doi:10.3772/j.issn.1006-6748.2020.04.013

A privacy-preserved indexing schema in DaaS model for range queries[①]

Hao Renzhi (郝任之)[*], Li Jun[②][**][***], Wu Guangjun[***]
([*] College of Control Science and Engineering, Zhejiang University, Hangzhou 310058, P. R. China)
([**] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100093, P. R. China)
([***] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, P. R. China)

Abstract

In a database-as-a-service (DaaS) model, a data owner stores data in a database server of a service provider, and the DaaS adopts the encryption for data privacy and indexing for data query. However, an attacker can obtain original data's statistical information and distribution via the indexing distribution from the database of the service provider. In this work, a novel indexing schema is proposed to satisfy privacy-preserved data management requirements, in which an attacker cannot obtain data source distribution or statistic information from the index. The approach includes 2 parts: the Hash-based indexing for encrypted data and correctness verification for range queries. The evaluation results demonstrate that the approach can hide statistical information of encrypted data distribution while can also obtain correct answers for range queries. Meanwhile, the approach can achieve nearly 10 times and 35 times improvement on encrypted data publishing and indexing respectively, compared with the start-of-the-art method order-preserving Hash-based function (OPHF).

Key words: database-as-a-service (DaaS) model, data privacy and security, data verification, range query

0 Introduction

The database as a service (DaaS) model was proposed by Hacigumus et al.[1], and NetDB2, SQLVM[2], PMEL[3] are effective DaaS models. In a DaaS model, a data owner stores encrypted data in a database server of the service provider, and the data user obtains query results from the service provider. In the DaaS model, data owner can access the data freely and can reduce the cost of deploying corresponding software and hardware of database service. Typically, a DaaS model includes 3 parts: (1) The data owner, which is the owner of data, publishes and shares data with data users; (2) The service provider provides service for the data owner and user; (3) The data user accesses and uses data stored in the service provider. Owing to the fact that a service provider cannot be trusted by a data owner, the data owner encrypts data before publishing data. Considering a relation TRADE (tno, cost, date) as an example, a data owner encrypts the original data items into encrypted data, builds index for each item, and publishes and stores them in the database of service provider. However, in current DaaS model, the published data suffers from the risk of privacy leaking to an attacker. In other words, the data owner and the data user do not trust the service provider. To support the following queries over encrypted data, the service provider usually stores the index along with the encrypted data, and an attacker might obtain data distribution and statistical information from the correspond indexing. For example, a tuple in TRADE (tno, cost, date) is recorded as (key_1, 5779520.12, 2018/12/3) and it might be represented as (Index on key_1, *kVpZCNJqcG8 =, lcXxF + LEl9TogO7ADeI/e, JHKIN + iZwiA = =*). An attacker can obtain the distribution of encrypted data from the index on tno.

A novel indexing schema is proposed for encrypted data stored in DaaS service provider. In this model, the data owner can use encryption technologies to store data at a service provider. Meanwhile, the model can avoid data privacy leaking and provide the capability of correctness verification to verify the following queries over the encrypted data. The main contributions of this paper are as follows:

A novel DaaS model using privacy-preserved Hash-based indexing schema is proposed, which is computed by summarization of Hash value from an interval of data items. The schema can support range query and avoid the false fit by a proposed marked data

[①] Supported by the National Natural Science Foundation of China (No. 61931019).

[②] To whom correspondence should be addressed. E-mail: lijun@iie.ac.cn

Received on Sep. 1, 2019

items technique.

A correctness verification method is described, which uses a verification matrix to check the answer of a range query. The verification matrix is computed by data items signature. The data users can check correctness of an answer for range query that is obtained via the proposed Hash-based indexing.

An extensive evaluation using 2 data sets to demonstrate the functionality and effectiveness of the proposed approach is conducted. The experimental results show that the approach can protect the statistical information and data distribution using the proposed schema over encrypted data.

1 The proposed approach

1.1 Approach overview

In this section, the overview of proposed approach for privacy-preserved data publishing and queries in a DaaS model is presented. In data owns' perspective, data privacy of the data owner includes data statistical information, such as maximum data item, minimum data item, and data value distribution.

This work proposes a privacy-preserved indexing schema for data owner and provides query correctness verification for data users. The core idea of the approach is shown in Fig. 1. The method mainly includes 3 stages: (1) The data owners encrypt their data and build a Hash-based indexing. Then, the data owners publish the encrypted data and built index to the service provider; (2) The service provider accepts the published wrapped data, including the encrypted data and the indexing, and stores them in an open database; (3) The data user conducts a range query in the service provider over the encrypted data, obtains the queried result, and verifies the correctness of the answer.

