A Method of Encrypting and Querying String Data for Database

Baohua Huang[1, a], Tianjing Wang[1, b] , Jiafeng Wei[1, c]

[1]College of Computer Science and Electronic Information,

Guangxi University, Nanning 530004 China

[a]email: Bhhuang66@gxu.edu.cn, [b]email:1154765126 @qq.com, [c]email: 974915678 @qq.com,

Keywords: Database Encrypting; String Data; Partition Value; Character Information Value; Two-phase Query

Abstract. As the information era is coming, more and more sensitive information will be encrypted into databases for secure accessing. This paper focuses on encrypting String data in databases and provides a method to build index for sensitive attributes. The index is composed of partition value, characteristic value and character information value. So the index can be used to filter most of unmatched records, and reduce the number of records need decode. Two-phase query is used in the proposed method. The first phase executes query on the index to filter most unneeded records; the second phase decodes the result set and executes exact query on it.

Introduction

As the network has risen in popularity and the application is constantly updated, people propose some application model. On the database side, the DAS(Databases-AS-Service) model has been paid much attention to. In the DAS model, the enterprise will outsource the work of database management to third-party service providers. Outsource the database management can effectively reduce the consumption of local resource. However, it also means that third-party service provider can get the sensitive information without the data owner's permission, which can make the data unsafe. In order to guarantee the data's safety, user usually chooses to encrypt the sensitive information before deliver it to the third-party service providers. Third-party service providers can't understand the data's meaning after the encryption even get the data .However, the method also will lead to some question, for example when the user only want to retrieve the data that satisfy the query statement, but the server will return a large number of encrypted data to user, so the user has to decode the encrypted data before perform the SQL query on the data, which will cause the waste of time and space. Therefore, improve query efficiency on encrypted data under the premise of safety is the key issue in database field.

In order to solve above problem, this paper propose a method to do two-phrase query on encrypted string data in database on the basis of existing research. It can filter out most of mismatch records; in addition, compared with the old method, the new method can improve the query efficiency and the sensitive data's safety.

Related research

So far, in the field of ciphertext retrieval, most researches are focus on numeric data[1],the researches about string data are very few[2][3].Wang [4]propose a method to create index according to the string data in database, then use the index to do two-phrase query on string data. The method split the string into dual coding, then map the dual coding to a string of binary bit string. The method can fulfill fuzzy search on encrypted string data, but can't fulfill the more and less query; Cui[5]combine the method proposed by Hacigumus[6]with the method proposed by Wang[4]to put forward a new method and the new method can support range query on string data, but Cui didn't discuss the way to determine the partition value in his paper. Zhang propose a new method to create index for string data.The index generated by compress a information matrix and we can do fuzzy query and range query on string data. However, Zhang's method exists some inadequacy, for

example, the information matrix only use”0” ,”1”to represent character information and almost no consideration for the character frequency. What’s more, if the characteristic value that generated by compress the information matrix is too short, it will lead to false detection, if it is too long, will lead to security issue. This paper is based on Zhang’s paper to propose a method to reduce the false detection and keep stable filtering efficiency even characteristic value is short.Zhang’s[7]method is called old method in this paper.

System architecture of encrypting store and query

Usually, there are three way to achieve encryption of data, the first one is to develop a new database management system with encrypting function; the second one is to reform the inner structure of existing database management ,extend the database management to have the encrypting function. But apparently, the two way isn’t appropriate for us, because it is difficult to develop a new database management system and the inner structure of existing database management system is very complex, it is not an easy work to reform it. Therefore, we choose the third way, which is to add an encrypting/decrypting layer between application and database management system, so we neither need to change the database management system nor application. The architecture of encrypting store and query on string data is showed in figure1,the metadata represent some parameters that need by encryption and process of convert SQL statement. When querying the database, the system will use the metadata to turn plaintext SQL statement into ciphertext SQL statement. The temporary table store the decrypted result set after first phrase.

