Secure Database Using Order-Preserving Encryption Scheme Based on Arithmetic Coding and Noise Function

Sergey Krendelev(✉), Mikhail Yakovlev, and Maria Usoltseva

Novosibirsk State University, Pirogova str. 2, 630090 Novosibirsk, Russia
{s.f.krendelev,m.o.yakovlev,m.a.usoltseva}@gmail.com

Abstract. Order-preserving symmetric encryption (OPE) is a deterministic encryption scheme which encryption function preserves numerical order of the plaintexts. That allows comparison operations to be directly applied on encrypted data in case, for example, decryption takes too much time or cryptographic key is unknown. That's why it is successfully used in cloud databases as effective range queries can be performed based on. This paper presents order-preserving encryption scheme based on arithmetic coding. In the first part of it we review principles of arithmetic coding, which formed the basis of the algorithm, as well as changes that were made. Then we describe noise function approach, which makes algorithm cryptographically stronger and show modifications that can be made to obtain order-preserving hash function. Finally we analyze resulting vulnerability to chosen-plaintext attack.

Keywords: Cloud computing security · Order-preserving encryption · Symmetric-key cryptosystems · Order-preserving hash functions

1 Introduction

Nowadays, the amount of information stored in various databases steadily increases. In order to store and effectively manage large amounts of data it is needed to increase data storages capacity and allocate funds for its administration. Another way that was chosen by many companies is to give the database management to a third-party. Such service is managed by a cloud operator and is called Database as a Service, DBaaS.

Obviously, this approach has its own flaws. And the most important of them is security issue. Data can be stolen by the service provider itself or by someone else from its storage. Fortunately, this problem can be solved by encryption. Of course if we just encrypt the whole database with a conventional encryption algorithm, we'll have to encrypt and decrypt it each time we need something. So, all advantages will be lost. That's why special encryption schemes, such as homomorphic encryption and order-preserving encryption, are developed. The first one allows us to handle encrypted data, and the second – to sort them and select the desired.

All known order-preserving schemes have significant problems, such as low level of security (polynomial monotonic functions [1], spline approximation [2], linear functions with random noise [3]), low performance (summation of random numbers

[4], B-trees [5]) or too-large numbers proceeding (scheme by Boldyreva [6]). Proposed scheme doesn't have these disadvantages and, furthermore, unlike all the others can be used to encrypt real numbers. Also it can be used to obtain order-preserving hash function.

This algorithm combines two main ideas, which the majority of OPE schemes operate with: monotonic functions design and elements of coding theory (implicit monotonic functions design). It is claimed that scheme is based on arithmetic coding and noise function, but, in fact, this article considers only the case with binary alphabet. In theory, nothing prevents the use of an arbitrary one.

First, let's give a definition of order-preserving encryption. Assume there are two sets A and B with order relation $<$. Function $f : A \rightarrow B$ is strictly increasing if $\forall x, y \in A, x < y \Leftrightarrow f(x) < f(y)$. Order-preserving encryption is deterministic symmetric encryption based on strictly increasing function.

The described order-preserving encryption scheme was developed in Laboratory of Modern Computer Technologies of Novosibirsk State University Research Department as a part of "Protected Database" project[1] and is based on arithmetic coding and noise function. Let us consider them precisely.

2 Splitting Procedure of Arithmetic Coding

Suppose c is non-negative integer number requiring for its representation n bits, i.e.

$$c = \sum_{i=1}^{n} \alpha_i 2^i$$

where $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ is a bit string, α_1 is the MSB. Let us define the bijection f. Assume that the string $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ defines certain real number $s \in [0, 1)$ as follows:

$$s = \frac{c}{2^n}.$$

Let us find another representation for the number s. In order to do it, we use the idea of arithmetic coding. Notice that the number s satisfies the equation $2^n s = c$. The equation

$$G(x) = 2^n x - c = 0$$

has only one solution on the interval $[0, 1)$. If we solve this equation using a standard binary search, we get the initial number s after n steps. The main idea of arithmetic coding is that intervals can be split into parts randomly. In this case approximate solution of the equation can be found after the less number of steps. That allows us to

[1] This research is performed in Novosibirsk State University under support of Ministry of Education and Science of Russia (contract no. 02.G25.31.0054).

achieve compression of data while using arithmetic coding. First of all, let us consider the splitting procedure.

