2, \ldots, x_n)$, and Bob has a private point $P_B = (y_1, y_2, \ldots, y_n)$, where $(n \geq 3)$. Alice and Bob want to know the Euclidean distance between the points $P_A = (x_1, x_2, \ldots, x_n)$ and $P_B = (y_1, y_2, \ldots, y_n)$, where $(n \geq 3)$, denoted by $D^2(P_A, P_B)$, without disclosing $P_A$ or $P_B$.

Problem 2: Secure multiparty computation of the area of a triangle (SMC-TA). Alice has a private point $P_A = (x_A, y_A, z_A)$, and Bob has private two points $P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1})$ and $P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2})$. Alice and Bob want to know the area of the triangle formed by the points $P_A = (x_A, y_A, z_A)$, $P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1})$, $P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2})$, denoted by $S_{\triangle A B_1 B_2}$, without disclosing $P_A, P_{B_1}$ or $P_{B_2}$.

Problem 3: Secure multiparty computation of the volume of a tetrahedron (SMC-TV). Alice has a private point $P_A = (x_A, y_A, z_A)$, and Bob has two private points $P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1})$, $P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2})$, and $P_{B_3} = (x_{B_3}, y_{B_3}, z_{B_3})$. Alice and Bob want to know the volume of the tetrahedron formed by the points $P_A = (x_A, y_A, z_A)$, $P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1})$, $P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2})$, and $P_{B_3} = (x_{B_3}, y_{B_3}, z_{B_3})$, denoted by $V_{\triangle A B_1 B_2 B_3}$, without disclosing $P_A, P_{B_1}, P_{B_2}, P_{B_3}$.
Problem 4: The distance between a point and a line (SMC-DPL). Alice has a private point \( P_A = (x_A, y_A, z_A) \), and Bob has a private line

\[
\begin{aligned}
L : \quad & A_1x + B_1y + C_1z + D_1 = 0 \\
& A_2x + B_2y + C_2z + D_2 = 0
\end{aligned}
\]

Alice and Bob want to know the distance between the point \( P_A \) and the line \( L \) without disclosing \( P_A \) and \( L \).

In the next sections, we will give our solutions to the four problems in detail.

**Building block**

Alice has an \( n \)-dimensional point \( P_A = (x_1, x_2, \ldots, x_n) \), and Bob has an \( n \)-dimensional point \( P_B = (y_1, y_2, \ldots, y_n) \), where \( n \geq 3 \). Alice and Bob want to compute the Euclidean distance without disclosing the messages of their points. We call this problem Secure multiparty computation of the square of Euclidean distance (SMC-SED). This problem is the building block of the other three problems. To solve this problem, we propose a protocol for the secure multiparty computation of the square of the Euclidean distance with the Boneh encryption algorithm. The solution transfers the main calculation to the cloud, which makes it possible for users to only perform encryption and other simple operations.

**The solution of SMC-SED**

The protocol for computing the square of the Euclidean distance is depicted in Fig 1. First, Alice and Bob encrypt their private values and then send their values to the server. To do so, Bob’s values are blinded so that the server cannot determine them, even if it colludes with Alice. Then, the server uses the Boneh algorithm to calculate \( c_0, c_1, \) and \( c_2 \) and sends them to Bob. In the third step, Bob eliminates the blinding factors \( (r \) in the protocol and computes the cipher values. Therefore, Bob cannot decrypt the obtained information in this step. At the end
of the protocol, Alice decrypts the cipher values and receives the square of the Euclidean
distance. The protocol is described as follows.

**Protocol 1. SMC-SED**

**Inputs:** Alice’s input is \( P_A = (x_1, x_2, \ldots, x_n) \). Bob’s input is \( P_B = (y_1, y_2, \ldots, y_n) \), where \( n \geq 3 \).

**Outputs:** The square of the Euclidean distance formed by \( P_A, P_B \).

**Step 1.** Based on the Boneh encryption algorithm, Alice picks suitable parameters and generates the public-private key pair (\( PK_A, SK_A \));

**Step 2.** Alice computes \( E_{PK_A}(x_i), i = 1 \ldots n \) and sends it to the cloud server;

**Step 3.** Bob picks a random number \( r(\neq 0) \) and computes \( E_{PK_A}(2ry_i) \), where \( i = 1 \ldots n \). Then, Bob sends \( E_{PK_A}(2ry_i) \) to the cloud server;

**Step 4.** The cloud server obtains the messages that originated from Alice and Bob, and it performs the following calculations with the homomorphic property of the Boneh encryption algorithm:

\[
\begin{align*}
c_0 &= e(E_{PK_A}(x_1), E_{PK_A}(2ry_1)) \ldots e(E_{PK_A}(x_n), E_{PK_A}(2ry_n)) \\
&= E(2rx_1y_1 + 2rx_2y_2 + \ldots + 2rx_ny_n) \\
&= E(-2x_1y_1 - 2x_2y_2 - \ldots - 2x_ny_n)^r
\end{align*}
\]

\[
\begin{align*}
c_1 &= e(E_{PK_A}(x_1), E_{PK_A}(x_1)) \ldots e(E_{PK_A}(x_n), E_{PK_A}(x_n)) \\
&= E_{PK_A}(x_1^2 + \ldots + x_n^2)
\end{align*}
\]

\[
\begin{align*}
c_2 &= e(E_{PK_A}(2ry_1), E_{PK_A}(2ry_1)) \ldots e(E_{PK_A}(2ry_n), E_{PK_A}(2ry_n)) \\
&= E_{PK_A}(4r^2(y_1^2 + \ldots + y_n^2)) \\
&= E_{PK_A}(y_1^2 + \ldots + y_n^2)^{4r^2}
\end{align*}
\]

Then, the cloud server sends \( c_0, c_1, c_2 \) to Bob;

**Step 5.** Bob computes \( R = c_0^r + c_1^r + c_2^{-4r^2} \) and sends \( R \) to Alice;

**Step 6.** Alice computes \( L_{AB}^2 = D_{SK_A}(R) \) and tells the result to Bob.

**Security**

As we know, the cloud server is untrusted. Therefore, we have to consider the security of private
information in the cloud and the complicity between the cloud and any participant. Therefore,
the security model of the protocol is slightly different than the traditional model. This will be
proved with five formulas in Table 1. [25]

Theorem 1. The SMC-SED protocol, denoted by \( \Pi \), is private, where \( n \geq 3 \).