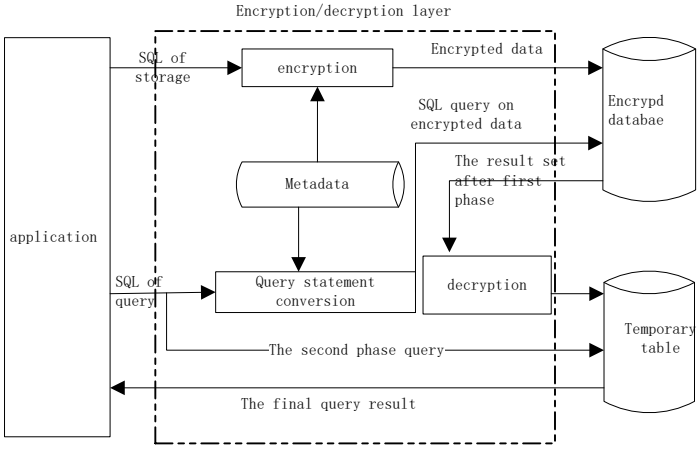

Fig .1. system architecture of encrypting store and query

The realization of encrypting store and querying on string data

This paper mainly describe two question, the first one is how to store the encrypted string data, the another one is how to query on the encrypted string data. In the field of storing encrypted string data, the major work is to generate index for sensitive data, then store the index and data together into database; in the field of querying encrypted string data, the major work is to turn plaintext SQL statement into ciphertext SQL statement.

Storage schema

Assume the old relation schema is $R(X_1,X_2,\cdots, X_r,\cdots,X_n)$,X_r is the attribute that need to be encrypted, after encryption, the old relation schema will be store in database in new form $R(X_1,X_2,\cdots,X_r^E,\cdots,X_n,X^S_r)$.$X_r^E$ is the ciphertext that generated by attribute X_r. $X_r^E=E(X_r)$,E represent the encryption algorithm,X_r^S is a new attribute that represent the index of X_r.The index is composed of the partition value, characteristic value, character information value. We will use the X_r^S to do query on the X_r^E. In the following section, we will make detailed argumentation about how to create an index and how to store the index.

Create index for string data

Because index is composed of characteristic value, character information value and partition value, we will describe the generation process of the three part first,then describe the process of storing encrypted string data.

Algorithm 1 generation of characteristic value

Input: string S that need to be encrypted; Output: characteristic value of string S

(1)apply the method that discussed in paper[7] to create an information matrix for string S, but there are some difference with the old method, we make the first row in the matrix to represent the frequency of the character. In the old method, it only use'0','1' to express the character exists in the string, which can lead to some problem. For example, the string "bba" and "bbba"is different ,but they correspond to the same information matrix.

For instance, suppose the universal set of character is{a,b,c,d},str="cbdda",creat a 7×4 information matrix for str, the matrix is showed as follows:

$$H=\begin{bmatrix} 1 & 1 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

(2)compress the matrix. Because if the account of characters in universal set are too many, the storage space cost by information matrix will be large. So inspired by the Bloom Filter, we compress each '1' in the information matrix in several locations of a bit string by use k hash functions. The location of each '1' in the matrix $H_{a\times b}$ is marked as location(i,j), the relevant formula as follows:

$$location(i,j) = (i-1)\times b + j - 1$$

$$location(i,j) \in [0, a\times b)$$

Use hash function to compute the location(i,j), then output a bunch of fixed length hash code. For example, if the hash function is SHA-1,it will turn the location(i,j) to 160-bit hash code, divide the 160-bit hash code into k group(k can be divisible by 160),then make the value of each group mod m, map the results to the corresponding position of characteristic information fields and set 1 to those position.

(3)output the fixed-length characteristic value

Algorithm 2 the generation method of character information value

Input:create information matrix for string S; Output: character information value

(1)read the first row of information matrix M[0][i](0≤i<b).

(2)use secret disturb function F to make the M[0][i] rearrange to a new string V. For example, if the effect of disturb function is make the element in the collection plus 4,3,2,1 in turn. Let us suppose the value of first row is{1,2,3,4},after the effect of function F, the value of first row is{5,5,5,5},V="5555".

(3)Output the character information value V

Algorithm 3 the generation method of partition value

The old method has defined the first and last character in the information matrix to distinguish the query types%s,s%,%s%, but because of the hash conflict,it usually can't work

well .Therefore,we add the partition value for the first and last character, it can distinguish those query type better.

Input: the string S need to be encrypted; Output: partition value

(1)Count of attribute value V_i appeared in S and its frequency f_i,let us suppose the range of attribute value X_i has been divided into m interval in accordance with the safety requirement, this can be represented as partition($R.X_i$)={P_1,P_2，⋯，P_m}.In order to avoid statistical attack, the total of frequency in each interval is approximately equal to the $\sum f_i / m$.For example, suppose X_i is divided into{(a.-c.),(d.-h.),(i.-o.),(p.-r.),(s.-z.)},{(.a-.e),(.f-.i),(.j-.p),(.q-.s),(.t-.z)} according to the frequency of the first and the last character. a. is represent that the first character is a, .a is represent that the last character is a.

