Cloud Database Encryption Technology Based on Combinatorial Encryption Meng Chen, Xiao Fu*, Bin Luo State Key Laboratory for Novel Software Technology Nanjing University, Nanjing, China mf1832019@smail.nju.edu.cn, {fuxiao, luobin}@nju.edu.cn Xiaojiang Du Dept. of Computer and Information Sciences Temple University Philadelphia, USA dxj@ieee.org Mohsen Guizani Dept. of Computer Science and Engineering Qatar University Qatar mguizani@ieee.org RELATED WORK Abstract—Traditional cloud database directly stores user plaintext information, information security is directly related to the security of Cloud server, which will create a great security risk. The user's information security is not guaranteed. In this paper, a database encryption technology is designed, which can balance the problem of information security and use efficiency well. In this paper, a simplified onion encryption model is designed and implemented, which can realize the full homomorphic encryption on the cloud database to a certain extent, and improve the efficiency of ciphertext operation to a certain extent. Keywords—cloud; database; full homomorphic encryption I. INTRODUCTION Cloud database is a cloud service mode formed under the concept of cloud computing software-as-a-service, which expands the overall storage capacity by changing the organization mode of the database server. Server software and hardware maintenance are managed by professional operation and maintenance personnel of cloud service providers to ensure service reliability. There is a big difference between data protection in a cloud environment and data protection in the traditional mode. In traditional mode, the data is stored on the user's servers, which are generally trusted. However, in the cloud environment, users outsource their data to the cloud service providers. The data security of users is directly guaranteed by the cloud service provider, which will have security risks [1]. Aiming at the hidden danger of data security in the cloud Database system, this paper focuses on the encryption algorithm which supports ciphertext operation. The main design of an encrypted middle layer, while ensuring the security of the cloud database, can reduce the pressure of network transmission and client decryption, the greatest extent to take advantage of the advantages of cloud computing [2]. Our framework composed of three parts. The first part is the application layer. The main features are generating keys, updating ciphertext, and sending SQL requests. The second part is the proxy layer. The main function is to override the SQL statement. The third part is the DBMS layer. The main function is to store ciphertext in the database [3]. II. 978-1-7281-0962-6/19/$31.00 2019 IEEE Authorized licensed use limited to: KING ABDUL AZIZ UNIVERSITY. Downloaded on August 26,2020 at 15:27:20 UTC from IEEE Xplore. Restrictions apply. The previous encryption algorithm was to encrypt plaintext directly with a combinatorial encryption algorithm and then store the ciphertext in the database. The ciphertext needs to be updated when different operations are performed. The advantage of this approach is that it is simple and straightforward, with the disadvantage that multiple ciphertexts need to be updated, which will result in greater network transmission pressure [4]. Full-state encryption is the best homomorphic encryption method in theory. The full homomorphic encryption technology can perform operations on the cipher without decrypting the data, and the results are consistent with the results after the corresponding plaintext operation. However, the fully homomorphic encryption technology is not mature, the key is huge, and the encryption efficiency is very low, can not be put into practical application [5]. Several papers (e.g., [7-11]) have studied related cloud and other security issues. In this paper, an improved combinatorial encryption algorithm is proposed. On the basis of the traditional combinatorial encryption algorithm, a layer of proxy layer is added, and by rewriting the SQL statement, the same operation is completed as much as possible at one time, in order to reduce the number of ciphertext updates and reduce the network transmission between the application layer and the proxy layer [6]. III. OVERVIEW This project uses symmetric encryption algorithm AES, asymmetric algorithm RSA, Paillier and keep–ordered encryption algorithm OPE to encrypt information. Different encryption strategies are used for different types of data. The AES encryption algorithm can compare the equivalent of ciphertext information, and the RSA encryption algorithm satisfies the multiplication homomorphism, that is, the result of the multiplication of ciphertext is equal to the ciphertext of plaintext multiplication. The AES encryption algorithm