Proof. We will prove this theorem by showing five simulators \( S_1, S_2, S_3, S_4, S_5 \).

1. We first show the construction of \( S_1 \). Based on the inputs \( P_A = (x_1, x_2, \ldots, x_n) \) and \( L_{AB}^2, S_1 \) proceeds as follows.
Table 1. The security of the protocol based on the cloud platform.

| Type               | Privacy                          | Simulation                                                                 |
|--------------------|----------------------------------|---------------------------------------------------------------------------|
| No collusion       | Bob’s privacy                    | \(\{\text{view}_1^1(P_A, P_B)\} \supseteq \{S_1(P_A, f_i(P_A, P_B))\}\) |
| No collusion       | Alice’s privacy                  | \(\{\text{view}_1^1(P_A, P_B)\} \supseteq \{S_1(P_A, f_i(P_A, P_B))\}\) |
| No collusion       | Alice’s privacy and Bob’s privacy| \(\{\text{view}_1^1(P_A, P_B)\} \supseteq \{S_1(E(P_A), E(P_B), E(P_A, P_B))\}\) |
| Collision          | Bob’s privacy (Server and Alice collision) | \(\{\text{view}_1^1(P_A, P_B)\} \supseteq \{S_1(P_A, E(P_A), E(P_B), f_i(P_A, P_B), E(P_A, P_B))\}\) |
| Collision          | Alice’s privacy (Server and Bob confusion) | \(\{\text{view}_1^1(P_A, P_B)\} \supseteq \{S_1(P_A, E(P_A), E(P_B), f_i(P_A, P_B), E(P_A, P_B))\}\) |

https://doi.org/10.1371/journal.pone.0217067.t001

1. \(S_1\) computes \(E_{PK_A}(x_i), i = 1, ..., n\). Then, \(S_1\) chooses a point \(P_B' = (y_1', ..., y_n')\) and a random number \(r'\). It computes \(E_{PK_A}(2r'y'_i)\), where \(i = 1, ..., n\);

2. \(S_1\) obtains \(E_{PK_A}(x_i), P_B' = (y_1', y_2', ..., y_n')\), where \(i = 1, 2, 3, ..., n\), and performs the following calculations with the homomorphic property of the Boneh encryption algorithm:

\[
c'_0 = e(E_{PK_A}(x_1), E_{PK_A}(2r'y'_1)) \cdots e(E_{PK_A}(x_n), E_{PK_A}(2r'y'_n)) = E(-2xy_1' - 2xy_2' - \ldots - 2xy'_n)^{-r'}
\]

\[
c_1 = e(E_{PK_A}(x_1), E_{PK_A}(x_1)) \cdots e(E_{PK_A}(x_n), E_{PK_A}(x_n)) = E_{PK_A}(x_1^2 + \ldots + x_n^2)
\]

\[
c'_2 = e(E_{PK_A}(2r'y'_1), E_{PK_A}(2r'y'_1)) \cdots e(E_{PK_A}(2r'y'_n), E_{PK_A}(2r'y'_n)) = E_{PK_A}(y_1'^2 + \ldots + y_n'^2)^{r'^2}
\]

Then, \(S_1\) computes \(R = c'_0^{-r'} \ast c_1 \ast c'_2^{-1} r'^2\);

3. \(S_1\) decrypts the result \(L_{AB}^{r'^2} = D_{SK_B}(R')\);

4. \(S_1\) outputs the message list of Alice:

\[
\{S_1(P_A, f_i(P_A, P_B))\} = \{(SK_A, PK_A), P_A, E_{PK_A}(P_A), R', L_{AB}^{r'^2}\}
\]

Note that in this protocol,

\[
\text{view}_1^1(P_A, P_B) = \{(SK_A, PK_A), P_A, E_{PK_A}(P_A), R, L_{AB}^{r'^2}\}
\]

\[
f_i(P_A, P_B) = f_i(P_A, P_B') = L_{AB}^{r'^2}
\]

Because of the choice of \(P_B' = (y_1', y_2', \ldots, y_n')\), and \(r'\), it must hold that \(L_{AB}^{r'^2} \equiv L_{AB}^{r'^2},\) and \(R \equiv R'\).
Thus, we have

\[
\text{view}_1^\Pi (P_A, P_B) \equiv S_1(P_A, f_1(P_A, P_B))
\]

(2). Now, let us examine the construction of \(S_2\). Based on the inputs \(P_B = (y_1, y_2, \ldots, y_n)\) and \(R, S_2\) proceeds as follows.

1. \(S_2\) first chooses a random generator \(r\) and then computes \(E_{PK_A}(2ry_i), i = 1, \ldots, n\). Then, \(S_2\) chooses a point \(P_A' = (x_1', \ldots, x_n')\) and computes \(E_{PK_A}(x_i'), \) where \(i = 1 \ldots n;\)

2. \(S_2\) performs the following calculations with the homomorphic property of the Boneh encryption algorithm:

\[
\begin{align*}
c_0' &= e(E_{PK_A}(x_1'), E_{PK_A}(2ry_1)) \cdots e(E_{PK_A}(x_n'), E_{PK_A}(2ry_n)) \\
&= E(-2x_1'y_1 - 2x_2'y_2 - \ldots - 2x_n'y_n)'
\end{align*}
\]

\[
\begin{align*}
c_1' &= e(E_{PK_A}(x_1'), E_{PK_A}(x_1')) \cdots e(E_{PK_A}(x_n'), E_{PK_A}(x_n')) \\
&= E_{PK_A}(x_1'^2 + \ldots + x_n'^2)
\end{align*}
\]

\[
\begin{align*}
c_2 &= e(E_{PK_A}(2ry_1), E_{PK_A}(2ry_1)) \cdots e(E_{PK_A}(2ry_n), E_{PK_A}(2ry_n)) \\
&= E_{PK_A}(y_1^2 + \ldots + y_n^2)^{2r}
\end{align*}
\]

3. \(S_2\) computes \(R' = c_0' * c_1' * c_2^{2r2}\).