(2)Each interval need to be allocated a identifier, marked as ident $R.X_i(p_j)$

(3)Map the string S to its identifier, the process can be shown as $Map_{R.Xi}(S)=ident_{R.Xi}(P_j)$, $Map'_{R.Xi}(S)=ident_{R.Xi}(P_j)$ .Map is about first character, Map' is about last character. As shown in the table 1 and table 2, $Map_{R.Xi}$(apple)=2， $Map'_{R.Xi}$(apple)=0

(4)Output the identifier of S $Map_{R.Xi}(S)$， $Map'_{R.Xi}(S)$

Table 1. divide according to first character

P_j	$Ident_{R.Xi}(P_j)$
(a.-c.)	2
(d.-h.)	7
(i.-o.)	5
(p.-r.)	1
(s.-z.)	4

Table 2. divide according to last character

P_j	$Ident_{R.Xi}(P_j)$
(.a-.e)	0
(.f-.i)	3
(.j-.p)	6
(.q-.s)	9
(.t-.z)	1

The storage process of string data

Through the above definition,we can make a summary of the stored procedure of sensitive string data:

(1) according to algorithm 1,generate the characteristic value S^{s1} for the string S that need to be encrypted .

(2) according to algorithm 2, generate the character information value $S^{s2}$for the string S that need to be encrypted.

(3) According to algorithm 3,generate the partition value S^{s3} for the string S that need to be encrypted.

(4) Output the index $S^s=S^{s1}+S^{s2}+S^{s3}$ and store it into the database.

(5) Encrypt the string S, store its cipher S^E in database.

Query method for encrypted string data

The solution use two-phase query method, in the first phase, we can filter out most of mismatch records by use the index; in the second phase, we decrypt the result set of the first phase,then do SQL query on the decrypted result set. The main emphasis of two-phase query is turn the plaintext SQL statement into ciphertext SQL statement.

(1) fuzzy query: such as X_i like s%, X_i like %s,X_i like %s%,X_i like'[A]s' ([A]represent a character interval),the basic idea to convert 'like' statement is to generate the index for the string in the 'like' statement by use the method described in 2.3.3,then compare the index with the index in database. Following is a example

<1>if the query statement include a wildcard ,such as X_i like '%s', X_i like 's%', we can extract the s from the statement, then use algorithm 3 to generate the partition value $Map_{R.Xi}(s)$ for s, use algorithm 1 to generate the characteristic value t' for s, use algorithm 2 to generate the character value c' for s(s% need map the first character of s to the information matrix, %s need map the last character of s to the information matrix). Firstly, select the record that match the $Map_{R.Xi}(s)$ from database.Secondly, make t' bitwise-and with the characteristic value T of X_i^s , make the character information value C of X_i^s compared with c',if the result of bitwise-and is t' and C>c', so the record can be moved to next phase. This can be shown as:

Trans(X_i like S)→$X_i^s(M)=Map_{R.Xi}(S)$ and $X_i^s(T)\&t'=t'$ and $X_i^S(C)>c'$, $X_i^s(M)$, $X_i^s(T)$, $X_i^S(C)$ are the partition value,characteristic value,character information of index.

If the query statement is %s%, the filter condition of first phase can be shown as:

Trans(X_i like S)→ $X_i^S(T)\&t'=t'$ and $X_i^S(C)>c'$

<2>If the query statement is X_i like'[A]s',X_i like 's[A]',convert it to several 'or ' query statement. For example, suppose the query statement is X_i like '[A]s' and the element in [A] is shown as e_i ,the number of elements is|A|, so the query statement can be convert to $X_i=e_1||s$ or $X_i=e_2||s$ or $\cdots X_i=e_i||s\cdots$or $X_i=e_{|A|}||s$, || is the connector, $e_i||s$ can be marked as s_i , the statement after conversion can be shown as following:

Trans(X_i like '[A]s')→

$X_i^S(M)=Map_{R.Xi}(s_1)$ and $X_i^s(T)\&t'(s_1)=t'(s_1)$ and $X_i^s(C)>c'(s_1)$ OR

$X_i^S(M)=Map_{R.Xi}(s_2)$ and $X_i^s(T)\&t'(s_2)=t'(s_2)$ and $X_i^s(C)>c'(s_2)$ OR

$X_i^S(M)= Map_{R.Xi}(s_{|A|})$ and $X_i^s(T)\&t'(s_{|A|})=t'(s_{|A|})$ and $X_i^s(C)>c'(s_{|A|})$

$t'(s_i)$ represent the characteristic value generated by s_i, $c'(s_i)$ represent the character information value generated by s_i.