Suppose $\gamma = \frac{p}{p+q}, \mu = \frac{q}{p+q}$, where p, q are random natural numbers. Obviously, $\gamma + \mu = 1$. Let us split the interval $[0, 1)$ into two parts $\left[0, \frac{p}{p+q}\right), \left[\frac{p}{p+q}, 1\right)$. If $G\left(\frac{p}{p+q}\right) > 0$, the interval $\left[0, \frac{p}{p+q}\right)$ is selected, and the output is 0-bit ($\beta_1 = 0$). If $G\left(\frac{p}{p+q}\right) < 0$, the interval $\left[\frac{p}{p+q}, 1\right)$ is selected, and $\beta_1 = 1$. Let us denote $[a_1, b_1)$ the interval was selected.

This interval is again split into parts in the ratio $\gamma : \mu$. According to the sign of function $G(x)$ in the splitting point, one of the segments is selected. Proceeding by induction, the interval $[a_k, b_k)$ can be calculated for $\forall k$. Its length is $\gamma^r \mu^{n-r}$, where r is the number of zeros in string β. If $\forall r : \frac{1}{2^n} < \gamma^r \mu^{k-r}$, then $s \in [a_k, b_k)$ and $c = 2^n s$ are uniquely defined by $\beta = (\beta_1, \ldots, \beta_k)$. It is also obvious that this mapping preserves an order.

Generalizing used in the adaptive arithmetic coding, as well as in the proposed algorithm, is that it is possible to use different ratio on each step. This allows us to achieve stronger security of encryption.

3 Noise Function

It is known that the composition of two strictly increasing functions strictly increases. Therefore, to provide stronger security of cryptographic algorithm special random strictly increasing function is used in addition to the splitting procedure. In fact, we use inverse function of the one that was generated.

It was proved [6] that OPE schemes cannot satisfy the standard notions of security, such as indistinguishability against chosen-plaintext attack (IND-CPA) [7], since they leak the ordering information of the plaintexts. If an adversary knows plaintexts p_1, p_2 and corresponding ciphertexts c_1, c_2 and c, such that $c_1 < c < c_2$, it is obvious that the plaintext for c lies in the interval (p_1, p_2). In addition, the adversary can always find the decryption function in some approximation, for instance, using linear interpolation.

And moreover, in case of using, for example, encryption method developed by David A. Singer and Sun S. Chung [1], where strictly increasing polynomial functions $f(x) = a_0 + a_1 x + \ldots + a_n x^n$ are used for encryption, the adversary can calculate the exact encryption function if he has $(n + 1)$ arbitrary pairs (plaintext, ciphertext). It is enough to solve the system of equations:

$$\begin{cases} a_0 + a_1 x_0 + \ldots + a_n x_0^n = y_0 \\ a_0 + a_1 x_1 + \ldots + a_n x_1^n = y_1 \\ \vdots \\ a_0 + a_1 x_n + \ldots + a_n x_n^n = y_n \end{cases}$$

Thus, the adversary can get $(a_0, \ldots a_n)$ and correspondingly encryption function $f(x)$.

In order to complicate his task it is necessary to maximize the amount of pairs required for this attack and complexity of the system of equations $f(x_i) = y_i$. Therefore, it was decided to generate noise function from class of function

$$f(x) = \int_c^x \left(a_0 + a_1 t + a_2 t^2\right)(a_3 + a_4 \sin(a_5 + a_6 t) + a_7 \cos(a_8 + a_9 t))dt,$$

where c is an arbitrary constant and coefficients a_i are selected so that

$$\left(a_0 + a_1 t + a_2 t^2\right)(a_3 + a_4 \sin(a_5 + a_6 t) + a_7 \cos(a_8 + a_9 t)) > 0$$

for $\forall t \in (c; x_{max})$. In this case $f(x)$ is strictly increasing function (see Fig. 1). This integral can be calculated explicitly, which increases the speed of function value calculation. Nevertheless, the system of equations

