

Accelerate the Paillier Cryptosystem in CryptDB by Chinese Remainder Theorem

Yao Liu*, Shuai Xue*

*Shanghai Key Lab of Scalable Computing and System, Shanghai Jiao Tong University, China

liuyao@sjtu.edu.cn , mattxue_sjtu@sjtu.edu.cn

Abstract—CryptDB is an encrypted database management system which was proposed by CSAIL Lab of MIT. It can ensure the privacy of private data by executing SQL queries on the encrypted data. However, some encryption algorithms used in CryptDB might be time-consuming. Homomorphic encryption (HOM) is a secure probabilistic encryption scheme, and in CryptDB, the author implemented HOM by Paillier Cryptosystem, which is a probabilistic asymmetric algorithm for public key cryptography. Due to the complex computation, Paillier has become a bottleneck of the system performance. In this paper, we use Chinese Remainder Theorem(CRT) to accelerate the process of encryption. And results show that our method improve the performance of cryptDB under certain condition.

Keywords—Database, Security, Privacy, CryptDB, Encryption, CRT, Paillier Cryptosystem

I. INTRODUCTION

Most people have been familiar with Infrastructure-as-a-Service(IaaS), Platform-as-a-Service(PaaS) and Software-as-a-Service(SaaS), and some of them have exploited these services that the third party service providers provided. The arrival of the big data era and the prosperity of machine learning make the data be more and more significant. In order to create, store, access and manage the increasing databases, organizations or companies have to purchase expensive hardware and software to fulfill the business needs. Hence, the notion of Database-as-a-Service(DaaS) [1] [2] is proposed and gaining much attention.

Today, efficient data storage and data processing are crucial for most Internet companies and organizations. Database service providers can provide seamless mechanisms for companies and organizations to create, store, and access their databases. There is no need for the companies and organizations to manage their own databases, so that they can focus on their core business instead of the tedious work related to databases.

The emergence of DaaS brings huge advantages for the companies and organizations, meanwhile, it also brings some challenges. One of the most serious challenges is data privacy. In DaaS model, user data needs to be placed in the servers of database service providers. However, data is a very valuable asset for some companies and organizations. The service providers should provide sufficient measures to ensure the security of user data. Even though users can store encrypted data in service providers' server, when they want to acquire some results of SQL statements, the database service providers

still have to decrypt the user data to get the results, which could make the user data leaked.

In order to solve this problem, many researchers have come up with some methods. For example, ALzain [7], Li [8] uses multi shares to protect the privacy of data and Zhou [9] investigates ways used for data privacy in cloud environment. Pasupuleti [10] proposed an efficient and secure privacy-preserving approach for outsourced data of resource-constrained mobile devices in the cloud computing. Song [11] established a set of strict privacy requirements for full-text retrieval in cloud storage systems. Gai [12] proposed a novel data encryption approach Dynamic Data Encryption Strategy for data encryption. In general, the method to solve the problem is to executing SQL queries over Encrypted Data [3], and CryptDB is one of them.

CryptDB [4] is a DBMS that provides provable and practical privacy even in the presence of a compromised database server or curious database administrators. Its key approach is to execute queries over encrypted data using a collection of efficient SQL-aware encryption schemes. In order to enable the encrypted data to support various operations, such as equivalence checking, sort operation and so on, CryptDB uses onions of encryption. In each onion, they will use 2 or 3 onion encryption layers to encrypt data and each layer of each onion enables certain kinds of functionality. In the HOM onion, CryptDB use paillier cryptosystem, which is a very complexed and time-consuming algorithm. In order to improve system performance, CryptDB used cache of 20000 random numbers that would reduce the reliability and security of the system. Hence, in this paper, we use Chinese Remainder Theorem(CRT) to accelerate the computation of paillier cryptosystem. The Chinese remainder theorem is a theorem which gives a unique solution to simultaneous linear congruences with coprime moduli. We can use it to simplify the calculation procedure of the paillier cryptosystem.

II. BACKGROUND AND MOTIVATION

A. CryptDB

In the face of snooping Database Administrators(DBAs) and compromise from attackers, MIT Computer Science and Artificial Intelligence Laboratory(CSAIL) presented CryptDB, a system that provides practical and provable confidentiality under these threats. CryptDB adds a proxy server and other components to the typical structure of database-backed applications, which consists of a DBMS server and a separate