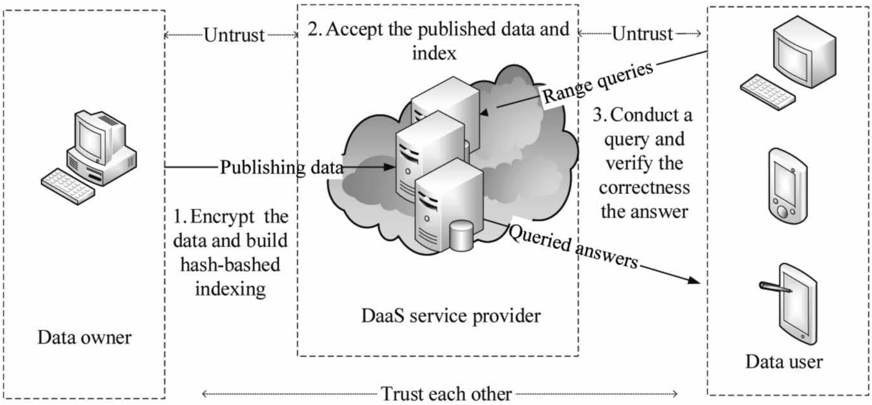
Untrust
2. Accept the published data and index
Untrust
Range queries
3. Conduct a query and verify the correctness the answer
Publishing data
1. Encrypt the data and build hash-bashed indexing
Queried answers
Data owner
DaaS service provider
Data user
Trust each other

Fig. 1 Overview of the approach in the DaaS model

1.2 Hash-based indexing

In the approach, the data owner first divides the data admin U into N intervals. Each interval U_i has N_i data items. Then the data owner assigns each internal a unique identifier. After dividing domain U into N intervals, the data owner calculates the index of data items. In an interval, when $d_i < d_{i+1}$, the schema defines Hash function $f(x)$, which satisfies $f(di) < f(d_{i+1})$. The order-preserving Hash function is defined by summing the partial Hash value in one internal. For an internal U_i, the items are $D_i = \{d_1, d_2, d_3, \cdots, d_{Ni}\}$, the Hash function satisfies $f(d_i) = \sum_1^i hash(d_i)$. Considering TRADE (tno, cost, date), the attribute cost domain $U = (0.0, 1000.0)$, and data items values $D = \{1, 1.1, 2.3, 3.45, 9.03, 34, 56\}$. Setting internal $[0, 10)$ as a domain internal, and its identifier is 5. Then when the data owner wants to publish the data item equals 1.1, the data owner will publish the encrypted data item along with identifier 5 and Hash value $f(1.1)$ to the service provider. In the schema, considering a range query $Q(a, b)$, if the data set D doesn't have the data items equals a or b, the data users cannot locate the query boundary. Therefore, the schema introduces query precision Φ. The interval U_i has the query precision Φ_i. For example, a data item equal to a is in the internal $U_i[c, d)$, and its query precision is Φ_i, and the data owner will calculate a mark data item equal to g that $c + g \times \Phi_i \leqslant a$ and $c + (g+1) \times \Phi_i > a$. Furthermore, it is important that Φ_i is higher precise than the data item in U_i. For example, if the precision of the data items is 0.1, and the Φ_i can be 0.01.

1.3 Publishing encrypted data and index

In this section, the process of publishing encrypted data and the Hash-based index in a DaaS model is represented. Supposing that the data items have index over one attribution, for a data item r to be inserted, firstly, the data owner should obtain the interval identifier, then the data owner should query all the data items in that interval from the service provider, then the data owner decrypts data, after that, the data owner inserts r into the decrypted query results, finally, the data owner calculates the Hash-bashed indexing. For an interval $U_i[c, d)$ with query precision Φ_i, when the data owner publishes a data item equal to a, the data owner will calculate a max multiple g that meets $c + g \times \Phi_i \leqslant a$. Then the data owner generates a marked item that has the same identifier with the published data item, and the data owner treats its Hash value $hash(d)$ as g and its value as $d = c + g \times \Phi_i$ when calculating the order-preserving index by $f(d_i) = \sum_1^{Ni} hash(d_i)$.