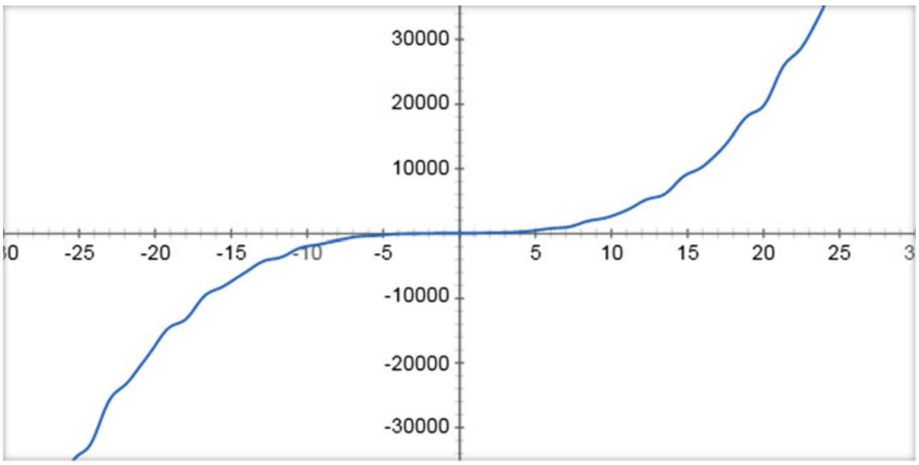

Fig. 1. Example of the correct noise function from the class. Due to such combination of sine and cosine, its behavior is hard to predict without $(a_0, \ldots a_9)$ coefficients knowledge.

$$\begin{cases} \displaystyle\int_c^{x_0} \begin{matrix}(a_0 + a_1 t + a_2 t^2)(a_3 + a_4 \sin(a_5 + a_6 t) + \\ a_7 \cos(a_8 + a_9 t))\, dt\end{matrix} = y_0 \\ \displaystyle\int_c^{x_1} \begin{matrix}(a_0 + a_1 t + a_2 t^2)(a_3 + a_4 \sin(a_5 + a_6 t) + \\ a_7 \cos(a_8 + a_9 t))\, dt\end{matrix} = y_1 \\ \vdots \\ \displaystyle\int_c^{x_k} \begin{matrix}(a_0 + a_1 t + a_2 t^2)(a_3 + a_4 \sin(a_5 + a_6 t) + \\ a_7 \cos(a_8 + a_9 t))\, dt\end{matrix} = y_k \end{cases}$$

is difficult to solve, which indicates that proposed algorithm is cryptographically strong against this type of attack.

4 Cryptographic Scheme

4.1 Key Generation

As a private key of encryption algorithm we consider noise function $f(x) = \int_c^x (a_0 + a_1 t + a_2 t^2)(a_3 + a_4 \sin(a_5 + a_6 t) + a_7 \cos(a_8 + a_9 t))dt$ and a set of ratios (p_i, q_i).

In order for an encrypted n-bit number to be uniquely decrypted, the length of intervals computed during decryption has to be less than $\frac{1}{2^n}$. The largest length of the interval that can be obtained during decryption is $\prod_i \frac{\max(p_i,q_i)}{p_i+q_i} f'_{\max}(x)$. So the algorithm of calculation the set of ratios is:

1. `Generate random ratios` p_i, q_i.
2. `Check the condition`

$$\prod_i \frac{\max(p_i, q_i)}{p_i + q_i} f'_{\max}(x) < \frac{1}{2^n}$$

   `If this conditions if satisfied, go to the step 3, else go back to the step 1.`
3. `Output the set of ratios` $(p_1, q_1), (p_2, q_2), \ldots, (p_k, q_k)$.
   `The key is the set` $K = [(a_0, \ldots, a_9), (p_1, q_1), (p_2, q_2), \ldots, (p_k, q_k)]$.

4.2 Encryption

Assume we need to encrypt n-bit integer s with the key $K = [f(x), (p_1, q_1), (p_2, q_2), \ldots, (p_k, q_k)]$, where $f(x)$ is a noise function, $f(a_0) = 0$, $f(b_0) = 2^n$, and (p_i, q_i) is a set of ratios. Consider the i-th iteration of algorithm.

