Research on order-preserving encryption scheme based on CryptDB

Xiaoning Jiang¹, Xiangfeng Kong²* and Zhenchi Xu³

¹School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, Zhejiang, 310018, China
²School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, Zhejiang, 310018, China
³School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, Zhejiang, 310018, China
*Corresponding author’s e-mail: kongxffs@foxmail.com

Abstract. With the rapid development of cloud computing technology, cloud database, as an important part of cloud computing services, has gradually become necessary for daily work of enterprises or individuals. However, entrusting data to third-party managing can lead to security issues such as data leakage, and users cannot guarantee data security. To this end, this paper is based on the CryptDB, an open source database encryption proxy system designed by MIT, in this thesis we propose an improvement scheme for the shortcomings of the original CryptDB system. Specific contents including: By studying the CryptDB system, we find that the system is lack of scalability for different databases and does not involve the management of system keys. For the inefficiency of the mutable Order-Preserving Encryption (mOPE) in CryptDB system, we propose an improved additive Order-Revealing Encryption (aORE) scheme by combining the Practical Order-Revealing Encryption (P-ORE) and mOPE. The scheme is based on pseudo-random function and double encryption. Compared with mOPE, it can improve the execution efficiency of the Order-Preserving scheme at the expense of security.

1. Introduction
In recent years, with the rapid development of cloud computing technology [1], various applications based on cloud computing have received extensive attention in the market. Cloud database [2] is a third-party database service provider that provides data storage and access functions for enterprises or individuals through network services. With the rapid development of cloud database storage technology, security has always been an "obstacle" to its further promotion, and with the continuous popularity of cloud database, this problem is increasingly prominent. In today's era of big data, data security has received more and more attention, and for enterprises, data security has always been the most important issue. Because some parts of the data may be sensitive, such as trade secrets, credit card numbers or other personal information, companies have reservations about data storage and processing in the form of outsourcing. If the infrastructure is attacked by an attacker, sensitive data stored on the infrastructure provided by a third party increases the risk of malicious disclosure. This paper mainly aims at the security requirements of cloud storage database, combined with the CryptDB system proposed by MIT laboratory, and gives the improvement scheme, aiming at improving the security and practicability of database encryption agent system. The main research content is as follows: As a research system,
CryptDB encryption agent system still has some shortcomings. The ciphertext retrieval function of the system depends on the custom function of the database, which leads to the poor expansibility of the system. In addition, the system does not involve the management of encryption keys. Through the in-depth study of the CryptDB system, the analysis shows that the algorithm efficiency of mOPE is low. Based on Chenette's practical order-encryption (p-ore), this paper proposes an improved additively order-encryption (aORE) scheme. Compared with mOPE, this improved scheme can improve the execution efficiency of order preserving scheme at the sacrifice of some security. The improved scheme of basic encryption algorithm proposed in this paper is applied to the CryptDB system, and test cases are designed to test the improved system, and the influence of the improved scheme on the performance of the system is analyzed. An improved encryption scheme aORE is proposed, which can improve the execution efficiency of the scheme compared with mOPE without sacrificing security.

2. Research on Preserving Encryption Scheme

Order-preserving encryption (OPE) is a special encryption scheme that can maintain the plaintext sequence after encrypting the plaintext data. Order-preserving encryption, while ensuring the confidentiality of data, can directly compare ciphertexts, so it has become an important tool for building a searchable encryption system. Rakesh Agrawal et al. [3] first proposed a formal definition of OPE, but lacked a security analysis. Then Boldyreva et al. [4] gave the first formal security definition of the OPE scheme, that is, indistinguishability (IND-OCPA) under an ordered selective plaintext attack. The definition of IND-OCPA indicates that encryption of a message sequence should not reveal information other than the plain text ordering. But at the time Boldyreva et al thought that IND-OCPA was the ideal security, and the OPE scheme could not achieve this security definition. Subsequently, Popa RA et al. [5] proposed a variable order-preserving coding scheme (mOPE) in 2013, which is the only OPE scheme that currently meets the security requirements of IND-OCPA. In this scheme, the ciphertext only retains the order information of the plaintext, and can also achieve IND-OCPA security when the ciphertext is variable. Boneh et al. [6] proposed an order-revealing encryption (ORE) scheme. ORE, as an extension of the OPE scheme, does not make any specific form of constraints on the ciphertext, but uses publicly available comparison Function to compare two ciphertexts and output the order of the plaintext. The ORE scheme implements a security concept comparable to IND-OCPA, but because the construction process relies on multi-linear mapping, this scheme is difficult to implement for most applications and is still a theoretical result. Chenette et al. [7] proposed a P-ORE scheme based on Boneh. This method uses a pseudo-random function to build a new ORE scheme based on the existing OPE scheme, reducing the complexity of the construction. And its security is basically consistent with the underlying OPE scheme.