(2)Exact-match query: such as $X_i>v$, $X_i=v$, $X_i<v$(v is the string that the query need to match)

<1> If the query statement is $X_i=v$,firstly, we select the records that is belong to the interval $Map_{R.Xi}(v)$,then use the algorithm1 to generate the characteristic value t'(v) of v and use the algorithm 2 to generate the character information value c'(v) of v. This can be shown as following:

Trans($X_i=v$)→$X_i^S(M)=Map_{R.Xi}(v)$

and $X_i^S(T)\&t'(v)=t'(v)$

and $X_i^S(C)=c'(v)$.

<2>If the query statement is $X_i>v$, the length of string v is L,convert the statement to L+1 'like'query and do 'OR' operation together.

For example, suppose the complete set of character is 26 lowercase letters, the query statement is X_i>'efg', convert it to the statement where X_i like'[f-z]%' OR X_i like 'e[g-z]%' OR X_i like 'ef[h-z]%' OR X_i like 'efg%'.

<3>If the query statement is $X_i<v$,the length of string v is L,convert it to L 'like' query and do 'OR' operation together.

<4>If the query statement is $X_i\geqslant v$, convert it to the compound query $X_i>v$ OR $X_i=v$,then use above method convert it to the query on index.

<5>Similarly, if the query statement is $X_i\leqslant v$,convert it to the compound query $X_i<v$ OR $X_i=v$,then use above method to convert it to the query on index.

Algorithm analysis

The security of data storage is very important for database management system. The security of a encryption algorithm depends on the algorithm itself, it is beyond the scope of this article. We only discuss the security of index in this paper. Our analysis focus on whether the index can reveal the plaintext, except the security, we also briefly discuss the effectiveness and storage capacity in this

section.

Security analysis

In this paper, the index of sensitive data is composed of partition value, characteristic value ,character information value. For partition value, we divide the sensitive data depend on their frequency of occurrence, the number of records in each group is nearly equal, so the attacker is hard to analysis the relation between partition value and each group by use statistical attack. For characteristic value, it is the result that use the position of non-zero element in information matrix to do k hash compute. If the attacker only know the characteristic value, it is hard to infer the plaintext, but if the attacker also know the generation method of characteristic value and some plaintext, so when the digit of characteristic value m is large and hash conflict is less , the attacker is easy to get the relation between some character pair and its hash position, which can help the attacker to infer other sensitive information. However, in this paper, we add character information value in the index. We can keep the filter efficiency even the digit of characteristic value m is small, which can guarantee the security. For the character information value, we use the disturb function to dispose the character information, so the attackers can't know the frequency of character even they has got the character information value.

Filter efficiency analysis

The filter efficiency mainly depends on the number of mismatch records that contained in the result set of first phase. The fewer mismatch records is contained ,the fewer data need to be decrypted in second phase, the fewer filter time is required, the better filter efficiency the system has. In the old method, when create the information matrix for sensitive data and compress the matrix can lead to some mistake. In the new method, we record the frequency of character in the information matrix first row, which can decrease the false detection that generate in the procedure of create information matrix. For example, the information matrix of string 'abbc'and'abbbbc' are same in the old method, but in the new method above problem can be avoided by recording the frequency of characters in the matrix's first row. What's more, add the partition value in the index can decrease the false detection caused by hash conflict, through the multiple filtering ,the new method improve the efficiency of query.

Storage space analysis

Add the index column in the encrypted data table will increase the storage space. There is a direct ratio relationship between the increase storage space and the length of the index. The longer of the index, the larger the storage space needed, but the better filter efficiency the system will has. With the continuous improvement of storage capacity expansion technology, people can accept to cost some storage space to increase the filter efficiency.

Experiment results

The aim of the experiment is to test the time performance and filter efficiency of the new method, then compare them with the old method. The experiment apply the dbgen to create 8 tables, the scaling factor is 0.1 and the size of database is 100M, the p_name column in part table as the sensitive attribute that need to be encrypted. The encryption algorithm used in the experiment is AES, the experiment system is coded in java, the experimental environment is windows7 , the experimental database is MSSQL2005,the computer configuration is intel CPU 2.40GHz, the memory is 4G. In the experiment, the times of hash is 4 and suppose the query statement is select * from part where p_name='aquamarine%'. There are two part in our experiment. The first part is to test the relation between the filter efficiency and the length of characteristic value m ,then compare it with the old method, because the method of query after decryption don't have filter function, so

its filter efficiency is 0.The second part is to test the relation between query time and the length of character value m, then compare it with the old method and the method of query after decryption. In order to get a accurate result, each experiment is repeated 10 times and take the average.