4. \(S_2\) outputs the message list of Bob as follows.

\[
\{S_2(P_B, f_2(P_A, P_B))\} = \{P_B, r, E_{PK_A}(2rP_B), c_0', c_1', c_2, R'\}
\]

Note that in this protocol,

\[
\text{view}_2^\Pi (P_A, P_B) = \{P_B, r, E_{PK_A}(2rP_B), c_0, c_1, c_2, R\}
\]

\[
f_2(P_A, P_B) = f_2(P_A', P_B) = R
\]

Because of the choice of the point \(P_A' = (x_1', x_2', \ldots, x_n')\), it must hold that \(c_0, c_1, c_2 \equiv c_0', c_1', c_2, R \equiv R'.\)

Thus, we have

\[
\{\text{view}_2^\Pi (P_A, P_B)\} \equiv \{S_2(P_B, f_2(P_A, P_B))\}
\]

(3). Then, we verify the construction of \(S_3\). Based on the inputs \(E_{PK_A}(P_A), E_{PK_A}(P_B), c_0, c_1,\) and \(c_2, S_3\) proceeds as follows.
1. $S_3$ chooses

$$P'_A = (x'_1, x'_2, ..., x'_n),$$

$$P'_B = (y'_1, y'_2, ..., y'_n)$$

and a random number $r'$. $S_3$ computes $E_{pk_a}(x'_i), E_{pk_A}(2r'y'_i)$, where $i = 1, 2, 3...n$.

2. $S_3$ computes

$$c'_0 = e(E_{pk_a}(x'_1), E_{pk_A}(2r'y'_1))\ldots e(E_{pk_a}(x'_n), E_{pk_A}(2r'y'_n))$$

$$= E(-2x'_1y'_1 - 2x'_2y'_2 - \ldots - 2x'_ny'_n)^{r'}$$

$$c'_1 = e(E_{pk_a}(x'_1), E_{pk_A}(x'_1))\ldots e(E_{pk_a}(x'_n), E_{pk_A}(x'_n))$$

$$= E_{pk_a}(x'_1^2 + \ldots + x'_n^2)$$

$$c'_2 = e(E_{pk_a}(2r'y'_1), E_{pk_A}(2r'y'_1))\ldots e(E_{pk_a}(2r'y'_n), E_{pk_A}(2r'y'_n))$$

$$= E_{pk_a}(y'_1^2 + \ldots + y'_n^2)^{r'2}$$

3. $S_3$ outputs the message list of cloud

$$S_3(E(P_A), E(P_B), E(P_A, P_B)) = \{E_{pk_a}(P'_A), E_{pk_A}(P'_B), c'_0, c'_1, c'_2 \}$$

Note that in this protocol, we have

$$view^3_{s_a}(E_{pk_a}(P_A), E_{pk_A}(P_B)) = \{E_{pk_a}(P_A), E_{pk_A}(P_B), c_o, c_1, c_2 \}$$

$$f_s(P_A, P_B) = f_s(P'_A, P'_B) = R$$

Because of the choice of the points $P'_A, P'_B$ and $r'$, it must hold that

$$E_{pk_a}(P_A), E_{pk_A}(P_B) \equiv E_{pk_a}(P'_A), E_{pk_A}(P'_B),$$

Obviously, $c_o, c_1, c_2 \equiv c'_0, c'_1, c'_2$.

Thus, we have

$$view^3_{s_a}(P_A, P_B) \equiv S_3(E(P_A), E(P_B), E(P_A, P_B))$$

$S_3$ proves that the cloud server can only calculate the values from Alice and Bob using its own input and output. Because the messages that it obtains are encrypted, the cloud server cannot decrypt them. Specifically, the cloud server cannot obtain any private information from Alice or Bob.

4. We prove the construction of $S_3$.

When Alice conspires with the server, their inputs are

$$P_A, E(P_A), E(P_B), f_s(P_A, P_B), E(P_A, P_B).$$
Combining with the construction of $S_1$ and $S_3$, we simulate $S_4$ in a similar manner. $S_4$ obtains the message list of Alice and the cloud server.

$$S_4(P_A, E(P_A), E(P_B), f_1(P_A, P_B), E(P_A, P_B))$$

$$= \{ P_A, E_{pk_A}(P_A), E_{pk_A}(P'_A), c_0', c_1', c_2', R', L_{AB}^2 \}$$

Clearly, Alice can decrypt all the ciphertexts coming from the information sequence with her own key pair. However, Alice still cannot obtain Bob’s private information through the above information. If she wants to obtain Bob’s private information, she has to solve the following problem:

$$\begin{cases} 
D_{sk_a}(c_0) = \left( \sum_{i=1}^{n} 2x_i y'_i \right)^{v'} \\
D_{sk_a}(c_2) = \left( y'_1^2 + \ldots + y'_n^2 \right)^{v''} \\
\left( \sum_{i=1}^{n} x_i^2 \right) = 2 \sum_{i=1}^{n} x_i y'_i + \left( \sum_{i=1}^{n} y'_i^2 \right) = L_{AB}^2
\end{cases}$$

The number of equations is less than the number of unknowns when $i \geq 3$. The view of Alice and the server is

$$\text{view}_4^{\mathcal{V}}(P_A, P_B) = \{ P_A, E_{pk_A}(P_A), E_{pk_A}(P_B), c_0, c_1, c_2, R, L_{AB}^2 \}$$

Because of the choice of the point $P'_A = (y'_1, y'_2, \ldots, y'_n)$, it must hold that

$$E_{pk_A}(P_B) \equiv E_{pk_A}(P'_B).$$

Thus, we have

$$c_i \equiv c'_i, \; i = 0, 1, 2, \quad R \equiv R'.$$

Therefore, we can obtain

$$\text{view}_{4}^{\mathcal{V}}(P_A, P_B) \equiv S_4(P_A, E(P_A)E(P_B), f_1(P_A, P_B), E(P_A, P_B))$$

(5). Similarly, $S_5$ can be constructed.