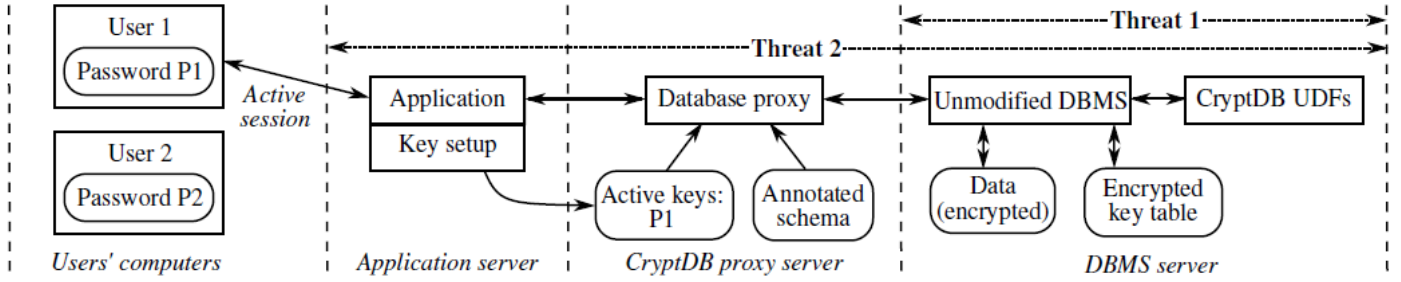


Figure 1. CryptDB's architecture.

application server, as shown in Figure 1. CryptDB's architecture consists of two parts: a database proxy and an unmodified DBMS. CryptDB uses user-defined functions(UDFs) to perform cryptographic operations in the DBMS. The database proxy protect the privacy of user data by intercepting all SQL queries, and rewrites queries to execute on encrypted data. The proxy encrypts and decrypts all data, and changes some query operations, while preserving the semantics of the query. To secure data the normal database schema is changed in order to hide any relations that can be read from the database, and then stored in the CryptDB proxy. In fact, table and column names are encrypted. Column encryption depends on the data held by that column, and the type of queries to be ran by the DBMS. Depending the type of data a column stores, there are six methods of encryption.

1) *Random(RND)*: RND produces a ciphertext from a column name using a randomly generated initial Vector(IV). Due to the randomness, RND provides a powerful encryption and is suitable when handling sensitive data. Unfortunately, it does allow running of queries that require computation e.g. MAX, SUM and ORDER BY.

2) *Deterministic(DET)*: DET provides a weaker security because of the leakage caused by producing the same ciphertext on same text. DET is a pseudo-random permutation.

3) *Order-perserving encryption(OPE)*: As the name suggests, it preserves the order of ciphertext to remain as they were in plaintext. For example, for any key k , if $x < y$, then $OPE_K(x) < OPE_K(y)$. OPE is comparatively weaker than DET because it reveal order relationship.

4) *Homomorphic(HOM)*: HOM is handy for any data that requires computation. With it, complex mathematical computations are possible as they could be done plaintext.

5) *JOIN and OPE-JOIN*: JOIN and OPE-JOIN are used to join columns to hide the correlation between cross-column. Because different DET keys are used. Joins are done for equality and order checks.

6) *SEARCH*: SEARCH is a method used to search encrypted words. This method is used in queries with SQL operations such as LIKE. SEARCH is as secure as RND because it does not allow the DBMS to see whether some of keyword repeats in many rows.

B. Paillier Cryptosystem

In order to implement HOM schema in CryptDB, they use Paillier cryptosystem [5] [6] to support summation. The Paillier cryptosystem, invented by and named after Pascal Paillier in 1999, is a probabilistic asymmetric algorithm for public key cryptography. It is also an additive homomorphic cryptosystem. With Paillier, the product of two encrypted values is equals to the encryption of the two values' sum, i.e., $HOM_K(x) * HOM_K(y) = HOM_K(x + y)$. There are three steps in Paillier Cryptosystem.

1) Key generation:

- Choose two large prime number p and q randomly and independently of each other, then compute $n = p * q$ and $\lambda = lcm(p-1, q-1)$ such that $gcd(p*q, (p-1)*(q-1)) = 1$.
- Select random integer g where $g \in \mathbb{Z}_{n^2}^*$, and ensure n divides the order of g by checking the existence of the following modular multiplication inverse: $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ where $L(x) = x - 1/n$.
- (n, g) acts as a public key.
- (λ, μ) acts as a private key.

2) Encryption:

- Let m be the message to be encrypted where $0 \leq m < n$.
- Choose r where $0 \leq r < n$
- Compute ciphertext as $c = g^m * r^n \bmod n^2$.

3) Decryption:

- Let c be the ciphertext to decrypt, where $c \in \mathbb{Z}_{n^2}^*$.
- Compute the plaintext message as: $m = L(c^\lambda \bmod n^2) * \mu \bmod n$

In order to meet the security requirements, the value of prime number p and q are both 1024 bits in CryptDB, so the public key n reaches a length of 2048 bits. As we all known, it's a time-consuming process to modulo and power on these large numbers.

C. Chinese Remainder Theorem

The Chinese remainder theorem [13] is a theorem of number theory, which states that, if one knows the remainders of the division of an integer n by several integers, then one can determine uniquely the remainder of the division of n by the

product of these integers, under the condition that the divisors are pairwise coprime.