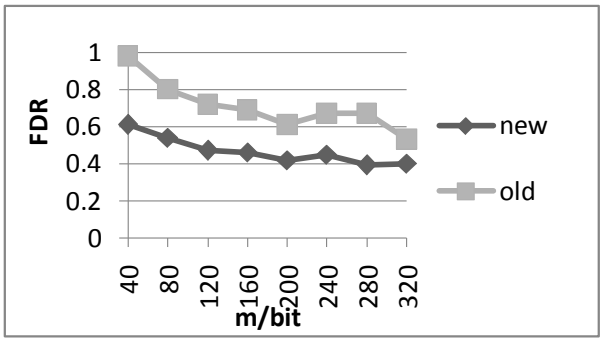

Fig.2. the variation of fault detection rate

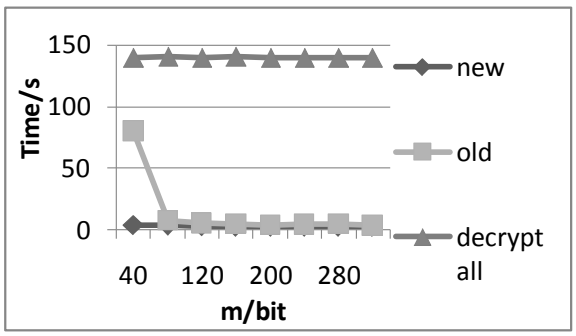

Fig.3. the variation of time performance

Filter performance test

We use following false detection rate(FDR) formula to evaluate the filtering effect, n_1 represent the number of records that after first phase filtering,n_2 represent the number of final records that after second phase filtering. The closer the false detection rate is to 0,the better filtering efficiency the system has.

$$\text{FDR}=\frac{n_1 - n_2}{n_2}$$

It can be seen from Fig. 2 that the filtering efficiency of the new method and old method all increase ,as the length of characteristic value m is increased. The filtering efficiency of new method is higher than the old method. Especially, when the query statement is ‘s%’’%s’ or the statement contain the same character repeatedly ,the new method can have the lower false detection rate. In other situations, when the length of characteristic value m is short, the new method’s false detection rate is obviously lower than the old method. As the length of characteristic value m is increased, the two method’s filtering efficiency is nearly. However, the old method mention that if the m is greater than kn/(ln2)(n represent the occurrence times of 1 in information matrix, k represent the times of hash), the security of system will be reduced, so we add character information value and partition value in the new method. The new method can keep stable filtering efficiency even the length of characteristic value m is short.

Time performance test

See Fig. 3, compare with the method of query after decryption, the old method and the new method all improve the time performance. The query time of the old method is tending towards stability, when the length of characteristic value m is increased to kn/(ln2) .However, use the new method can keep the stability of query time even the length of characteristic value is short. What’s more, compared with the old method, it also have a slightly improvement in time performance.

Conclusion

This paper proposes a method for encrypting store and query string data in database. Through transform the information matrix and add the filter condition, the new method can increase the filtering performance and distinguish the query ‘%s’,’s%’. Experiments show that the new method has a good filtering effect under the condition that the length of character value m is short, which can guarantee the system’s security.

The proposed method only support the encrypted string data in database .The future research is to find a method to support other kinds of data.

Acknowledgement

This work was sponsored by the National Natural Science Foundation of China (61262072).

References

[1]Satoshi Wang P, Ravishankar CV, Secure and efficient range queries on out-sourced databases using rp-trees,In ICDE, Brisbane, Australia,2013:314- 325.

[2] Kuzu M.,Islam M S.,Kantarcioglu M. Efficient similarity search over encrypted data[C].//Proc of the the IEEE28th International Conference on Data Engineering,2012:1156-1167.

[3] Chuah M,Hu W.Privacy-aware bedtree based solution for fuzzy multi- keyword search over encrypted data[C]//Proc of the 31st International Conference on Distributed Computing Systems workshops,2011:273-281

[4]Wang Zhengfei,Wang Man,Wang Wei.Encrypted storage and query over string data in database[J].Computer research and development,2004,41(suppl):66-71.

[5]Cui Binge,Liu Daxin,Wang Tong.Practical techniques for fast searches on encrypted string data in database.[J]Computer science.2006,33(6):115-118

[6] Hacigumus H, Lyer B ,Li Chen,et al. Executing SQL over encrypted data in the database server provider model [C]//Proc of ACM SIG-MOD ,2002:216-227

[7] Zhang Jing Zhu Zhixiang,Du Jianbo.Encryption storage and query of character data in database.[J]Xi’an university of posts and telecommunications.2012,17(4):56-60