satisfies the additive homomorphism; that is, the result of the addition of ciphertext is equal to the cipher of plaintext addition. Using the characteristics of different encryption algorithms, some SQL operations can be carried out on the ciphertext stored in the database. For security reasons, the database does not store the encryption key, the decryption work is carried out entirely by the client, and the cloud server undertakes the work of the ciphertext operation to the maximum extent. This approach not only plays the computing power and storage ability of the cloud but also avoids the data security problem on the cloud to the greatest extent. This paper uses the combination of homomorphic encryption instead of full–homomorphic encryption, which will reduce the usability of the encryption algorithm but will improve the efficiency of the algorithm. This trade–off is worth it before there is a major breakthrough in the fully homomorphic encryption algorithm. The encryption hierarchy is shown in table 1.In the table1 the hierarchical structure and the functions of each layer are briefly introduced. Layer TABLE 1. ENCRYPTION LEVEL Function Application Layer 1. Generate the key for the encryption algorithm; 2. Run the application code and initiate the SQL query; 3. Updating ciphertext; Proxy Layer 1. Override SQL statements, but maintain semantics; 2. Returns the ciphertext results returned by the DBMS to the application layer; DBMS Layer 1. All data is stored encrypted; 2. Processing encrypted data, just as it does with plaintext data; The workflow for each encryption layer is shown in Figure 1. The application layer sends the SQL statement of the query to the proxy layer, and then the proxy layer overrides the SQL statement and queries to the DBMS layer. There are two purposes for overwriting SQL statements, the first is to turn clear text queries into ciphertext queries, and the second is to prioritize the same operations to reduce the number of ciphertext updates. The proxy layer then returns the calculated ciphertext to the application layer, the application layer updates the ciphertext, and the updated cipher is sent to the agent layer. The agent layer then continues with the next round of operations. Over a period of up to 30 years, there has been no breakthrough in the work of Homomorphic encryption. This is because the Homomorphic encryption algorithm must support any operation at the same time in order to become a common method, that is, the fully homomorphic encryption algorithm, but it is almost impossible to achieve this goal through a single function. A homomorphic encryption scheme that can only support a finite number operation or a single operation type is called a partial homomorphic encryption algorithm. For example, RSA, EIGamal, GoldwasserMicali, Benalo, and Paillier are partial homomorphic encryption algorithms. These algorithms can either only support multiplication homomorphism or additive homomorphism, or can only support a small number of multiplication operations on the basis of addition homomorphism, so they are not full homomorphic encryption algorithms. The full homomorphic encryption algorithm needs to be constructed by a more ingenious method. All of these "ingenious" methods have a common denominator, all of which need to be realized through the modulus operation to support the homomorphic addition and homomorphic multiplication at Authorized licensed use limited to: KING ABDUL AZIZ UNIVERSITY. Downloaded on August 26,2020 at 15:27:20 UTC from IEEE Xplore. Restrictions apply. IV. Fig.1. The query process APPLICATION LAYER the same time within a certain number of operations. However, this can have a side effect: there is always a noise in the ciphertext. And the noise will increase with the number of ciphertext operations, and eventually increase to the decryption function cannot restore the clear text correctly. In this paper, the combination of additive homomorphic encryption and multiplication homomorphic encryption is used to realize the fully homomorphic encryption to a certain extent. A. Multiplication homomorphic Encryption Scheme This paper selects RSA as the multiplication homomorphic encryption scheme. The security of the encryption scheme is based on the difficulty of large number decomposition. The specific encryption scheme is as follows: Key generation: Select two different large prime numbers p,q. Set □ = □ · □. Calculate its Euler function r = (p − 1) · (q − 1). Randomly select an integer e to meet 1< er, and gcd(e, r) = 1. characteristics of the OPE algorithm make it suitable for processing order BY,MIN,MAX,SORT and other operations in database query statements without destroying the confidentiality of user data. V. PROXYLAYER The step for the remote database server to execute the