Research Article

Privacy Preserving Inner Product of Vectors in Cloud Computing

Gang Sheng,1 Tao Wen,1 Quan Guo,2 and Ying Yin3

1 Software Center, Northeastern University, Shenyang 110819, China
2 Department of Computer Science and Technology, Dalian Neusoft University of Information, Dalian 116023, China
3 College of Information Science and Engineering, Northeastern University, Shenyang 110819, China

Correspondence should be addressed to Gang Sheng; shenggang@neusoft.edu.cn

Received 11 January 2014; Revised 9 April 2014; Accepted 23 April 2014; Published 25 May 2014

Copyright © 2014 Gang Sheng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The problem of privacy preserving inner product of vectors has been widely studied. Much work has been done on the scenario of two parties involved in the computation. In this paper, we consider the scenario where three parties are involved in the computing process of cloud computing. We propose a new privacy preserving scheme for inner product of two vectors in the cloud and give the correctness analysis and performance analysis for the scheme. The proposed scheme is based on homomorphic encryption, and the security can be guaranteed. Experiments show the efficiency of the scheme.

1. Introduction

The computation of inner product of vectors is a fundamental mathematical operation. The inner product of vectors has found wide application in the research field of computer science, such as cooperative statistical analysis [1], data mining [2], and similarity computation [3]. The model depicted in Figure 1 is adopted in much work, where two parties, that is, the server and the client, are involved. The server owns a large amount of data, and the client may issue query to the server. In Model 1, the problem of data privacy preservation is one of the most important security issues. The problem of privacy preserving inner product of vectors has gained thorough research, and much work has been done on it, such as [1–7].

When the server possesses particularly large amounts of data or the amount of query from the clients is considerably abundant, the server may be overburdened. Now, the cloud computing has gained wide research and many applications have been deployed in the cloud. Three parties are generally involved in the cloud computing model [8]. The existing work on privacy preserving inner product of vectors between two parties cannot work well any longer.

A secure privacy preserving scheme was presented for inner product of vectors in cloud computing in [9]. The scheme is effective and achieves the goal of data privacy preservation. However, it is the array that acts as the shared secret key between the data owner and the client. The array may be large when the vector has big size of dimensionality. It is inconvenient for the data owner to update the secret key when there are many clients.

We propose a new scheme on the basis of homomorphic encryption, which aims at the problem of privacy preserving inner product of vectors in cloud computing. Our proposed scheme can work seamlessly with the other secure scheme of inner product of vectors, such as correctness verification of inner product of vectors [10].

The rest of this paper is organized as follows. Section 2 describes the background knowledge, including the inner product of vectors, homomorphic encryption, and the system model and threat model. Section 3 summarizes the existing works on privacy preserving inner product of vectors. Section 4 introduces our proposed scheme for privacy preserving inner product of vectors in the cloud. Following that, experimental results are given in Section 5. Finally, Section 6 concludes the paper.
2. Background

2.1. Inner Product of Vectors. Let \( u = (u_1, u_2, \ldots, u_n) \) and \( v = (v_1, v_2, \ldots, v_n) \) be two vectors, where \( n \) is a positive integer and denotes the dimensionality of the vectors, and \( u_i \) and \( v_i \) (\( 1 \leq i \leq n \)) are vector elements. The inner product of \( u \) and \( v \) is the sum of the products of the corresponding elements; that is, \( u \cdot v = u_1v_1 + u_2v_2 + \cdots + u_nv_n = \sum_{i=1}^{n} u_iv_i \), where the symbol “\( \cdot \)” denotes the mathematical operation of computing the inner product between two vectors.

2.2. Homomorphic Encryption. Let \((pk, sk) \) be a key pair for a public key encryption scheme, where \( pk \) is the public key for encrypting data and \( sk \) is the secret key for decrypting. An encryption scheme is called homomorphic if it holds the following two properties simultaneously:

\[
E_{pk}(x_1) \oplus E_{pk}(x_2) = E_{pk}(x_1 + x_2),
\]

\[
(E_{pk}(x))^c = E_{pk}(cx),
\]

where \( x_1, x_2, \) and \( x \) are plain data, \( E_{pk}(x_1), E_{pk}(x_2) \), and \( E_{pk}(x) \) are the ciphertext of \( x_1, x_2, \) and \( x \) with \( pk \) as the encryption key, respectively, and \( \oplus \) is an operation on the ciphertext and \( c \) is a positive integer.