The current interval $[a_{i-1}, b_{i-1})$ is split in the ratio $p_i : q_i$. Let it be split at the point $x \in [a_{i-1}, b_{i-1})$, i.e.

$$x = a_{i-1} + \frac{(b_{i-1} - a_{i-1})p_i}{p_i + q_i}.$$

If $f(x) > s$, then $\beta_i = 0, a_i = a_{i-1}, b_i = x$. Otherwise, $\beta_i = 1, a_i = x, b_i = b_{i-1}$.

Notice that $\forall i, f^{-1}(s) \in [a_i, b_i)$ according to the selection of a_i and b_i. After performing k iterations, (where k is the size of the key, i.e. the number of ratios) we obtain the bit sequence $\beta = (\beta_1, \ldots, \beta_k), \beta_i \in \{0, 1\}$, which is a ciphertext for s.

4.3 Decryption

Suppose there is a bit sequence $\beta = (\beta_1, \ldots, \beta_k), \beta_i \in \{0, 1\}$, which is the ciphertext for s, encrypted with some key K. Let us consider the i-th iteration of the algorithm.

Similar to the encryption algorithm, current interval $[a_{i-1}, b_{i-1})$ is split in the ratio $p_i : q_i$. Let it be split at the point $x \in [a_{i-1}, b_{i-1})$, i.e.

$$x = a_{i-1} + \frac{(b_{i-1} - a_{i-1})p_i}{p_i + q_i}.$$

If $\beta_i = 0$, then $a_i = a_{i-1}, b_i = x$. Otherwise, $a_i = x$, $b_i = b_{i-1}$.

After performing k iterations, we obtain the interval $[a_k, b_k)$ and the condition $(f(b_k) - f(a_k)) < \frac{1}{2^n}$ is satisfied according to the key selection. As $s \in [f(a_k), f(b_k))$, the s is uniquely decoded as follows:

$$s = 2^n f(a_k) + 1,$$

where $\lfloor x \rfloor$ is the largest integer, which comes before x.

5 Scheme Modifications

5.1 Application of the Scheme for Fixed-Point Arithmetic

It is easy to see that this scheme can be generalized to the set of rational numbers. Encryption and decryption algorithms are the same except for the final operation – the length of the segment $[a_k, b_k)$ that determines encrypted number is reduced to 2^l times, where l is the number of bit decimal places. It should be known at the stage of key generation and condition from point 2 takes the following form:

$$\prod_i \frac{\max(p_i, q_i)}{p_i + q_i} * f'_{\max}(x) < \frac{1}{2^{n+l}}$$

After key generation number l can't be modified and is a part of the key. So, the secret key K now is the set $[l, (a_0, \ldots, a_9), (p_1, q_1), (p_2, q_2), \ldots, (p_k, q_k)]$.

5.2 Strictly Increasing Hash Function

This algorithm can also be modified to produce a strictly increasing hash function. It can be used, for example, in encrypted database, if it stores two entities for each data: ciphertext, that was obtained from cryptographically strong algorithm and hash value returned by hash function. This allows both to be sure that the data won't be decrypted by adversary (first entity is secure and the second can't be decrypted at all) and apply comparison operations on encrypted data to some extent.

To begin, we note that output has the same bit size as the number of ratios p_i, q_i from the secret key. So, in order to obtain a hash function, it is enough to change the procedure of key generation, and more precisely, its ratios generation part.

Instead of the condition checking from the point 2, satisfaction of which guaranteed that the data can be decrypted, now we need to perform the first point – pair p_i, q_i generation – a number of times. This number, evidently, is equal to the number of bits that hash function returns.

Thus, the key generation algorithm for order-preserving m-bit hash function is:

1. Select strictly increasing noise function f(x). To do this, generate $(a_0, \ldots a_9)$ so that

$$\left(a_0 + a_1 t + a_2 t^2\right)\left(a_3 + a_4 \sin(a_5 + a_6 t) + a_7 \cos(a_8 + a_9 t)\right) > 0$$

   for $\forall t \in (c; x_{max})$, where c is a fixed constant.
2. Generate random set of ratios $(p_1, q_1), (p_2, q_2), \ldots, (p_m, q_m)$.
3. The key is the set $K = [(a_0, \ldots, a_9), (p_1, q_1), (p_2, q_2), \ldots, (p_m, q_m)]$.