Given pairwise coprime positive integers $n_1, n_2, \dots, n_k$ and arbitrary integers $a_1, a_2, \dots, a_k$, the system of simultaneous congruences

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ x &\equiv a_2 \pmod{n_2} \\ &\vdots \\ x &\equiv a_k \pmod{n_k} \end{aligned} \quad (1)$$

has a solution, and the solution is unique modulo $N = n_1 n_2 \dots n_k$.

The Chinese remainder theorem is widely used for computing with large integers, as it allows replacing a computation for which one knows a bound on the size of the result by several similar computations on small integers. The Chinese remainder theorem is true over every principal ideal domain. It has been generalized to any commutative ring, with a formulation involving ideals.

III. IMPLEMENTATION

From the above, we know that in Paillier Cryptosystem, we use the formula $c = g^m * r^n \pmod{n^2}$ to compute ciphertext. Since n is the product of p and q and both p and q are integers of 1024 bits, the bits of n is 2048. Thus, the most time-consuming part is to calculate r^n in Paillier Cryptosystem. As we all known, n^2 can be split into $p^2 * q^2$, we can use the CRT to convert the formula used for encryption into two smaller computational parts. For this method, there are two steps.

1) Compute intermediate quantity:

$$qInv = (1/q^2) \pmod{p^2} \quad (2)$$

2) For a given plaintext m , calculate ciphertext c :

$$\begin{aligned} c_1 &= r^n \pmod{p^2} \\ c_2 &= r^n \pmod{q^2} \\ h &= qInv * (c_1 - c_2) \pmod{p^2} \\ cTemp &= c_2 + h * q^2 \\ c &= g^m * cTemp \pmod{n^2} \end{aligned} \quad (3)$$

Using CRT, we convert the process of calculating ciphertext c into two smaller parts(calculating ciphertext c_1 and calculating ciphertext c_2). Then, we combine the formula $h = qInv * (c_1 - c_2) \pmod{p^2}$ and the intermediate quantity $qInv$, and finally, we can get the result c from c_1 and c_2 . As the computational scale reduced, the performance of the entire encryption process is improved.

Using CRT to optimize performance, we need to know the values of p and q at first. While in Paillier Cryptosystem, we just know the public key (n, g) in the phase of encryption and we need to know private key (p, q) in the phase of decryption. In general, the encryption side and decryption side are belong to different objects. The encryption side only gets public key that used for encryption and the decryption side knows private

key used for decryption. Thus, CRT can only be used in the phase of decryption.

However, as we can see from Figure 1, in CryptDB, the proxy intercepts all SQL queries, and encrypts queries. When receiving encrypted results, the proxy will decrypt the results and then send results to clients. Since the proxy takes on the job of encryption and decryption, the proxy knows both public key and private key. That is, the proxy knows the values of p and q . Then, we can use CRT to improve the performance.

IV. EVALUATION

In our evaluation, the server we used is equipped with Intel(R) Xeon E5-2680 CPUs, 70G RAM. We use oltpbench [14] as our benchmark framework and choose the YCSB benchmark to evaluate the system's performance. The server runs the benchmark, MySQL-Proxy and MySQL server. All the workloads fit in the server's RAM.

Since Paillier Cryptosystem is used in HOM and HOM is a operation for integers, we mainly evaluate the effect on integers.

First, we test the encryption/decryption speed of 1 integer. The results was shown in Table 1, and we can see that when encrypting or decrypting an integer, the speed of CRT version is obviously faster than the original version.

TABLE I
THE ENCRYPTION/DECRYPTION SPEED OF ORIGINAL/CRT

	Original	CRT
min	2.57ms	1.41ms
max	7.47ms	4.29ms
avg	6ms	2.8ms

YCSB benchmark has one int field as primary key and other fields in its table, and we can change the number and data type of the fields. In our evaluation, we tested the system performance on different numbers of int fields. For example, YCSB-int1field means the table has one int fields and YCSB-int2field represents the table has one primary key field and one int field.

The results of our evaluation in Figure 2 shown that CRT largely improved the system performance under our circumstance. We tested the performance of CryptDB under YCSB-int1field, YCSB-int2field and YCSB-int4field, and compared with original CryptDB, the CRT version brings 28.14%, 34.38% and 61.55% improvement respectively.

V. CONCLUSION

In this paper, we use CRT to accelerate the Paillier Cryptosystem in CryptDB. Then, we use YCSB benchmark to evaluate the performance of CryptDB before/after CRT optimization. The results shown that CRT largely improve the system performance under certain condition. Today, there have been many new methods to accelerate data encryption or decryption, and maybe we can use them to improve CryptDB's performance later.