For convenience, we introduce a symbol \( \text{Sum} \), which denotes the result of operation \( \oplus \) on multiple homomorphic ciphertexts. Sum is defined as

\[
\text{Sum}_{i=1}^{n} x_i = E_{pk}(x_1) \oplus E_{pk}(x_2) \oplus \cdots \oplus E_{pk}(x_n).
\]

The Paillier’s homomorphic encryption is adopted in the following [11].

2.3. System and Threat Models. A simplified architecture of cloud computing is given in Figure 2. Three parties are involved in Model 2, that is, the data owner, the service provider, and the client. The data owner holds large quantities of data. The data owner outsources the heavy data management task to the service provider to reduce the total cost. The service provider is a specific cloud service provider, which possesses rich resources, including hardware, software, and network. The service provider provides the users with the services of storage, computation, and so forth. The authorized client may issue a request to the service provider. Here, the user may be a company, an organization, or an individual, who makes use of the cloud service. The data owner and the client are both users. The client is a specific user, which refers to the party that can only issue query to the service provider.

In the above model, the data owner, who is the owner of the data stored on the service provider side, is honest. The service provider and the client are both semihonest. The service provider attempts to obtain the original data of the data owner and the client. The client attempts to get the original data of the data owner. Suppose that the data are transferred between the three parties through secure channel.

3. Related Works

A privacy preserving scheme of inner product of vectors was proposed for privacy preserving data mining between two parties in [2], in which the matrix multiplication operation was adopted. The privacy of the input data and the obtained result was preserved simultaneously. A state-of-the-art overview of the privacy preserving inner product of vectors problem and the properties of some solutions were given in [4], where the scheme proposed in [2] was analyzed and proved to be not secure, and a new scheme was presented based on the homomorphic encryption, which has strong security guarantee.

For the purpose of detecting similar documents as well as preserving data privacy between two parties, two privacy preserving schemes for inner product of vectors, that is, the random array scheme and the homomorphic encryption scheme, were utilized in [3]. Privacy preserving inner product of vectors was applied to determine the relative position of two spatial geometric objects in [12].

An efficient and secure inner product protocol in the presence of malicious adversaries was presented in [6]. The protocol was based on the proof of knowledge of a discrete logarithm and the verifiable encryption.

Secure two-party inner product of vectors was extended to the quantum field in [7]. The given protocol was built upon quantum entanglement and quantum measurement.

The problem of correctness verification of inner product of vectors was studied in the cloud computing scenario in [9], in which the array multiplication operation was adopted to preserve the data privacy.
4. The Proposed Scheme

Suppose the data owner holds a vector \( v = (v_1, v_2, \ldots, v_n) \) and the request vector of the client is \( u = (u_1, u_2, \ldots, u_n) \), where \( n \) is a positive integer and denotes the dimensionality of the vectors. Our proposed privacy preserving scheme for inner product of vectors in the cloud consists of five algorithms, that is, KeyGen, Encrypt, Request, Compute, and GetResult.

4.1. Overview. In our scheme, three parties are involved in the computation. Data privacy of each party needs to be preserved. The privacy preserving scheme for inner product of vectors in the cloud scenario works as follows.

1. **KeyGen.** First of all, the data owner and the client generate the necessary keys, respectively, and share the keys with the related party.

2. **Encrypt.** The data owner encrypts the original vector to obtain the ciphertext and sends it to the service provider. The service provider receives the data from the data owner and monitors the query request from the client.

3. **Request.** The client encrypts the original request vector to get the transformed vector and sends it to the service provider.

4. **Compute.** The service provider performs computing on the ciphertext from the data owner and the client and obtains the intermediate results and sends them back to the client.

5. **GetResult.** The client computes the returned results to get the final result.

4.2. Details. In this section, we describe the design details of each algorithm.

1. **KeyGen.** Whatever encryption scheme or data processing algorithm is chosen, the key(s) is (are) necessary. The keys are generated on the data owner side and client side, respectively, which are given as follows.

   - The data owner randomly generates a key pair (pk, sk) for homomorphic encryption scheme, where pk is the public key and sk is the secret key. pk is published publicly and sk is shared with the client through secure channel.
   - The client randomly generates an \( n \times n/2 \) array \( A \) and shares it with the service provider through secure channel, where
     \[
     A = \begin{pmatrix}
     a_{1,1} & a_{1,2} & \cdots & a_{1,n/2} \\
     \vdots & \vdots & \ddots & \vdots \\
     a_{n,1} & a_{n,2} & \cdots & a_{n,n/2}
     \end{pmatrix}.
     \]

2. **Encrypt.** Aiming at privacy preservation of the data, the data owner encrypts each element \( v_i \) \((i = 1, 2, \ldots, n)\) of vector \( v \) in turn using the homomorphic encryption scheme with public key pk. The obtained ciphertext of \( v \) is denoted as \( E_{pk}(v) \), where \( E_{pk}(v) = (E_{pk}(v_1), E_{pk}(v_2), \ldots, E_{pk}(v_n)) \). Then \( E_{pk}(v) \) is sent to the service provider.