We have

$$\text{view}_{5}^{\mathcal{V}}(P_A, P_B) \equiv S_5(P_B, E(P_A)E(P_B), f_2(P_A, P_B), E(P_A, P_B))$$

$S_4$ and $S_5$ demonstrate that even if the cloud server and the participants conspire, they cannot obtain any private information about another participant.

In conclusion, the SMC-SED protocol is privacy preserving when $n \geq 3$.

**Performance analysis**

In this section, the performance analysis of SMC-SED and other similar protocols will be discussed. For the convenience of comparison with SMC-SED, we choose protocols based on the privacy homeomorphism technique, including those documented in Refs. [28] [29] [30]. The comparison details are displayed in Table 2.

---

PLOS ONE | https://doi.org/10.1371/journal.pone.0217067  June 13, 2019 12 / 22
Communication round complexity: In SMC-SED, Alice and Bob communicate with each other one time in Step 4; thus, the communication round complexity is 1 round. Except for Rang’s protocol, whose communication cost is proportional to the original vector dimension n, all protocols have a communication efficiency of 2.

Computational complexity: We ignore the computational cost of creating random numbers and the key pair for homomorphic encryption, which can be completed in the preprocessing stage. Only the calculation phase, whose primary computational cost is a function of the dimensions and the complexity of homomorphic encryption, is considered. Let $M_N$, $M_E$, and $M_M$ represent the homomorphic encryption, modular exponentiation and modular multiplication, respectively. Different homomorphic algorithms are adopted by different protocols. The schemes in [28], [29], [30] adopted the Paillier homomorphic algorithm, whose modular operator is $p^2q^2$, while SMC-SED adopts the Boneh homomorphic algorithm, whose modular operator is $pq$. For convenience of comparison, the modular operator of SMC-SED is $M_N$. The modular operator of the other schemes is $M_{NP}$.

In SMC-SED, to calculate a distance with n dimensions, Alice needs to perform n encryptions and one decryption, i.e., $E_{PK_A}(x_i), i = 1...n$, $L_{AB}^2 = D_{SK_A}(R)$. Bob must perform n encryptions, 2 modular exponentiations and 2 modular multiplications, i.e., $E_{PK_A}(2y_i), i = 1...n$, $R = c_0' * c_1 * c_2^{-q^2}$. Therefore, the computational complexity of SMC-SED can be simplified as $(2n + 1)M_N + 2M_E + 2M_M$ (the modular operator is N). Amirbekyan’s protocol needs 2n encryptions, n modular multiplications, and n decryptions, a total of $3nM_N + nM_M$ (the modular operator is $N^2$). The computational complexity of Huang’s protocol is $(n + 2)M_N + 2nM_E + nM_M$ (modular operator is $N^2$). Although Rane’s protocol has a similar to computational complexity to Huang’s protocol, it does not suffer an non-satisfactory communication round cost.

As observed in Table 2, SMC-SED does not possess a satisfactory computational complexity, but it achieves the lowest communication round complexity using the cloud server. However, in SMC-SED, the modular operator is $N = pq$, and the radix is g. In the other protocols, the modular operator is $N_2 = N_1^2 = p^2q^2$, and the radix is $g^m$. Because $N_1 \gg N_2, g^m \gg g$, SMC-SED reduces the number of modular power operations at high orders of magnitude and reduces the users’ computing costs. Therefore, the computational complexity of SMC-SED is better than that of the other protocols.

**Experimental analysis**

Here, we give a quantitative analysis for our scheme and the other protocols. The runtime environment is an Intel Core i5 @CPU at 3.2GHz with 4.00 GB of RAM. The software runtime environment is Win10 64-bit and Python 3.6. The modulus N in the homomorphic algorithms is 256, 512, 1024 and 2048 bits. The times for modular exponentiation under the different modules and different exponential sizes are listed in Table 3.

From Table 3, we see that if the exponent size remains unchanged, the time will increase with increasing modulus, and the growth factor is approximately 2.1. According to Table 2, if
we suppose that the time cost of SMC-SED is $T_A = (2n + 1)M_N$, the time cost of Amirbekyan’s protocol is $T_A = 3n^2 2.1M_N$, the time cost of Rane’s protocol is $T_R = (n + 1)^2 2.1M_N$, and the time cost of Huang’s protocol is $T_A = (n + 2)2.1M_N$. Compared with $M_N$, the cost of $M_M$ and $M_E$ is very small, and from Table 2, we can see that different protocol differ little in the cost of $M_E$ and $M_M$. For convenience of calculation, we ignore the time cost of the modular exponentiation and the modular multiplication. Obviously, the overall computational cost of the secure computation protocols is consistent with that of modular exponentiation. Assume that the exponent size is 160, N is 256, 512, 1024 or 2048 and $n = 3$. We compare the trend of the time cost of the different protocol (see Fig 2). The horizontal axis is the bits of N, and the vertical axis is the time cost (ms).

It can be observed from Fig 3 that the actual experimental simulation is completely consistent with the theoretical analysis. SMC-SED indeed reduces the computing time for the users compared with the protocol in [28–30]. Therefore, SMC-SED is effective and efficient in a practical environment.

**Solutions**

In this section, we use SMC-SED as the basic subprotocol, and we solve the problems of the triangle area and volume calculations and give an extend protocol about the point-line distance.
SMC-TA

The problem of SMC-TA is as follows: Alice has a point \( P_A = (x_A, y_A, z_A) \), and Bob has two points \( P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1}) \), \( P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2}) \). These three points can form a triangle in space. Alice and Bob want to know the area of the triangle without disclosing their private messages. To solve this problem, we first calculate the square of the Euclidean distances \( L_{AB_1}^2 \) and \( L_{AB_2}^2 \), and then, we convert the problem of a triangle area into a determinant calculation about distances. The details of the protocol are as follows.

Protocol 2: Secure multiparty computation of the triangle area

**Input:** Alice’s input is \( P_A = (x_A, y_A, z_A) \), and Bob’s inputs are \( P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1}) \), \( P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2}) \).