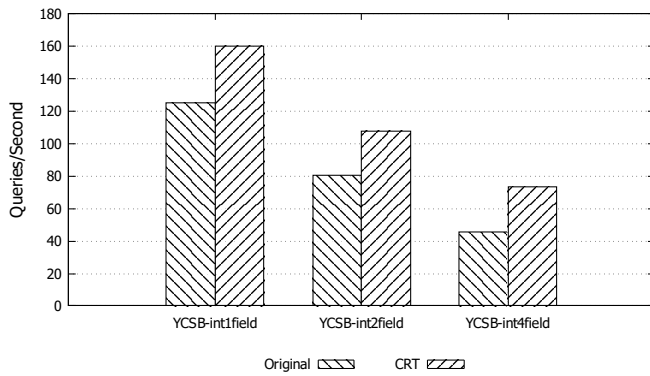


Figure 2. CryptDB's performance using YCSB benchmark.

REFERENCES

- [1] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer and W. Karl, "Scientific Cloud Computing: Early Definition and Experience," 2008 10th IEEE International Conference on High Performance Computing and Communications, Dalian, 2008, pp. 825-830. doi: 10.1109/HPCC.2008.38
- [2] H. Hacigumus, B. Iyer and S. Mehrotra, "Providing database as a service," Proceedings 18th International Conference on Data Engineering, San Jose, CA, 2002, pp. 29-38. doi: 10.1109/ICDE.2002.994695
- [3] Hakan Hacigm, Bala Iyer, Chen Li, and Sharad Mehrotra. 2002. Executing SQL over encrypted data in the database-service-provider model. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data (SIGMOD '02). ACM, New York, NY, USA, 216-227. DOI: <https://doi.org/10.1145/564691.564717>
- [4] Popa R A, Redfield C, Zeldovich N, et al. CryptDB: protecting confidentiality with encrypted query processing. Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. ACM, 2011: 85-100.
- [5] The wikipedia website. [Online].https://en.wikipedia.org/wiki/Paillier_cryptosystem
- [6] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the 18th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Prague, Czech Republic, May 1999.
- [7] M. A. ALzain and E. Pardede, "Using Multi Shares for Ensuring Privacy in Database-as-a-Service," 2011 44th Hawaii International Conference on System Sciences, Kauai, HI, 2011, pp. 1-9. doi: 10.1109/HICSS.2011.478
- [8] Ping Li, Jin Li, Zhengang Huang, Tong Li, Chong-Zhi Gao, Siu-Ming Yiu, Kai Chen, Multi-key privacy-preserving deep learning in cloud computing, In Future Generation Computer Systems, Volume 74, 2017, Pages 76-85, ISSN 0167-739X, <https://doi.org/10.1016/j.future.2017.02.006>.
- [9] M. Zhou, R. Zhang, W. Xie, W. Qian and A. Zhou, "Security and Privacy in Cloud Computing: A Survey," 2010 Sixth International Conference on Semantics, Knowledge and Grids, Beijing, 2010, pp. 105-112. doi: 10.1109/SKG.2010.19
- [10] Syam Kumar Pasupuleti, Subramanian Ramalingam, Rajkumar Buyya, An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing, In Journal of Network and Computer Applications, Volume 64, 2016, Pages 12-22, ISSN 1084-8045, <https://doi.org/10.1016/j.jnca.2015.11.023>.
- [11] Wei Song, Bing Wang, Qian Wang, Zhiyong Peng, Wenjing Lou, Yihui Cui, A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications, In Journal of Parallel and Distributed Computing, Volume 99, 2017, Pages 14-27, ISSN 0743-7315, <https://doi.org/10.1016/j.jpdc.2016.05.017>.
- [12] K. Gai, M. Qiu and H. Zhao, "Privacy-Preserving Data Encryption Strategy for Big Data in Mobile Cloud Computing," in IEEE Transactions on Big Data, vol. PP, no. 99, pp. 1-1. doi: 10.1109/TBDDATA.2017.2705807
- [13] The wikipedia website. [Online].https://en.wikipedia.org/wiki/Chinese_remainder_theorem
- [14] Difallah D E, Pavlo A, Curino C, et al. Oltp-bench: An extensible testbed for benchmarking relational databases. Proceedings of the VLDB Endowment, 2013, 7(4): 277-288.



Liu Yao received the B.S. degree in software engineering from Southeast University in 2015. She is now studying in the school of computer science and technology in Shanghai Jiao Tong University, Shanghai, China. Her research interests are mainly focused on encrypted databases, cloud computing, hardware acceleration, high availability.



Xue Shuai received the B.S. degree in software engineering from Huazhong University of Science and Technology in 2015. He currently pursues the M.S. degree in the school of software engineering in Shanghai Jiao Tong University, Shanghai, China. His research interests are mainly focused on hardware-assisted virtualization, cloud computing, hardware acceleration, Internet of things, and distributed in-memory databases.