3. **Request.** Before issuing the request to the service provider, it is necessary for the client to transform the request data to preserve privacy. The data transformation on the client side works as follows.

   1. The client randomly generates a vector \( r \) with dimensionality of \( n/2 \), where \( r = (r_1, r_2, \ldots, r_{n/2}) \);
   2. compute \( z \), where
     \[
     z = u + rA^T,
     \]
     and send it to the service provider, where \( A^T \) denotes the transpose of array \( A \).

Let \( z \) be denoted as \( z = (z_1, z_2, \ldots, z_n) \). Then, we have

\[
z_i = u_i + \sum_{j=1}^{n/2} a_{i,j}, \quad i = 1, 2, \ldots, n.
\]

It is vector \( z \) that is sent to the service provider, which cannot reveal the value of \( u \). Though the service provider holds \( A \) and \( z \), it cannot detect \( u \) and \( r \) via solving the system of linear equations. Thus, the data privacy of the client is preserved through mathematical transformation.

4. **Compute.** Upon receiving the request \( z \) from the client, the service provider starts to compute \( S \) and \( y \). \( S \) is an intermediate result and contains the final result, where

\[
S = (E_{pk}(v_1))^z_1 \oplus (E_{pk}(v_2))^z_2 \oplus \cdots \oplus (E_{pk}(v_n))^z_n
= \sum_{i=1}^{n} (E_{pk}(v_i))^{z_i}.
\]

\( y \) is a vector with dimensionality of \( n/2 \), which is used to compute the final result on the client side. Let \( y = (y_1, y_2, \ldots, y_{n/2}) \), where, for \( i = 1, 2, \ldots, n/2 \),

\[
y_i = (E_{pk}(v_1))^{a_{i,1}} \oplus (E_{pk}(v_2))^{a_{i,2}} \oplus \cdots \oplus (E_{pk}(v_n))^{a_{i,n}}
= \sum_{j=1}^{n} (E_{pk}(v_j))^{a_{ij}}.
\]

Then, the service provider responds to the client with \( S \) and \( y \).

5. **GetResult.** Based on the returned results \( S \) and \( y \) from the service provider, the client first computes

\[
S' = y_1^{r_1} \oplus y_2^{r_2} \oplus \cdots \oplus y_{n/2}^{r_{n/2}} = \sum_{i=1}^{n/2} (y_i)^{r_i}.
\]

Then, the client computes

\[
D_{pr}(S) - D_{pr}(S'),
\]
which is just the inner product of \( u \) and \( v \), where \( D_{pr}(S) \) denotes decrypting \( S \) with secret key sk.

Thus, the final result is obtained; that is,

\[
u \cdot v = D_{pr}(S) - D_{pr}(S').
\]
4.3. Correctness Analysis. In this section, we illustrate the correctness of our proposed scheme.

In the algorithm Compute, the service provider obtains $S$ as the intermediate result. By (1) and (6), we have

$$S = \sum_{i=1}^{n} \left( E_{pk}(v_i) \right)^{z_i} = \sum_{i=1}^{n} E_{pk}(v_i z_i)$$

(11)

By (5) and (11), we have

$$S = E_{pk} \left( \sum_{i=1}^{n} v_i z_i \right)$$

$$= E_{pk} \left( \sum_{i=1}^{n} v_i \left( u_i + \sum_{j=1}^{n/2} r_j a_{i,j} \right) \right)$$

(12)

$$= E_{pk} \left( \sum_{i=1}^{n} v_i u_i + v \sum_{j=1}^{n/2} r_j a_{i,j} \right)$$

$$= E_{pk} \left( \sum_{i=1}^{n} v_i u_i + \sum_{i=1}^{n/2} r_j a_{i,j} \right)$$

$$= E_{pk} \left( \sum_{i=1}^{n} v_i u_i + \sum_{i=1}^{n/2} r_j a_{i,j} \right) .$$

Along with $S$, the service provider obtains a vector $y = (y_1, y_2, \ldots, y_{n/2})$ in the algorithm Compute. By (1) and (7), for $i = 1, 2, \ldots, n/2$, we have

$$y_i = \sum_{j=1}^{n} \left( E_{pk}(v_j) \right)^{a_{i,j}} = E_{pk} \left( \sum_{j=1}^{n} v_j a_{i,j} \right) .$$

(13)