**Output:** The area \( S_{\Delta AB_1B_2} \) of the triangle is formed by \( P_A, P_{B_1}, P_{B_2} \).

**Step 1.** Alice and Bob compute \( L_{AB_1}^2, L_{AB_2}^2 \) using SMC-SED. In addition, Alice obtain \( L_{AB_1}^2, L_{AB_2}^2 \).

**Step 2.** Bob computes \( L_{B_1B_2}^2 \) and sends it to Alice.

**Step 3.** Alice computes

\[
S_{\Delta AB_1B_2} = \sqrt{\left( \frac{1}{4} \right) \cdot 2 \cdot \left( \frac{-1}{2} \right)^2 \cdot A}
\]

where

\[
A = \begin{bmatrix}
0 & 1 & 1 & 1 \\
1 & 0 & L_{AB_1}^2 & L_{AB_2}^2 \\
1 & L_{AB_1}^2 & 0 & L_{B_1B_2}^2 \\
1 & L_{AB_2}^2 & L_{B_1B_2}^2 & 0
\end{bmatrix}
\]
Then, Alice sends the result to Bob.

**SMC-TV**

Similar to protocol 2, if Alice has a point \( P_A = (x_A, y_A, z_A) \) and Bob has three points \( P_{B_1} = (x_{B_1}, y_{B_1}, z_{B_1}), P_{B_2} = (x_{B_2}, y_{B_2}, z_{B_2}), P_{B_3} = (x_{B_3}, y_{B_3}, z_{B_3}) \). These four points form a tetrahedron in space. Alice and Bob want to know the volume of a tetrahedron without disclosing their private information. We call this problem the secure multiparty computation of the volume of a tetrahedron.

**Protocol 3: Secure multiparty computation of the tetrahedron volume**

**Input:** Alice’s input is \( P_A \), and Bob’s inputs are \( P_{B_1}, P_{B_2}, P_{B_3} \).

**Output:** The volume \( V_{AB_1B_2B_3} \) of the tetrahedron is formed by \( P_A, P_{B_1}, P_{B_2}, P_{B_3} \).

Step 1. Using SMC-ED, Alice and Bob compute

\[
L_{AB_1}^2 = D(P_A, P_{B_1}),
\]

\[
L_{AB_2}^2 = D(P_A, P_{B_2}),
\]

\[
L_{AB_3}^2 = D(P_A, P_{B_3}).
\]

In addition, Alice gets the results.

Step 2. Bob computes \( L_{B_1B_2}^2, L_{B_1B_3}^2, L_{B_2B_3}^2 \) and sends them to Alice;

Step 3. Alice computes

\[
V_{AB_1B_2B_3} = \frac{1}{36} \cdot 2 \cdot \left( \frac{1}{2} \right)^3 \cdot \begin{vmatrix}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & L^2AB_1 & L^2AB_2 & L^2AB_3 \\
1 & L^2AB_1 & 0 & L^2B_1B_2 & L^2B_1B_3 \\
1 & L^2AB_2 & L^2B_2B_1 & 0 & L^2B_2B_3 \\
1 & L^2AB_3 & L^2B_3B_1 & L^2B_3B_2 & 0
\end{vmatrix}
\]

and sends the result to Bob.

**SMC-DPL**

Using the SMC-TA as a building block, we further solve the problem of the secure multiparty computation of a point and a line. Specifically, Alice has a private point \( P_A = (x_A, y_A, z_A) \), and Bob has a private line

\[
L : \begin{cases}
A_1x + B_1y + C_1z + D_1 = 0 \\
A_2x + B_2y + C_2z + D_2 = 0
\end{cases}
\]

Alice and Bob want to know the distance between the point and the line without disclosing their private information.
Theorem 2. In the semi-honest model, SMC-TA is private.

Security

Specifically, Alice must solve the following problem:

\[
\begin{align*}
(x_A - x_{B_1})^2 + (y_A - y_{B_1})^2 + (z_A - z_{B_1})^2 &= L_{AB_1}^2, \\
(x_A - x_{B_2})^2 + (y_A - y_{B_2})^2 + (z_A - z_{B_2})^2 &= L_{AB_2}^2, \\
(x_{B_1} - x_{B_1})^2 + (y_{B_1} - y_{B_1})^2 + (z_{B_1} - z_{B_1})^2 &= L_{B_1B_1}^2.
\end{align*}
\]

Clearly, there are three equations. The number of equations is less than the number of unknowns. Therefore, it is difficult for Alice to obtain Bob’s private information. On the other side, Bob has \(L_{B_1B_2}^2, S_{AB1B_2} \) and he can calculate the height of the triangle according to the formula for the area of a triangle \( S_{AB1B_2} = \frac{1}{2} L_{B_1B_2} \cdot h \). However, Bob can only extrapolate the potential location of \( P_1 \) and not obtain the exact location.

The construction of simulators of the proof is similar to that of Theorem 1. Therefore, we construct the simulators \( S_1, S_2, S_3, S_4, \) and \( S_5 \) as follows:

1. The construction of the simulator \( S_1 \)
   Based on \( P_A \) and \( S_{AB1B_2} \), \( S_1 \) chooses two points \( P'_{B_1} = (x'_{B_1}, y'_{B_1}, z'_{B_1}), P'_{B_2} = (x'_{B_2}, y'_{B_2}, z'_{B_2}) \) and computes \( L_{AB_1}^2, L_{AB_2}^2, L_{B_1B_2}^2 \). Then, \( S_1 \) computes \( S_{AB1B_2} \) by the Cayley-Menger determinant. \( S_1 \) obtains the following list:

\[
S_1(P_A, f_S(P_A, P_B)) = \{ P_A, L_{AB_1}^2, L_{AB_2}^2, L_{B_1B_2}^2, S_{AB1B_2} \}
\]
Note that in this protocol, we have
\[
\text{view}_1^j(P_A, P_B) = \{P_A, L_{AB_1}^2, L_{AB_2}^2, L_{B_1B_2}^2, S_{AAB_1B_2}\}
\]