To get rid of the big numbers processing, for instance, if we need to get hash of a large file, it is possible to split input data into parts with acceptable size and calculate hash for each of them. The result hash value of the whole file can be found as their concatenation. This approach allows us to hash data of any predetermined dimension.

So, there are three parameters that we can select arbitrarily depending on our purpose: s_1 – size of the processed parts, s_2 – hash size for each of them ($s_2 < s_1$), and s_3 – maximum file size. Obviously, final hash is $\frac{s_2 s_3}{s_1}$-bit.

Since encryption algorithm remains the same, the hash function running time depends linearly on its output size (it is equal to the number of algorithm iterations). Therefore, it is not recommended to choose too-big s_2 number.

In order to process files smaller than the maximum size, they can be padded with zeros on the left. In this case, order is still preserves. Since this is a hash function algorithm, decryption is no longer exists.

6 Encryption Security

As we have seen (see Sect. 3) OPE schemes cannot satisfy the standard notions of security against chosen-plaintext attack. Different methods of cryptoanalysis are considered to determine the notion of order-preserving encryption security [2, 8–10]. Generally, the security of such schemes is based on the fact that monotonic function, the scheme is based on, must be completely indistinguishable from truly random monotonic function. This means that only an access to the private key allows performing accurate data decryption.

So let us check this algorithm for this condition in practice. To do that, we encrypted all 16-bit numbers (from 0 to 65535) with the same random key and analyzed the results.

As a subject of analysis we chose the difference between two ciphertexts for nearby integers. For example, if $f(x) = 2186003864819$ and $f(x+1) = 2186004033407$, where $f(x)$ is encryption function, then $f(x+1) - f(x) = 168588$ is considered. One of the reasons for this choice was the fact that success of chosen-plaintext attack by interpolation depends on this differences (see Fig. 2).

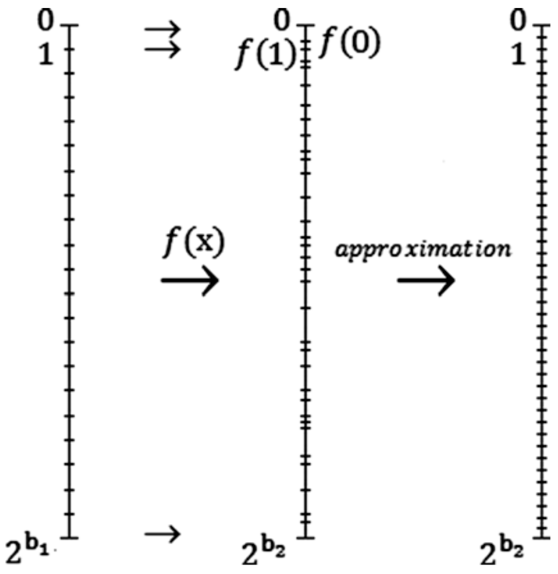

Fig. 2. Chosen-plaintext attack using values interpolation. Ciphertext for some b_1-bit plaintext x is approximated by the value of $\frac{X}{2^{b_2}}$, where b_2 is size of ciphertext. Approximation in the other direction is counted similarly.

As a result, we obtained the following data (see Fig. 3). In this chart the Y-axis displays the difference value between two ciphertexts (higher values were rounded), and the X-axis shows the number of them was found.

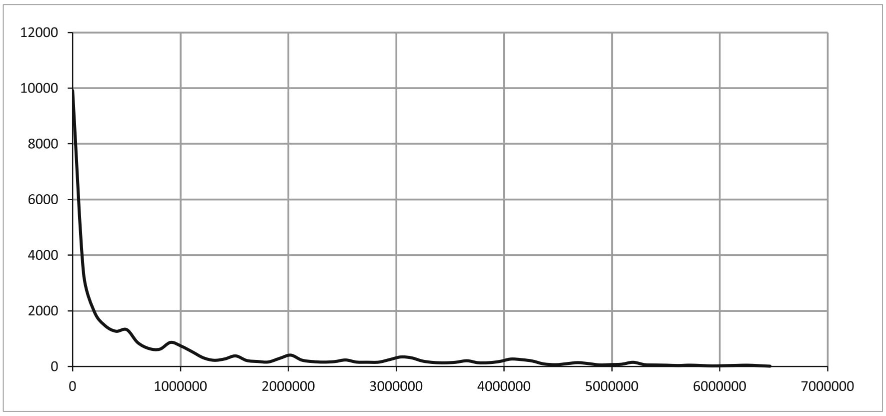

Fig. 3. Frequency distribution of the differences between ciphertext.