1.4 Query in a DaaS model

After representing the Hash-based indexing, the process of a range query can be described. The steps can be described as Fig. 2. For a range query $Q(a, b)$, the range query condition is expressed as $[a, b]$. The query condition can be divided into 2 parts: (1) $[a, a_1)$ and $(b_1, b]$, which contains the boundary of the query; (2) $[a_1, b_1]$, which covers the whole internals. To obtain the range query results, the data user firstly maps all the queried range into interval identifiers. For the range $[a, a_1)$, which is in one interval supposed as $[c, d)$, the data user calculates g that satisfies $c + g \times \Phi_i \leqslant a$ and $c + (g+1) \times \Phi_i > a$ to locate the boundary by value a, and the process of locating the query boundary can be described as Algorithm 1.

Algorithm 1 Computing a range boundary for a range query

```
Input: $hash(a)$, marked item $g$, hash-based index in one
       interval, sum = 0, boundary = 0
While indexes! = NULL
    index = indexes.next
    hash = index - sum
    if index is of marked items
        if hash > $g$
            return boundary
    else if hash = = $hash(a)$
        return index
    boundary = index
    sum = index
  return boundary
```

1.5 Correctness verification

To prevent data privacy leaked, the data owner encrypts the data before publishing it. In this section, the schema adopts a verification matrix to verify the query results. Along with each data item, the service provider stores a signature calculated by the data owner. The signature of a data item is determined by 2 adjacent data items as Fig. 3.

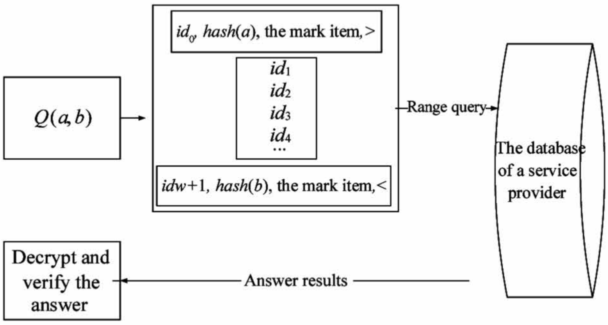
id_0, $hash(a)$, the mark item, >
id_1
id_2
id_3
id_4
...
$idw+1$, $hash(b)$, the mark item,<
$Q(a,b)$
Range query
The database of a service provider
Decrypt and verify the answer
Answer results

Fig. 2 Range query via Hash-based indexing

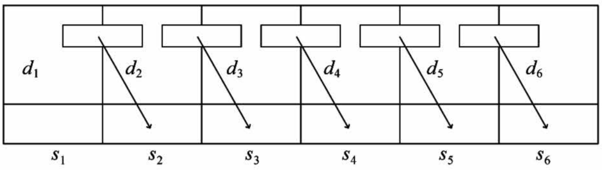

Fig. 3 The signature chain

The signature is calculated via the signature function presented as Eq. (1). In Eq. (1), $SHash(d)$ is a Hash function, and it calculates the data item' s Hash value, and $MaxP(a, b)$ calculates the maximum multiple of b less than a, and b must be a rounded number. In the Eq. (1), ε_1, ε_2 are error parameters that equal to $1/p_1$, $1/p_2$, where p_1 and p_2 are decided by the data owner.

$$S(d_i) = MaxP(SHash(d_i), 1/\varepsilon_1) + MaxP(SHash(d_{(i-1)}), 1/\varepsilon_2) \quad (1)$$

The schema defines the verification matrixes by the signatures. As described in Fig. 3, a data item d_i and its signature s_i can be verified through 3 ways: (1) The neighbor verification, which means verification by the relationship between d_{i-1} and s_i; (2) The self-verification, which means verification by the relationship between d_i and s_i; (3) The mutual verification, which means verification by the Eq. (1). For the 3 ways, the verifications can be presented as the following equation:

$$(s_i - MaxP(SHash(d_{(i-1)}), 1/\varepsilon_2)) \bmod (1/\varepsilon_1) = 0 \quad (2)$$

$$(s_i - MaxP(SHash(d_{(i-1)}), 1/\varepsilon_1)) \bmod (1/\varepsilon_2) = 0 \quad (3)$$

$$s_i = (MaxP(SHash(d_i), 1/\varepsilon_1)) + MaxP(SHash(d_{(i-1)}), 1/\varepsilon_2)) \quad (4)$$