Because of the choice of the points $P_{B_1}, P_{B_2}$, it must hold that
\[
L_{AB_1}^2 \leq L_{AB_2}^2 \leq S_{AAB_1B_2}
\]

Thus, we have
\[
\{\text{view}_1^j(P_A, P_B)\} \equiv \{S(P_A, f_A(P_A, P_B))\}
\]

2. Similarly, we can construct the other four simulators $S_2, S_3, S_4$, and $S_5$ and obtain the following formula.
\[
\{\text{view}_2^j(P_A, P_B)\} \equiv \{S_2(P_B, f_B(P_A, P_B))\}
\]
\[
\text{view}_2^j(P_A, P_B) \equiv S(E(P_A), E(P_B), E(P_A, P_B))
\]
\[
\text{view}_4^j(P_A, P_B) \equiv S_4(P_B, E(P_A), f_2(P_A, P_B), E(P_A, P_B))
\]
\[
\text{view}_5^j(P_A, P_B) \equiv S_5(E(P_B), E(P_A), f_3(P_A, P_B), E(P_A, P_B))
\]

Theorem 3. In the semi-honest model, SMC-TV, denoted by $\Pi$, is private.

Proof: In our protocol, Alice knows $L_{AB_1}^2, L_{AB_2}^2, L_{B_1B_2}^2, L_{B_1B_2}^2$, and $L_{B_2B_2}^2$, which are the squares of the six sides of the tetrahedron. However, if Alice wants to obtain Bob’s private information, she has to find a triangle on the concentric sphere with radii $L_{AB_1}^2, L_{AB_2}^2, L_{AB_2}^2$, where the lengths of the three sides of this triangle are $L_{B_1B_2}, L_{B_1B_2}$, and $L_{B_2B_2}$. Thus, Alice should solve the following problem:
\[
\begin{align*}
(x_A - x_B_1)^2 + (y_A - y_B_1)^2 + (z_A - z_B_1)^2 &= L_{AB_1}^2 \\
(x_A - x_B_2)^2 + (y_A - y_B_2)^2 + (z_A - z_B_2)^2 &= L_{AB_2}^2 \\
(x_A - x_{B_1b})^2 + (y_A - y_{B_1b})^2 + (z_A - z_{B_1b})^2 &= L_{AB_1}^2 \\
(x_{B_1b} - x_B_1)^2 + (y_{B_1b} - y_B_1)^2 + (z_{B_1b} - z_B_1)^2 &= L_{B_1B_1}^2 \\
(x_{B_1b} - x_B_2)^2 + (y_{B_1b} - y_B_2)^2 + (z_{B_1b} - z_B_2)^2 &= L_{B_1B_2}^2 \\
(x_{B_2b} - x_B_2)^2 + (y_{B_2b} - y_B_2)^2 + (z_{B_2b} - z_B_2)^2 &= L_{B_2B_2}^2
\end{align*}
\]

Clearly, the number of equations is less than the number of unknowns. Alice cannot determine Bob’s private information. On the other hand, Bob knows $L_{B_1B_1}^2, L_{B_1B_2}^2, L_{B_2B_2}^2$ and $V_{AB_1B_2B_2}$. According to the relation between the area of a triangle and the volume of a
tetrahedron, Bob can calculate the height of the tetrahedron. However, he cannot determine the exact location of \( P_A \). The construction of the simulators in theorem 3 is similar to that of theorem 2.

Theorem 4: The protocol for computing the point-line distance is private.

Proof: Protocol 4 is an extension of protocol 2; thus, its privacy is determined by Theorem 2.

### Complexity and comparison

#### Complexity

Computational complexity: In SMC-TA, two parties utilize cooperation twice using SMC-SED. Alice and Bob can send the initial data to the cloud server once. In addition, the cloud server delivers the results to Bob when finishing its computation. In Step 2, Bob performs the normal addition operation 3 times and the normal multiplication operation 2 times. In Step 3, Alice performs a fourth-order matrix operation one time, the normal multiplication operation 3 times and the exponentiation operation one time. If we ignore the ordinary operations, the computational complexity of our protocol is \( 11M_N + 3M_E + 2M_M \). In SMC-TV, Bob computes the square of the Euclidean distance 3 times. Alice performs a fifth-order matrix operation one time, the common multiplication operation 3 times and the common exponentiation operation one time. Thus, the computational complexity of SMC-TV is \( 15M_N + 6M_E + 4M_M \). The computational complexity of SMC-DPL is the same as that of SMC-TA.

Communication complexity: In SMC-TA, Bob can keep the results of the distance calculation and send them \( L_2^{b_i,b_j} \) to Alice in Step 2. Alice sends \( S_{AB}^{b_i,b_j} \) to Bob in Step 3. Thus, there are 2 rounds between Alice and Bob in our protocol. In the same way, in SMC-TV, the communication between two participants consists of 2 rounds. The communication complexity of SMC-DPL is the same as that of SMC-TA.

#### Comparison

This section provides comparisons of the complexity and performance of this protocol with the schemes in references [6] and [25]. We show the comparison of the complexity of our protocol with that of [6] in Table 4. In Table 5, we compare the performance of our protocol with the protocols in [6] and [25].

Complexity: Because the spatial location problems involved in the literature are not identical, to perform the comparison, we chose the volume protocol in [6] to make a comparison. In

---

**Table 4. Complexity comparison between our scheme with the scheme in [6].**

| Problem       | Protocol | Computational Complexity | Communication between parties |
|---------------|----------|--------------------------|-------------------------------|
| Triangular Area | Protocol 2 | \( 11M_N + 3M_E + 2M_M \) | 2                             |
| Tetrahedral Volume | Protocol 3 | \( 15M_N + 6M_E + 3M_M \) | 2                             |
|               | Ref. [6]  | 9M                       | 2                             |

[https://doi.org/10.1371/journal.pone.0217067.t004](https://doi.org/10.1371/journal.pone.0217067.t004)