query is to receive the SQL query statement, then parse it, convert the substantive text format to the internal binary structure combination, and then submit it to the internal optimizer to handle the query structure. However, in the case of a dense database, a database cannot process a user's regular plaintext SQL query request. Therefore, for this system, before the proxy server submits the real SQL query statement, the first thing we have to do is to process the user's regular plaintext SQL query request. Therefore, for this system, before the proxy server submits the real SQL query statement, the first thing we have to do is to parse the user's query, and then the system's encryption and decryption module overrides the original plaintext SQL query statement. In this paper, a simplified onion encryption strategy is designed. The cryptographic ciphertext of Rsa,paillier, and OPE is connected with special symbols in order to make the encrypted cipher satisfy multiplication, addition and comparison operation at the same time. One drawback of this approach, however, is that when ciphertext does one operation, only the corresponding homomorphic encryption text is updated, while the other homomorphic encryption text is not updated, so that no other operation can be performed. The solution is to send the previously updated cipher to the application layer between two different operations, decrypt and re-encrypt the data. Finally, the updated ciphertext data is sent to the proxy layer, and the proxy layer performs the next operation. This creates a new problem, which is that transferring ciphertext back and forth and updating the ciphertext creates an additional burden. In this paper, an optimization scheme is proposed to calculate the same operation once as far as possible. As many ciphertext as possible are then transferred to the application layer for ciphertext updates, which greatly reduces the pressure on the network transmission and application layer to update the ciphertext. The specific encryption process is shown in Figure 2. The first step is to encrypt the data with a combinatorial encryption algorithm. The second step analyzes the composition of the SQL statement and then rewrites the same operation into the same semantics. The third step calculates the different operations separately and then merges the results. Finally, the results are returned to the application layer, and the application layer is responsible for presenting the results to the user. B. Calculate the inverse of e, d = □□□ mod r. The public key is (n, e), The private key is (n, d). Encryption: Enter clear text m, ciphertext □ = □□ □□□ □. Decryption: Enter ciphertext c, clear text □ = □□ □□□ □. Addition homomorphic Encryption Scheme Addition homomorphic Encryption Scheme we select Paillier encryption scheme, which is based on the probabilistic public key cryptosystem of combined number factorial residual class. The specific encryption scheme is as follows: Key generation: Select two different large prime numbers. Set n=p.q. Calculate its Euler function □ = (□ − 1) · (□ − 1). Calculate □ = □□□(□ − 1, □ − 1). Randomly select an integer □ □ □□□ , and calculate □ = □□□□ □ (□(□ □□□ □)) □□□ □, Where the function L (u) = (u−1)/n is the division on the rational domain. The public key is(n, g), The private key is(□, □). Encryption: Enter clear text □ □ □□,Randomly select □ □ □□□□ □□,ciphertext□=□ ·□ □□□□ . Decryption: Enter ciphertext C, clear text □= □□□□ □□□ □□□ · □ □□□ □. C. Order-preserving Symmetric Encryption Order-preserving Symmetric Encryption(OPE) is a deterministic encryption scheme that enables ciphertext to maintain the order of its plaintext numeric value. That is if there is a key K, clear text x Authorized licensed use limited to: KING ABDUL AZIZ UNIVERSITY. Downloaded on August 26,2020 at 15:27:20 UTC from IEEE Xplore. Restrictions apply. operations, the improved plaintext database performance far exceeds the original ciphertext database. TABLE 2. Type PlainText database Original ciphertext Database Improved ciphertext database COMPARISON OF EXPERIMENTS Add operation Mixed operation 0.12s 0.3s 0.81s 3.2s 0.82s 1.4s Fig.2. The encryption process VI. DBMS LAYER The database management system that stores ciphertext is an important cornerstone of the project and the function of this layer is to provide database storage of ciphertext data. It has no essential difference from the database that the proxy layer has isolated stores plaintext and significant differences. The proxy layer is completely transparent to the user who executes the SQL query and to the DBMS. This paper uses a relational database MySQL. A relational database instead of putting all the data in a large repository, it saves the data in different tables, which speeds up the run and increases