In the algorithm GetResult, the client computes and gets $S' = \sum_{i=1}^{n/2} \left( y_i \right)^{r_i}$. Then, by (13), we have

$$S' = \sum_{i=1}^{n/2} \left( y_i \right)^{r_i}$$

$$= \sum_{i=1}^{n/2} \left( E_{pk} \left( \sum_{j=1}^{n} v_j a_{i,j} \right) \right)^{r_i}$$

(14)

$$= \sum_{i=1}^{n/2} \left( E_{pk} \left( r_i \sum_{j=1}^{n} v_j a_{i,j} \right) \right)$$

$$= \sum_{i=1}^{n/2} \left( E_{pk} \left( \sum_{j=1}^{n} r_i v_j a_{i,j} \right) \right)$$

$$= E_{pk} \left( \sum_{i=1}^{n/2} \sum_{j=1}^{n} r_i v_j a_{i,j} \right) .$$

Thus, in the algorithm GetResult, when the client computes $D_{pr}(S) = D_{pr}(S')$, by (12) and (14), we have

$$D_{pr}(S) = D_{pr}(S')$$

$$= \sum_{i=1}^{n} v_i u_i + \sum_{i=1}^{n/2} r_i a_{i,j} - \sum_{i=1}^{n/2} r_i v_j a_{i,j}$$

(15)

$$= \sum_{i=1}^{n} v_i u_i = u \cdot v .$$

In the end, the client obtains the deserved result $u \cdot v$, which is the inner product of $u$ and $v$.

4.4. Performance Analysis. In this section, we give the performance analysis of each algorithm.

1. In the algorithm Encrypt, each vector element $v_i$ ($1 \leq i \leq n$) is encrypted in turn on the data owner side. The time complexity of algorithm Encrypt is $O(n)$.

2. In the algorithm Request, the original user request vector $u$ is transformed into $z$ on the client side, where matrix multiplication is used. The time complexity of algorithm Request is $O(n^2)$.

3. In the algorithm Compute, $S$ and $y$ are obtained on the service provider side, where $S = \sum_{i=1}^{n} \left( E_{pk}(v_i) \right)^{z_i}$, and $y = (y_1, y_2, \ldots, y_{n/2})$ with $y_i = \sum_{j=1}^{n} \left( E_{pk}(v_j) \right)^{a_{i,j}}$. The time complexity of algorithm Compute is $O(n^2)$.

4. In the algorithm GetResult, $S'$ is obtained by $y$ and $r$, where $S' = \sum_{i=1}^{n/2} \left( y_i \right)^{r_i}$. The time complexity of algorithm GetResult is $O(n)$.

It is worth mentioning especially that magnitude of the time complexity of algorithm Request and that of Compute are different. Homomorphic encryption is performed on exponents in the algorithm Encrypt, and multiplication and addition are performed on integers in the algorithm Request. The actual time cost is quite different, though the time complexity of both Request and Compute is $O(n^2)$.

4.5. Security Analysis. In this subsection, we give the security analysis, which is presented from three aspects, that is, data privacy preservation of the data owner, the client, and the result.

Data Privacy of the Data Owner. Because the data on the data owner side are encrypted by homomorphic encryption scheme, the service provider can detect nothing about the original data. Then, the data privacy of the data owner is preserved with security guarantee.

Data Privacy of the Client. The query issued by the client is transformed via matrix operation, and the service provider gets $z = u + rA^T$. The service provider holds $z$ and $A$, and
it may attempt to detect $u$ by solving the following system of linear equations:

\[
\begin{align*}
    z_1 &= u_1 + r_1 a_{1,1} + r_2 a_{1,2} + \cdots + r_{n/2} a_{1,n/2} \\
    z_2 &= u_2 + r_1 a_{2,1} + r_2 a_{2,2} + \cdots + r_{n/2} a_{2,n/2} \\
    &\quad \cdots \quad \cdots \quad \cdots \\
    z_n &= u_n + r_1 a_{n,1} + r_2 a_{n,2} + \cdots + r_{n/2} a_{n,n/2}.
\end{align*}
\]

(16)

In (16), there are $n$ equations and $n + n/2$ unknown variables, and it is impossible for the service provider to solve the system of linear equations to get the value of $u$. Thus, the data privacy of the client is preserved.

**Data Privacy of the Result.** In our proposed scheme, the result of inner product of vectors is not given directly. The result is given by the intermediates $S$ and $y$, which are both encrypted by homomorphic encryption scheme. The security of $S$ and $y$ can be guaranteed by the homomorphic encryption scheme. Thus, the data privacy of the result is preserved.