The errors of verification Eq. (1) and Eq. (2) are ε_2, and ε_2, respectively, and $\varepsilon_1 \cdot \varepsilon_2$ for verification Eq. (3). The schema defines the verification matrix $\boldsymbol{Au}$:

$$\boldsymbol{Au} = \begin{bmatrix} au_{11}, & au_{12} \\ au_{21}, & au_{22} \end{bmatrix}$$

$$[au_{11}, au_{12}] = [s_{11}, s_{12}, s_{13}] \times \begin{bmatrix} \beta_1, & \beta_1 \\ \beta_2, & 0 \\ \beta_3, & \beta_1 \end{bmatrix},$$

$$[au_{21}, au_{22}] = [s_{21}, s_{22}, s_{23}] \times \begin{bmatrix} \beta_1, & 0 \\ \beta_2, & \beta_1 \\ \beta_3, & \beta_1 \end{bmatrix}$$

The ($s_{ij} \in \{0, 1\}$) equals to 1 if the data item or the verification signature satisfies the Eqs(2), (3), and (4) respectively, otherwise, s_{ij} equals 0. And $\beta_1 = 1 - \varepsilon_1$, $\beta_2 = 1 - \varepsilon_2$, $\beta_3 = 1 - \varepsilon_1 \cdot \varepsilon_2$, and define the matrix operation ' $*$ ' appearing in Eq. (5).

$$\boldsymbol{Ma} = \boldsymbol{Mb} * \boldsymbol{Mc}, \text{ where } Ma_{ij} = \max(Mb_{ik} \cdot Mb_{kj}) \quad (5)$$

In the verification matrix of the data item d_i, au_{11} indicates the confidence of correctness of the data item d_i, au_{12} for the neighbor data item d_{i+1}, au_{21} for s_i, and au_{22} for d_{i-1}. Based on the verification matrix $\boldsymbol{Au}$, the data user can verify that the data item d_i is correct with the confidence au_{11}, and the data user can confirm with the confidence $au_{21} \cdot (1 - au_{22})$ that the data item d_{i-1} is lost and with the confidence $au_{11} \cdot (1 - au_{12})$ that the data item d_{i+1} is lost. Furthermore, the verification method can be extended to a two-dimensional zone query.

1.6 Experimental setup

This work uses 2 types of data sets to evaluate the approach. One data set is the generated relational data TRADE (tno, cost, date), which has 10 000 data items, and the attributes cost is from 0 to 10 000 000 with precision 0.01 and attributes tno is an integer attribute. Another data set is a real data set, which comes from the Labor Statistic Public Database. This work conducted the experiment in a computer with Intel (R) Core (TM) i5-4210M CPU @ 2.60 GHz, 8 G RAM. The software environment is Ubuntu 17.10, JDBC, and JAVA 1.8, and the DBMS is the PostgreSQL 9.5. Based on the experimental environment, this work conducted the following experimental tests: (1) Data publishing evaluation; (2) Range queries via Hash-based indexing; and (3) Data correctness verification.

1.7 Hash-based indexing schema evaluation

Next, this work verifies the functionality of the privacy-preserved indexing on the simulated data set TRADE, whose attribution tno data items are randomly generated from a linear distribution as shown in Fig. 4, and Fig. 4 shows that the distribution of the index obtained by order-preserving Hash-based function (OPHF)[4] is similar to the data value. Fig. 5 shows the original data value distribution and the Hash-based index obtained by the approach, and the Hash-based index of the approach can protect the privacy and statistical information of the data value.

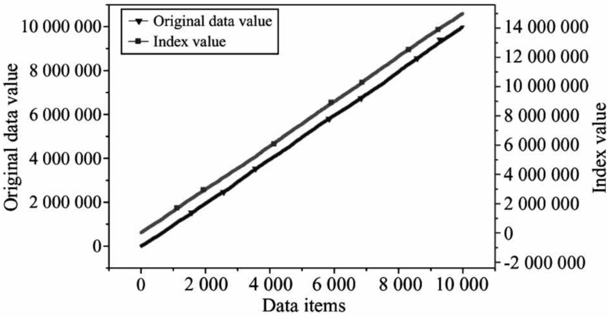

Fig. 4 Original data distribution vs. index distribution in OPHF[4]