CryptDB database encryption agent system uses mOPE as the system's OPE scheme. MOPE meets the definition of provable ideal security, but its efficiency is low. Based on in-depth research on order-preserving encryption schemes, this paper uses Chenette's construction ideas to design an improved order-preserving scheme based on P-ORE and mOPE. Compared with mOPE, this improved scheme sacrifices some security, but improves It improves the execution efficiency of order-preserving encryption.

3. Variable order preserving encryption scheme mOPE

mOPE is the first order-preserving encryption scheme that implements provable ideal security. It constructs a balanced search tree that contains all plaintext values encrypted by the application. It encodes the plaintext values according to the order of their size before inserting, and The encoding is stored in the database as part of the ciphertext. In the mOPE scheme, the order-preserving encoding of a value refers to the path from the root node of the search tree to the node in the tree where the value is incremented from left to right in the tree. MIT researchers use binary encoding to represent the path. For plaintext, if $x$ is less than $y$, the path of $x$ will be on the left side of the path of $y$. 
MIT researchers have proposed a new order-preserving encryption model that includes an OPE client and an OPE server that interact with each other. The client is the owner of the encrypted data, so the client is trusted and the server is not trusted. The model structure is shown in Figure 1.

![OPE structure and coding example diagram](image)

**Figure 1.** mOPE structure and coding example diagram

**4. Sequentially visible encryption scheme P-ORE**

P-ORE is an order-preserving encryption scheme constructed by Chenette et al. By using a pseudo-random function, this scheme overcomes the problem that the previous ORE scheme is difficult to effectively implement and improves security. Researchers have proposed a new security definition for the ORE scheme, which states that even if an attacker can learn the encryption process of the ORE scheme, he can only obtain the sequence information of the ciphertext.

**5. Improved order-preserving encryption scheme aORE**

Based on the in-depth study of the P-ORE scheme, this paper first realized the P-ORE scheme by constructing a pseudo-random function, and then combined the ORE scheme with the m-OPE scheme to propose an improved order-preserving encryption scheme Order-Revealing Encryption (additively Order-Revealing Encryption, aORE).

Based on Chenette’s ideas, this paper constructs an improved order-preserving encryption scheme aORE by combining P-ORE and mOPE. The scheme consists of \((KeySetup, Encrypt, Cmp, Decrypt)\) algorithm tuple. The main implementation steps are as follows:

1. Key generation algorithm \(KeySetup(1^k)\rightarrow sk\): The key generation algorithm consists of two parts, \(SM4.KeyGen(1^k)\rightarrow sk_1\) and \(Setup(1^k)\rightarrow sk_2\), which is the encryption key \(sk=(sk_1,sk_2)\).

2. Encryption algorithm \(Encrypt(sk,pt)\rightarrow ct\): The encryption algorithm first uses the national secret algorithm SM4 to encrypt the plaintext \(pt\) once, then based on this, the plaintext is encrypted twice, which is \(SM4.Enc(sk_1,pt)\), and finally the ciphertext \(ct\) is obtained.

In the encryption algorithm \(SM4.Enc(sk_1,pt)\) is used to encrypt the plaintext once, and the encryption is implemented by using the SM4_CMC mode. For the \(Enc\) algorithm, the plaintext \(pt\) is first converted into the binary bit sequence \(\left( pt_1, \cdots, pt_n \right) \), and each bit value is bit-calculated based on the pseudo-random number sequence. And large integer operations, finally the encrypted value is recombined into ciphertext sequence \(\left( ct_1, \cdots, ct_n \right)\) and ciphertext \(ct\) is returned.

3. Ciphertext comparison algorithm \(Cmp(ct1,ct2)\rightarrow -1/0/1\): The ciphertext comparison algorithm compares the two input ciphertext, if \(ct1 > ct2\), it returns 1; if \(ct1 = ct2\), it returns 0; if \(ct1 < ct2\), it returns -1. For ciphertext \(ct1=(ct1_1,ct1_2,\cdots,ct1_n)\) and ciphertext \(ct2=(ct2_1,ct2_2,\cdots,ct2_n)\), it
compares in order \((ct_{l_1}, ct_{l_2}), \ldots, (ct_{l_n}, ct_{l_n})\), if \(i\) does not exist such that \(ct_{l_i} \neq ct_{l_i}\) then \(ct_{l} = ct_{l_2}\) returns 0; if \(i\) exists and \(ct_{l_i} = ct_{l_2} + 1 \pmod{M}\) then \(ct_{l} > ct_{l_2}\) returns 1; otherwise returns -1.