storage flexibility. MySQL uses Structured Query Language SQL, which is the most commonly used standardized language for accessing databases. MySQL is the best choice for this project because of its small size, fast speed, and open source characteristics. VII. IMPLEMENT AND EVALUATION This paper selects four SQL statements to test the performance differences between the original ciphertext database and the improved ciphertext database, with the following SQL statements: select sum(row1) from table1; select sum(row3) from (select row1*row2 as row3 from table1); The experimental results are as shown in the table2. From the experimental results, it can be seen that the performance of the plaintext database is far greater than that of the ciphertext database. When there is only one operation, the efficiency of the original ciphertext database is the same as that of the improved ciphertext database. When there are multiple The improved encryption scheme is to update the ciphertext after multiple identical operations, and then perform the next operation. When there are only one type of operation, neither the original encryption scheme nor the improved encryption scheme requires an application layer update ciphertext, so there is no difference between the two. When there are several types of operations, the original encryption scheme switches back and forth between multiple operations, so the ciphertext needs to be updated frequently. The improved encryption scheme greatly reduces the number of times the cipher is updated because it completes as many identical operations as possible each time. VIII. CONCLUSION AND FUTURE WORK Based on the design idea of database ciphertext processing, this paper designs an onion database encryption model under the premise that fully homomorphic encryption cannot meet the practicality, and completes the principal system of database encryption system. By setting up a simple test environment, the principle system of MySQL and onion database encryption system is tested respectively, the feasibility and practicability of the design idea of onion database encryption system are verified, and the design intention of SQL operation of dense database data is achieved. The following points need to be further worked out in the future: 1. The practical design of database encryption system and the design of Man-machine interface are insufficient, and further optimization is needed. 2. The replacement of the cryptographic algorithm of database encryption system will also be a problem to be considered. For different database applications, it is necessary to configure cryptographic algorithms with different strength� 3. How database encryption systems are deployed in a distributed environment further research is needed, and distributed databases, such as REDIS,MONGODB, may be used instead of Mysql in the future. REFERENCES

[1] R. Zhou�Research on encryption access control scheme of data cloud storage[D]�University of Electronic Science and Technology of China �2013. [2] D. Li�Research on cloud database encryption technology supporting ciphertext query[D] � Nanjing University of Aeronautics and Astronautics�2018. [3] D. G. Feng , Min. Z, Yan. Z, et al. Study on Cloud Computing Security[J]. Journal of Software, 2011, 162(1):388-396. [4] K. Zeng�Research and implementation of database encryption system based on onion model[D]�Northeastern University�2016. [5] L. Zhao � Research on the practical study of full homomorphic encryption technology[D] � University of Electronic Science and Technology of China�2017. [6] K. Wu�Design and implementation of database encryption system[D] �University of Electronic Science and Technology of China�2014. [7] Q.Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du and M. Guizani, “MeDShare: Trust-less Medical Data Sharing Among Cloud Service Providers Via Blockchain”, IEEE Access, Volume: 5, Pages: 14757 - 14767, July 2017 [8] Y. Sang, B. Ji, G. Gupta, X. Du, and L. Ye, “Provably Efficient Algorithms for Joint Placement and Allocation of Virtual Network Functions,” in Proc. of IEEE INFOCOM 2017, Atlanta, Ga, May 2017. [9] P. Dong, X. Du, H. Zhang, and T. Xu, “A Detection Method for a Novel DDoS Attack against SDN Controllers by Vast New Low-Traffic Flows,” in Proc. of the IEEE ICC 2016, Kuala Lumpur, Malaysia, May 2016. [10] Y. Xiao, X. Du, J. Zhang, and S. Guizani, “Internet Protocol Television (IPTV): the Killer Application for the Next Generation Internet,” IEEE Communications Magazine, Vol. 45, No. 11, pp. 126–134, Nov. 2007. [11] X. Du, M. Guizani, Y. Xiao and H. H. Chen, Transactions papers, “A Routing-Driven Elliptic Curve Cryptography based Key Management Scheme for Heterogeneous Sensor Networks,” IEEE Transactions on Wireless Communications, Vol. 8, No. 3, pp. 1223 - 1229, March 2009.