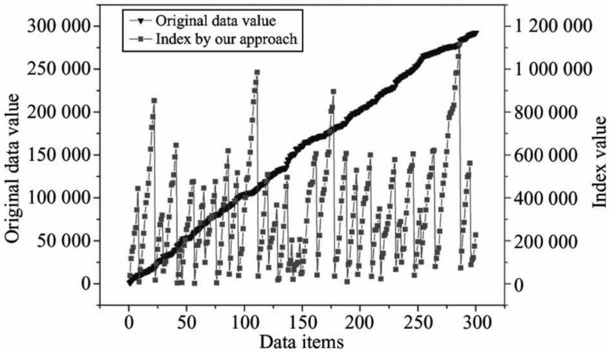

Fig. 5 Original data distribution vs. the Hash-based index

The evaluations use public dataset from the Bureau of Labor Statistics to conduct the evaluation. Fig. 6 and Fig. 7 present the original data value and Hash-based index by the approach, and the figure demonstrates that the distribution of original data and the Hash-based index by the approach are uncorrelated. The attacker can't defer the data information via the encrypted data or the Hash-based index accurately.

1.8 Correctness verification evaluation

To test the efficiency of publishing the data items and conducting a range query, this work conducts the experiments on the simulated data set TRADE. Table 1 shows the efficiency of the publishing data items. The data set TRADE includes 100 000 items. This work inserts the data items one by one and calculates the average time for per item. When the number of intervals N varies, the time of the approach almost keeps the same as shown in Table 1. Table 1 demonstrates that the approach is more efficient than OPHF[4] whose parameter $(x_i - x_{i-1})$ is assigned 0.01. The evaluation demonstrates that the approach has better efficiency on data publishing and indexing. In the testing, the approach can achieve nearly 10 times and 35 times improvement on data encryption (decryption) and indexing respectively.

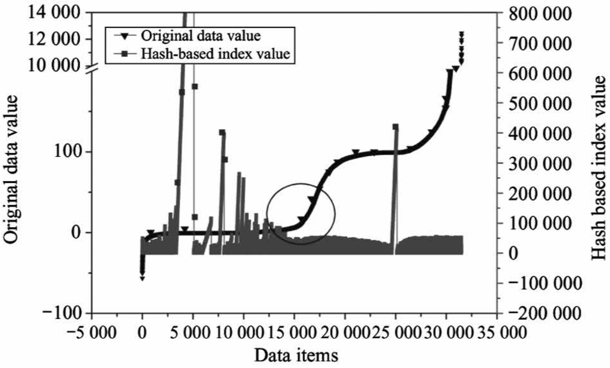

Fig. 6 Overview of the original data value vs. Hash-based index value

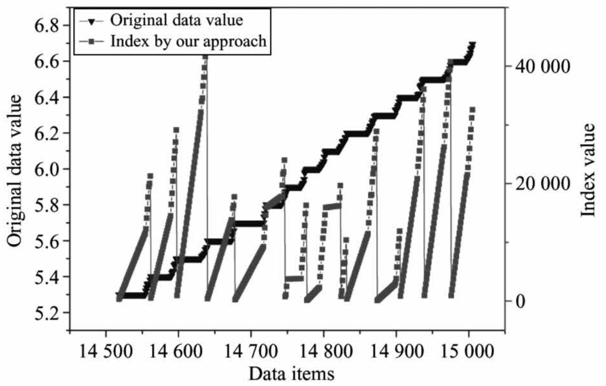

Fig. 7 Partial view of the original data value vs. Hash-based index value

Table 1 Time used for publishing and indexing data

Method	Number of intervals	Encryption and decryption (ms)	Indexing (ms)	Query in service provider (ms)
The proposed approach	500	0.52868	0.3628	54.01606
	1 000	0.45218	0.22302	53.9679
	2 000	0.40784	0.14428	52.13378
OPHF	/	4.842	4.93	22.096

This work conducts the range query evaluations, in which a data user requests the range queries that cover 20%, 40%, 60%, and 80% of dataset TRADE. The process of a range query includes 4 parts as labeled in Fig. 8. Step 1 is the process that the data user maps the interval identifiers and calculates the mark data items, Step 2 is the process that works in the service provider, Step 3 is the process that the data user filters the mark items, and Step 4 is the process that the data user encrypts and decrypts data items. Fig. 9 is the process that the data user calculates the signature matrix and verifies the query results. In the DaaS model, the most computation is calculated by the service provider and the process that is executed by the data user uses less proportion time, which reflects the advantage of the DaaS model, and Fig. 9 also demonstrates that the process of correctness verification costs reasonable time consumption in the approach.