(4) Decryption Algorithm \(\text{Decrypt}(sk, ct) \rightarrow pt\): Use the encryption algorithm and ciphertext comparison algorithm to form the decryption algorithm. First, the ordered plaintext sequence is sequentially encrypted, and then the encrypted ciphertext is sent to the backend (database). The backend uses a ciphertext comparison algorithm to perform a binary search on the ciphertext to be decrypted, and returns the corresponding index value when found.

6. aORE program analysis

This section will test the performance of the mOPE scheme, P-ORE scheme, and the aORE improvement scheme proposed in this article. Among them, P-ORE and the aORE proposed by this article are implemented by C language. For mOPE, it is implemented using the OPE module in the CryptDB system.

6.1. Test environment

The test environment used in the experiments in this chapter is as follows: the operating system is Ubuntu-12.04.1-server-amd64; the hardware environment is: the CPU is configured as Intel (R) Core (TM) CPU i5-4590 @ 3.30GHz 2 cores, and the memory configuration is 4GB 1333MHz, hard disk configuration is 50GB SATA mechanical hard disk; software environment is gcc4.6.

6.2. Experiment design

In this experiment, encryption algorithms in mOPE, P-ORE, and aORE are used to perform encryption and decryption tests on four sets of data with different plaintext lengths of the same content. The test program uses the time function library that comes with C language, and calculates the encrypted time difference through the clock () and difftime () functions. The time is accurate to microseconds. Each algorithm performs 1000 operations on the plaintext space of each length, taking the average as the final time consuming.

6.3. Experimental results

The test results are shown in Table 1, Table 2, Figure 2 and Figure 3. From the test data, it can be seen that the improvement scheme proposed in this paper is better than mOPE in terms of encryption and decryption time, because the improvement scheme does not use multilinearity. Complex operations such as mapping, while abandoning a large number of interactive operations in the mOPE scheme, have improved the algorithm's performance. For the P-ORE scheme, because the improved scheme proposed in this article adds additional key initialization and secondary encryption operations, the algorithm encryption time is increased.

| Table 1. Encryption of three order-preserving schemes takes time |
|---------------------------------------------------------------|
|                  | 32bit   | 64bit   | 96bit   | 128bit  |
| mOPE             | 235.1μs | 463.7μs | 692.7μs | 914.2μs |
| P-ORE            | 6.49μs  | 13.47μs | 20.45μs | 27.90μs |
| aORE             | 39.82μs | 53.11μs | 65.80μs | 79.47μs |
Figure 2. Encryption time consuming for three order-preserving scheme

Table 2. Decryption of three order-preserving schemes takes time

|       | 32bit | 64bit | 96bit | 128bit |
|-------|-------|-------|-------|--------|
| mOPE  | 108.2μs | 206.5μs | 329.1μs | 466.3μs |
| aORE  | 54.13μs  | 67.3μs  | 80.21μs | 97.6μs |

Figure 3. Decryption time of mOPE and aORE scheme

7. Conclusion
This paper first introduces the research background of the order-preserving encryption scheme, and then introduces the order-preserving encryption scheme mOPE used by the CryptDB system and a sequential visible encryption scheme P-ORE. Aiming at the problem of low execution efficiency of the mOPE scheme, this chapter proposes an improved scheme aORE based on P-ORE and mOPE, and gives the algorithm's pseudocode. Secondly, the security of aORE scheme is analyzed according to the unidirectional security definition proposed by Chenette. From the analysis, it can be seen that the aORE scheme inherits the definition of unidirectional security and improves the security by introducing secondary encryption, but fails to achieve the ideal security of the mOPE scheme. Finally, the performance of the improved scheme proposed in this paper is compared with the performance of mOPE and P-ORE through encryption and decryption performance test experiments. The experimental results show that for mOPE, the improved aORE scheme proposed in this paper can improve the execution efficiency of order-preserving schemes at the expense of certain security.
References
[1] Mell P M, Grance T. The NIST Definition of Cloud Computing[M]. National Institute of Standards & Technology, 2011.
[2] Elmore A J, Das S, Agrawal D, et al. Zephyr: live migration in shared nothing databases for elastic cloud platforms[C]. ACM SIGMOD International Conference on Management of Data. ACM, 2011:301-312.
[3] Agrawal R, Kiernan J, Srikant R, et al. Order preserving encryption for numeric data[C]. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM, 2004: 563-574.
[4] Agrawal R, Kiernan J, Srikant R, et al. Order preserving encryption for numeric data[C]. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM, 2004: 563-574.
[5] Popa R A, Li F H, Zeldovich N. An ideal-security protocol for order-preserving encoding[C]. Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, 2013: 463-477.
[6] Boneh D, Lewi K, Raykova M, et al. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation[C]. Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2015: 563-594.
[7] Chenette N, Lewi K, Weis S A, et al. Practical order-revealing encryption with limited leakage[C]. International Conference on Fast Software Encryption. Springer, Berlin, Heidelberg, 2016: 474-493.