As we see, this chart and right hyperbola $y = \frac{1}{x}$ are alike. It is typical for monotonic functions that were generated randomly and indicates that the maximum available security of the algorithm was achieved.

But the distribution of the differences itself is also important (see Fig. 4). The Y-axis displays $f(x + 1) - f(x)$ when the X-axis shows x (from 0 to 65535).

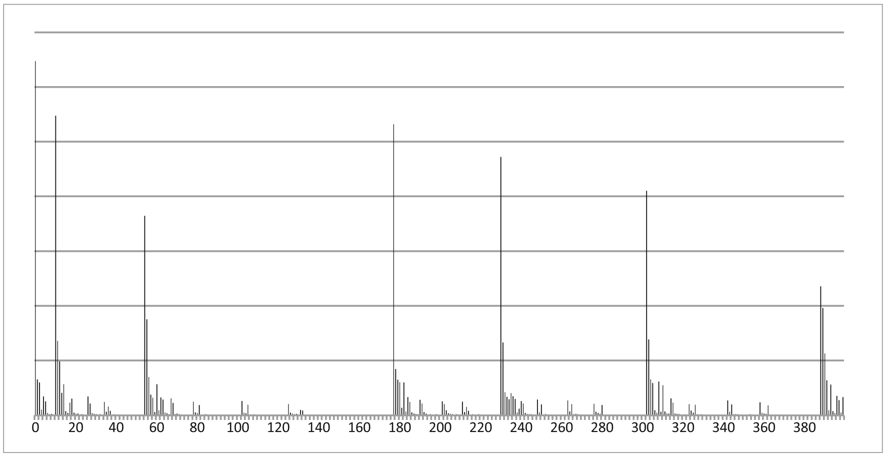

Fig. 4. Distribution of the differences on the interval.

We can see that the differences are distributed very irregularly. As it is a feature of secure encryption, we can claim that proposed algorithm is cryptographically strong.

References

1. Ozsoyoglu, G., Singer, D.A., Chung, S.S.: Anti-tamper databases: querying encrypted databases. EECS Department, Math Department, Case Western Reserve University, Cleveland, OH 44106. doi:http://dx.doi.org/10.1109/ICDEW.2006.30
2. Agrawal, R., Kiernan, G.G., Srikant, R., Xu, Y.: System and method for order-preserving encryption for numeric data
3. Kerschbaum, F.: Commutative order-preserving encryption. Karlsruhe, United States Patent 20120121080, DE, 17 May 2012
4. Bebek, G.: Anti-tamper database research: Inference control techniques. Technical Report EECS433 Final Report, Case Western Reserve University (2002)
5. Popa, R.A., Li, F.H., Zeldovich, N.: An ideal-security protocol for order-preserving encoding, MIT CSAIL (2011). doi:http://dx.doi.org/10.1109/CISS.2012.6310814
6. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-preserving symmetric encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 224–241. Springer, Heidelberg (2009)
7. Shnayder, B.: Applied Cryptology, 816 pp. Triumph, Moscow (2002)

8. Boldyreva, A., Chenette, N., O'Neill, A.: Order-preserving encryption revisited: improved security analysis and alternative solutions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 578–595. Springer, Heidelberg (2011)
9. Martinez, S., Miret, J.M., Tomas, R., Valls, M.: Security analysis of order preserving symmetric cryptography. Appl. Math. Inf. Sci. **7**(4), 1285–1295 (2013)
10. Xiao, L., Bastani, O., Yen, I.: Security analysis for order preserving encryption schemes. Technical Report UTDCS-01-12 (2012). https://utd.edu/~ilyen/techrep/OPE-proof1.pdf