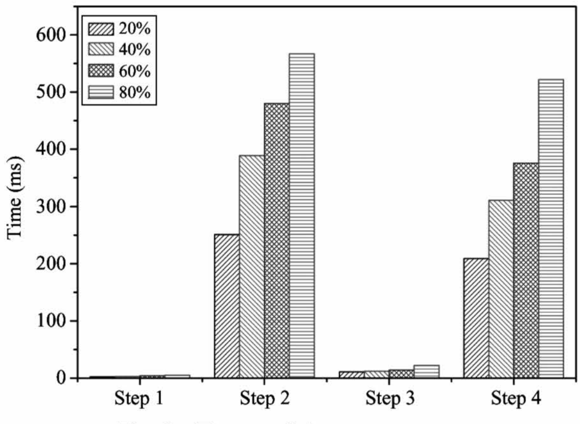

Fig. 8 Time used for a range query

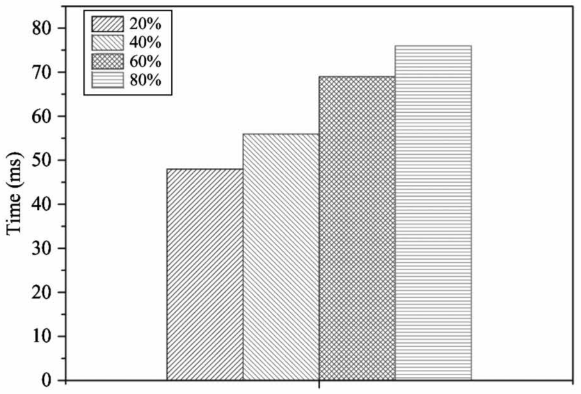

Fig. 9 Time used for correction verification

2 Related work

The database as a service (DaaS) was proposed by Hacigumus et al[1], and they have developed a DaaS instantiation called NetDB2. Some researches demonstrate privacy issue in the DaaS model[1,5,6], and many researches focus on the solution of the problem[7,8]. To keep data security, the data owner set up a reasonable access control mechanism, such as role-based access control (RABC)[9], attribute-based access control (ABAC)[10], discretionary access control (DAC), and mandatory access control (MAC)[11]. Many encryption algorithms, such as DES[12], 3DES, Blowfish[13], RC5[14], IDEA[15], RSA, and Elgamal[16] will be used in DaaS model.

Order-preserving data encryption methods By adopting an order-preserving encryption, the encrypted data would meet $Encrypt\ (d_1) < Encrypt\ (d_2)$, in the case of $d_1 < d_2$. Through the order-preserving encryption[17-21], the range queries can be conducted easily. The encrypted data would still leak privacy information by the index, such as the min and max item, and the distribution of data items, etc. The order-preserving encryptions are also time-consuming[22].

Indexing methods The bucket partitioning[23] is an effective approach to support the range query on encrypted data. The primary idea of bucket partitioning method is that the data owner divides data domain U into N discrete buckets, and each bucket has an index value. However, the data user might get the false hit data. Order-preserving Hash-based function (OPHF)[4] can be adopted to support range query. This method has a risk of privacy leaking, such that data distribution of the data will be exposed by the Hash index.

Correction verification The service providers wouldn't be trusted absolutely, so it is important to verify the correctness of the query results[24,25]. On the verification of data correctness, the data owner and the data user share a matrix to verify query results[4]. However, the matrix can just show whether there are data items added or deleted. However, when a data item is corrupted, the matrix can't show this situation. There are some researches using Merkle Hash tree[25] to verify query results.

3 Conclusion

The DaaS model makes it easy to share and manage data between different data users. To keep data privacy and security for data owners, the DaaS model needs an efficient and secured strategy to ensure data privacy in the process of data publishing, storage, and sharing between users. This work proposes a novel indexing schema to satisfy privacy-preserved data management requirements. The proposed approach includes 2 parts: the Hash-based indexing for encrypted data and correctness verification for range queries between the DaaS service provider and the data users. The evaluation results demonstrate that the proposed approach can hide statistical information of encrypted data distribution while can also obtain correct answers for range-queries. The future plan is to promote query efficiency over publishing data items.