**Table 5. Performance comparison of our scheme and schemes in [6, 25].**

| Protocol | Triangular Area | Tetrahedral Volume | Point-Line Distance | Point-Plane Distance | Line-Plane Relationship | Plane-Plane Relationship | Cloud Platform |
|----------|----------------|--------------------|---------------------|----------------------|------------------------|-------------------------|----------------|
| Ref. [6] | N              | Y                  | N                   | Y                    | Y                      | Y                       | N              |
| Ref. [25]| N              | N                  | Y                   | Y                    | N                      | N                       | Y              |
| Ours     | Y              | Y                  | Y                   | Y                    | Y                      | Y                       | Y              |

[https://doi.org/10.1371/journal.pone.0217067.t005](https://doi.org/10.1371/journal.pone.0217067.t005)
[6], Bob performs third-order matrix operations 4 times, and Alice performs common multiplication calculations 5 times. Thus, the computational complexity is $9M$, where $M$ represents the number of common multiplications. The computational complexity can be neglected. Our schemes use a homomorphic cipher algorithm to protect the user’s private information, and the encryption and decryption calculations need more time. From Table 4, we can see that our schemes have the same communication complexity as the protocol in [6]; the computational complexity of our schemes is not optimal.

Performance: In [6], Li et al. studied security geometry problems such as tetrahedral volumes, point-line distances, line-plane relationships and plane-plane relationships. However, the schemes in [6] did not concern problems about triangular areas and point-line distances. In [25], the protocol was extended with the help of a cloud platform. However, the protocol can only solve the point-line distance problem and the point-plane distance problem. In contrast, this paper solves the above six problems, i.e. the triangular area, tetrahedral volume, point-line distance, point-plane distance, line-plane relationship and plane-plane relationship problems, in the same way with the cloud platform. From Table 5, we can see that the solutions in [25] have the worst performance, and their application is limited. The method in [6] is not suitable for cloud platforms. Our schemes achieve the best performance.

From the above, the complexity of our solutions is not the most satisfactory, but our schemes represent a new technique for solving secure multi-party solid geometry computation problems and can be used to solve a wider range of problems while maintaining the same level of security and being more universal.

Application

To illustrate our motivation in developing these solutions, we present the following interesting scenario inspired by [6]: Bob has a mixture $\xi_1$ that contains 10% of component $M_1$, 25% of component $M_2$ and 10% of component $M_3$ and another mixture $\xi_2$ that contains 2% of component $M_1$, 11% of component $M_2$ and 4% of component $M_3$. Additionally, assume that Alice needs a mixture that contains 6% of component $M_1$, 20% of component $M_2$ and 3% of component $M_3$. Alice wants to know if she can produce this mixture from the two mixtures that Bob has, but she does not want to disclose her needs. Similarly, Bob does not want to disclose the contents of his mixtures. If we represent the mixtures $\xi_1$ and $\xi_2$ by points in three-dimensional space, namely, by $P_{B_1}, P_{B_2}$ (see Fig 3), we can produce the mixtures represented by any point on the line segment $P_{B_1}, P_{B_2}$ by mixing $\xi_1$ and $\xi_2$ at various ratios. Thus, privately determining whether Alice can produce her mixture from Bob’s two mixtures can be reduced to privately determining whether the point that represents Alice’s mixture is on the line segment $P_{B_1}, P_{B_2}$. This computational geometry problem is called the point-inclusion problem.

How does one solve this problem? First, Alice and Bob use Protocol 1 to privately compute $L^2_{AB_1}$ and $L^2_{AB_2}$, and they use Protocol 2 to privately compute $S_{AB_1B_2}$. Then, Alice and Bob use Protocol 4 to privately determine $d$, the distance between point $P_A$, which represents Alice’s mixture in three-dimensional space, and the line $P_{B_1}P_{B_2}$, determined by the two points that represent Bob’s two mixtures. If $d \neq 0$, Alice cannot produce her mixture from the two mixtures that Bob has; otherwise, if $d = 0$, which implies that point $P_A$ is on the line determined by the two points that represent Alice’s two mixtures, then Alice and Bob can negotiate to project these points onto a line and to privately determine whether the projection of $P_A$ is inside the projection of the line segment $P_{B_1}P_{B_2}$. If the projection of $P_A$ is inside that of line segment $P_{B_1}P_{B_2}$, Alice can produce her mixture from Bob’s two mixtures; otherwise, Alice cannot
produce her mixture from his mixtures. In this way, Alice and Bob can solve this problem while keeping their mixtures private.

If Alice has three mixtures, Alice and Bob first use Protocol 1 to privately compute $L_{AB}^1$, $L_{AB}^2$, and $L_{AB}^3$, and they use Protocol 3 to privately compute $V_{AB, B_1, B_2}$. Then, Alice and Bob use Protocol 2 in [6] to determine the distance between point $P_A$, which represents Alice’s mixture in three-dimensional space, and the line $P_{B_1}$, $P_{B_2}$, and $P_{B_3}$ is determined by the three points that represent Bob’s three mixtures (see Fig 3). If there are more than three components of interest in the mixtures, a similar analysis can be performed using a higher dimensional space.

Conclusion

With the emergence and widespread use of cloud computing, solving the problem of multi-party computing with cloud platforms has become a new research direction. The introduction of cloud computing resources has lead to changes in the secure multi-party computing model and solutions. However, the computational task of spatial geometry calculations in traditional problems is completed via mutual interactions between two parties. Therefore, it is difficult for these protocols to be used in untrusted cloud computing environments. In this paper, we propose a more general solution to solve spatial geometry security problems using a cloud platform. First, we transform the problems into the calculation of a distance. Then, we design a security protocol to solve for the Euclidean distance. Based on the above protocol, we solve two problems concerning the calculations of triangular areas and tetrahedron volumes. We prove that our protocols can resist collusion between the parties and the untrusted cloud platform, and it effectively protects the users’ privacy. In addition, we note that the proposed protocol can be used for the calculation of spatial distances such as point-line, point-surface, and line-line distance. Because of the transformation of the problems, we can solve more spatial geometry security problems with a cloud platform. However, the complexity of the protocol should be improved. Therefore, as future work, we will attempt to reduce the computational complexity of the protocol.

Author Contributions

Writing – original draft: Jing Zhang, Lixiang Li, Yongli Tang, Shoushan Luo, Yixian Yang, Yang Xin.