5. Experiments

We now evaluate the performance overhead of our proposed scheme for privacy preserving inner product of vectors in cloud computing. We use a Pentium Dual E2200 2.20 GHz PC with 1.99 GB RAM. Code is developed in C using the MIRACL library. Synthetic 5-dimensional data are used, which are integers generated randomly within $[1,10000000]$ with even distribution. Four algorithms of Encrypt, Request, Compute, and GetResult are implemented respectively, whose running results on the given data are shown in Figure 3.

As can be seen in Figure 3, the performance overhead of each algorithm is consistent with the performance analysis in Section 4.4. Homomorphic encryption is utilized in the algorithms of Encrypt, Compute, and GetResult, where the operation is performed on exponents. The actual time cost of the algorithm Compute is the highest, and that of the algorithm Encrypt is the second among the four algorithms, which is shown in Figure 3(a).

In Section 4.4, we have mentioned that the actual time cost of algorithm Request and Compute is quite different, though the time complexity of the two algorithms is $O(n^2)$. Figure 3(a) demonstrates that is true. The actual time cost of the algorithm Compute is far higher than that of the algorithm Request.

The actual time cost of the algorithms Request and GetResult is so small compared with the algorithms Encrypt and Compute that they cannot be distinguished from each other clearly. The performance of the algorithms Request and GetResult is shown again in Figure 3(b), where it demonstrates that the time cost of the algorithm Request is higher.

6. Conclusion

Aiming at the problem of privacy preservation of inner product of vectors in cloud computing, a new scheme is presented with the homomorphic encryption as the fundamentals. The scheme is applicable for the cloud scenario where three parties are involved in the computation.

In future work, we plan to consider the case where the service provider of the cloud is malicious. Another interesting direction is access control of the client when there are multiple clients with different access permissions.
Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 61170169, 61170168, 11271226, and 61100028, by the Foundation of Dalian Scientific and Technical Planning Project under Grant 2013A16GXI15, by the Science Foundation of Liaoning Education Ministry under Grant L2013517, and by the Natural Science Foundation of Shandong Province under Grants ZR2012AL07 and ZR2013AM013.

References

[1] W. Du and M. J. Atallah, “Privacy-preserving cooperative statistical analysis,” in Proceedings of the 17th Annual Conference on Computer Security Applications, pp. 102–110, IEEE, 2001.

[2] J. Vaidya and C. Clifton, ”Privacy preserving association rule mining in vertically partitioned data,” in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639–644, ACM, July 2002.

[3] M. Murugesan, W. Jiang, C. Clifton, L. Si, and J. Vaidya, “Efficient privacy-preserving similar document detection,” VLDB Journal, vol. 19, no. 4, pp. 457–475, 2010.

[4] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikäinen, “On private scalar product computation for privacy-preserving data mining,” in Proceedings of the 7th International Conference on Information Security and Cryptology (ICISC ’04), pp. 104–120, Springer, 2005.

[5] A. Amirbekyan and V. Estivill-Castro, “A new efficient privacy-preserving scalar product protocol,” in Proceedings of the 6th Australasian Conference on Data Mining and Analytics, pp. 209–214, Australian Computer Society, 2007.

[6] B. Yang, Y. Yu, and C. H. Yang, ”A secure scalar product protocol against malicious adversaries,” Journal of Computer Science and Technology, vol. 28, no. 1, pp. 152–158, 2013.

[7] L. B. He, L. S. Huang, W. Yang, and R. Xu, “A protocol for the secure two-party quantum scalar product,” Physics Letters A, vol. 376, no. 16, pp. 1323–1327, 2012.

[8] M. D. Ryan, “Cloud computing security: the scientific challenge, and a survey of solutions,” Journal of Systems and Software, vol. 86, no. 9, pp. 2263–2268, 2013.

[9] G. Sheng, T. Wen, Q. Guo, and Y. Yin, "Secure scalar product computation of vectors in cloud computing," Journal of Northeastern University, vol. 34, no. 6, pp. 786–791, 2013.

[10] G. Sheng, T. Wen, Q. Guo, and Y. Yin, "Verifying correctness of inner product of vectors in cloud computing," in Proceedings of the 2013 International Workshop on Security in Cloud Computing, pp. 61–68, ACM, 2013.

[11] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in Proceedings of Advances in Cryptology (EUROCRYPT ’99), pp. 223–238, Springer, 1999.

[12] Y. Luo, L. Huang, W. Jing, and W. Xu, “Privacy protection in the relative position determination for two spatial geometric objects,” Journal of Computer Research and Development, vol. 43, no. 3, pp. 410–416, 2006.
Submit your manuscripts at
http://www.hindawi.com