References

[1] Hacigumus H, Iyer B, Mehrotra S. Providing database as a service[C] // Proceedings of the International Conference on Data Engineering, San Jose, USA, 2002: 29-38

[2] Narasayya V, Das S, Syamala M, et al. A demonstration of SQLVM: performance isolation in multi-tenant relational database-as-a-service[C] // Proceedings of the ACM SIGMOD International Conference on Management of Data, New York, USA, 2013: 1077-1080

[3] Burger E F, Schweitzer R, O'Brien K, et al. Common data servers as a foundation for specialized services[C] // American Geophysical Union Fall Meeting Abstracts, New Orleans, USA, 2017: 12A-08

[4] Chen F, Liu A X. Privacy and integrity preserving multi-dimensional range queries for cloud computing[C] // IFIP Networking Conference, Trondheim, Norway, 2014: 1-9

[5] Agrawal D, Abbadi A E, Emekci F, et al. Database management as a service: challenges and opportunities [C] // Proceedings of the IEEE International Conference on Data Engineering, Shanghai, China, 2009: 1709-1716

[6] Gallagher J, West P, Potter N, et al. Software architectures expressly designed to promote open source development: using the hyrax data server as a case study[J]. *ACM Transactions on Software Engineering and Methodology*, 2004, 11(11): 309-346

[7] Nicolas A, Mehdi B, Luc B, et al. Querying visible and hidden data without leaks[C] // Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, 2007: 677-688

[8] Ünay O, Gündem T I. A survey on querying encrypted XML documents for databases as a service[J]. *ACM SIGMOD Record*, 2008, 37(1): 12-20

[9] Sandhu R S, Coyne E J, Feinstein H L, et al. Role-based access control models[J]. *Computer*, 1996, 29(2): 38-47

[10] Hu V C, Kuhn D R, Ferraiolo D F. Attribute-based access control[J]. *Computer*, 2015, 48(2): 85-88

[11] National Computer Security Association. Department of Defense Trusted Computer System Evaluation Criteria [M]. London: Palgrave Macmillan, 1985

[12] Diffie W, Hellman M E. Special feature exhaustive cryptanalysis of the NBS data encryption standard[J]. *Computer*, 1977, 10(6):74-84

[13] Schneier B. Description of a new variable-length key, 64-bit block cipher (Blowfish) [C] // Fast Software Encryption, Cambridge Security Workshop Proceedings, Cambridge, UK, 1993: 191-204

[14] Rivest R L. The RC5 encryption algorithm[J]. *Lecture Notes in Computer Science*, 1994, 1008: 86-96

[15] Zimmermann R, Curiger A, Bonnenberg H, et al. A 177 Mb/s VLSI implementation of the international data encryption algorithm[J]. *IEEE Journal of Solid-State Circuits*, 1994, 29(3):303-307

[16] Gamal T E. A public key cryptosystem and a signature scheme based on discrete logarithms[J]. *IEEE Transactions on Information Theory*, 1985, 31:469-472

[17] Agrawal R, Kiernan G G, Srikant R, et al. System and method for order-preserving encryption for numeric data [P]. US patent: 7873840, 2008

[18] Diffie W, Hellman M E. Multiuser cryptographic techniques[C] // American Federation of Information Processing Societies, New York, USA, 1976: 109-112

[19] Boldyreva A. Order-preserving symmetric encryption[J]. *Advances in Cryptology*, 2009, 5479: 224-241

[20] Malkin T, Teranishi I, Yung M. Order-preserving encryption secure beyond one-wayness[J]. *IACR Cryptology ePrint Archive*, 2013, 2013: 409

[21] Teranishi I, Yung M, Malkin T. Order-preserving encryption secure beyond one-wayness [C] // International Conference on the Theory and Application of Cryptology and Information Security, Taiwan, China, 2014: 42-61

[22] Gamal T E. A public key cryptosystem and a signature scheme based on discrete logarithms[J]. *IEEE Transactions on Information Theory*, 1985, 31(4): 469-472

[23] Hacigümüs H, Iyer B R, Li C, et al. Common data servers as a foundation for specialized services [C] // Proceedings of the ACM SIGMOD International Conference on Management of Data, Madison, USA, 2002: 216-227

[24] Pang H H, Jain A, Ramamritham K, et al. Verifying completeness of relational query results in data publishing [C] // Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, USA, 2005: 407-418

[25] Merkle R C. A Certified digital signature[J]. *Lecture Notes in Computer Science*, 1989, 435: 218-238

Hao Renzhi, born in 1999. He is currently studying in College of Control Science and Engineering of Zhejiang University. His research interests include data mining and artificial intelligence.