Writing – review & editing: Jing Zhang, Lixiang Li, Yongli Tang, Shoushan Luo, Yixian Yang, Yang Xin.

References

1. Yao ACC. Protocols for secure computations. In: FOCS. vol. 82; 1982. p. 160–164.
2. Goldreich O, Micali S, Wigderson A. How to play any mental game. In: Proceedings of the nineteenth annual ACM symposium on Theory of computing. ACM; 1987. p. 218–229.
3. Goldreich O. Secure multi-party computation. Manuscript Preliminary version. 1998; 78.
4. Neugebauer G, Brutschy L, Meyer U, Wetzel S. Privacy-preserving multi-party reconciliation secure in the malicious model. In: Data Privacy Management and Autonomous Spontaneous Security. Springer; 2013. p. 178–193.
5. Bogdanov D, Laur S, Talviste R. A practical analysis of oblivious sorting algorithms for secure multi-party computation. In: Nordic Conference on Secure IT Systems. Springer; 2014. p. 59–74. https://doi.org/10.1007/978-3-319-11599-3_4
6. Shundong L, Chunying W, Daoshun W, Yiqi D. Secure multiparty computation of solid geometric problems and their applications. Information Sciences. 2014; 282:401–413. https://doi.org/10.1016/j.ins.2014.04.004
7. Ranbaduge T, Christen P, Vatsalan D. Tree based scalable indexing for multi-party privacy-preserving record linkage. 2014;

8. Hazay C. Oblivious polynomial evaluation and secure set-intersection from algebraic PRFs. Journal of Cryptology. 2018; 31(2):537–586. https://doi.org/10.1007/s00145-017-9263-y

9. Liu L, Chen X, Lou W. Secure three-party computational protocols for triangle area. International Journal of Information Security. 2016; 15(1):1–13.

10. Nair DG, Binu V, Kumar GS. An improved e-voting scheme using secret sharing based secure multi-party computation. arXiv preprint arXiv:150207469. 2015;

11. Guanciale R, Gurov D, Laud P. Business process engineering and secure multiparty computation. Cryptology and Information Security. 2015; 13:129–149.

12. Aly A, Van Vyve M. Securely solving classical network flow problems. In: International Conference on Information Security and Cryptology. Springer; 2014. p. 205–221.

13. Marwan M, Kartit A, Ouahmane H. Applying secure multi-party computation to improve collaboration in healthcare cloud. In: 2016 Third International Conference on Systems of Collaboration (SysCo). IEEE; 2016. p. 1–6.

14. Maheshwari N, Kiyawat K. Structural framing of protocol for secure multiparty cloud computation. In: 2011 Fifth Asia Modelling Symposium. IEEE; 2011. p. 187–192.

15. Kamara S, Mohassel P, Raykova M. Outsourcing Multi-Party Computation. IACR Cryptology ePrint Archive. 2011; 2011:272.

16. Kamara S, Mohassel P, Riva B. Salus: a system for server-aided secure function evaluation. In: Proceedings of the 2012 ACM conference on Computer and communications security. ACM; 2012. p. 797–808.

17. Carter H, Mood B, Traynor P, Butler K. Secure Outsourced Garbled Circuit Evaluation for Mobile Devices. In: Presented as part of the 22nd (USENIX) Security Symposium ((USENIX) Security 13); 2013. p. 289–304.

18. Asharov G, Jain A, López-Alt A, Tromer E, Vaikuntanathan V, Wichs D. Multiparty computation with low communication, computation and interaction via threshold FHE. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer; 2012. p. 483–501.

19. López-Alt A, Tromer E, Vaikuntanathan V. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the forty-fourth annual ACM symposium on Theory of computing. ACM; 2012. p. 1219–1234.

20. Kerschbaum F. Collusion-resistant outsourcing of private set intersection. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing. ACM; 2012. p. 1451–1456.

21. Kamara S, Mohassel P, Raykova M, Sadeghian S. Scaling private set intersection to billion-element sets. In: International Conference on Financial Cryptography and Data Security. Springer; 2014. p. 195–215.

22. Abadi A, Terzis S, Dong C. O-psi: delegated private set intersection on outsourced datasets. In: IFIP International Information Security and Privacy Conference. Springer; 2015. p. 3–17.

23. Veugen T, de Haan R, Cramer R, Muller F. A framework for secure computations with two non-colluding servers and multiple clients, applied to recommendations. IEEE Transactions on Information Forensics and Security. 2015; 10(3):445–457. https://doi.org/10.1109/TIFS.2014.2370255

24. Chen Z, Li S, Wang D, Huang Q, Zhang W. Secure MultiParty Computation of Set Membership and Its Applications. Tien Tzu Hsueh Pao/acta Electronica Sinica. 2017; 45(5):1109–1116.

25. Chen ZH, Li SD, Huang Q, Ding Y, Sun. Privacy-preserving determination of spatial location relation in cloud computing. Chin J Comput. 2017; 40(2):351–363.

26. Boneh D, Goh EJ, Nissim K. Evaluating 2-DNF formulas on ciphertexts. In: Theory of Cryptography Conference. Springer; 2005. p. 325–341.

27. Thomas F, Ros L. Revisiting trilateration for robot localization. IEEE Transactions on robotics. 2005; 21 (1):93–101. https://doi.org/10.1109/TRO.2004.833793

28. Amirbekyan A, Estivill-Castro V. A new efficient privacy-preserving scalar product protocol. In: Proceedings of the sixth Australasian conference on Data mining and analytics-Volume 70. Australian Computer Society, Inc.; 2007. p. 209–214.

29. Rane S, Sun W, Vetro A. Privacy-preserving approximation of L1 distance for multimedia applications. In: 2010 IEEE International Conference on Multimedia and Expo. IEEE; 2010. p. 492–497.

30. Huang H, Gong T, Chen P, Malekian R, Chen T. Secure two-party distance computation protocol based on privacy homomorphism and scalar product in wireless sensor networks. Tsinghua Science and Technology. 2016; 21(4):385–396. https://doi.org/10.1016/S1007-7536(16)60024-7