International Journal of Engineering &Technology, 7 (4) (2018) 6951-6955

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.22068
Research paper



Technique for querying over an encrypted database

Sahab Dheyaa Mohammed [1] *, Prof. Dr. Abdul Monem S. Rahma [2]

[1] University of Information Technology and Communications, Baghdad, Iraq
[2] University of Technology ,Department of Computer Science , Baghdad, Iraq
**Corresponding author E-mail: sahab7dia@yahoo.com*

Abstract

The increase in the amount of data in encrypted databases has caused problems in data processing and retrieval. In conventional method, difficulties in querying an encrypted database are experienced because it is time consuming and requires large computations when designing encryption algorithms or creating indexing tables that expose non-authorized disclosure. In this study, we proposed a new technique for querying encrypted databases. This technique allows legitimate users to immediately query without decrypting all the encryption records. In this method, index fields were created by using the hash function, and new approaches of encoding and hiding data information were used. Thus, the query can be used without encryptions. Hence, the index values will not be exposed to speculation by outside attackers when the index table has been detected. The proposed technique could provide high data security and consume lesser time when retrieving records compared with traditional methods.

Keywords: *Encryption Database; Encoding Data; Indexing Techniques; Index Field Generation; Query Processing.*

1. Introduction

Data are critical resources that should be securely stored for the efficient operation of a company. Companies typically store data in secure databases. However, this practice poses a challenge for administrators in creating a data protection strategy against intruders. Cryptography is a critical matter in database security. Unlike encryption, conventional security techniques cannot provide adequate data security. Data encryption introduces an important dimension in security and prevents users from obtaining data illegally and stealing database contents, which are saved in storage media, such as CD-ROM, tapes and disks [1].Organisation databases include sensitive data that can be unprotected from attacks and misuse [2]. Many techniques, such as encryption and other steganography methods, can solve this type of problem.
Steganography is a process that involves hiding a message in an appropriate carrier; for example, an image or an audio file [3].
Cryptography provides important database security. Unlike encryption techniques, conventional security techniques cannot provide sufficient data security. Data encryption introduces a significant dimension of security that prevents users from gaining illegal access and stealing data from the database when saved in storage mediums (e.g. CD-ROM, tapes and disks). Nevertheless, encryption safely assists in system execution because querying cannot be directly performed in the structural query language (SQL) of an encrypted database. SQL query can only work when encrypted data are decrypted. This entire process requires a certain amount of processing time. Although these mechanisms somehow restrict their applicability, several mechanisms have been suggested to solve this performance deterioration problem [4].

2. Related work

Providing security is an additional problem for databases. Thus, the encryption techniques of database management systems (DBMSs) can be used. However, despite the high security provided by encryption, issues, such as reduced system efficiency caused by encryption, still exist.
Reference [5] suggested a private database query protocol for seeking encrypted records by using an equality test algorithm on the encrypted databases. This suggestion aims to find and execute an effective form of search condition at each fully homomorphic encryption (FHE) cipher-text by using the algorithm of an equality test.
In [6], an indexing technique for searching the range queries was suggested. However, this technique is only helpful for numerical data and not for character data.
Reference [7] suggested a method for searching queries on the encryption database by using a homomorphic encryption technique based on the ideas of Gandhi's method. This method has two phases. In the first phase, homomorphic query can be used with a ring-based FHE. In the second phase, we use the homomorphic query to build a keyword search method in the smart grid.
Reference [8] suggested dividing the client's range of attributes into a set of intervals. The conformity between the interval and the original values is preserved on the client's side, whereas encrypted tables with interval information are stored in the database. Data are efficiently queried by mapping the original range and equivalent query values with the corresponding interval values.
Reference [9] proposed the creation of a B+-tree with the plaintext values, and then each B+-tree node was encrypted and stored at an unauthenticated DBMS. The main B+-tree was then performed at the unauthenticated DBMS as a table with two attributes, namely, a node ID, which is mechanically assigned by the system upon insertion, and the encrypted node content. This tech-



Copyright ©2018 Sahab Dheyaa Mohammed, Prof. Dr. Abdul Monem S. Rahma. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

nique has both advantages and disadvantages. An advantage is that the content of B+ -tree is invisible to an untrusted database service provider, whereas the disadvantage is that it involves considerable data processing on client machines.

3. Hash table

A hash function is a mathematical algorithm that converts a large set of data into smaller ones. The retrieved values are usually integers that act as a pointer to a set of data. These values are denoted as hash values. The difference between hash and compression values is that compression can be decompressed, and the data can be restored to its original size. However, the opposite is true for hash values. Hash functions are often used to develop a table or to search for items within a database and detect similar rows in a large table. These functions are also used in cryptography. A cryptographic hash function easily allows the user to verify a certain input data map onto a given hash value and confirm the integrity of transferring data. Moreover, such functions are also the building blocks for HMACs, which provide message authentication.
The hash function assigns a key or two to the hash value itself. In many applications, a collision problem exists amongst hash values. These problems should be reduced. Hash functions are primarily used in a hash table to rapidly determine where the record can be located. This function is used to transform the search key into an index that provides the location in the hash table, where the corresponding record should be stored. For this reason, each part of the hash table is often called a bucket, and hash values are called bucket indexes. Therefore, each bucket of a hash table is associated with a set of records instead of only one [10] [11].

4. Environment of the proposed work

This method proposes the construction of an SQL query to perform encryption database and retrieve records safely. The proposed technique consists of the proxy server of an enterprise, and the clients are represented as users. The proxy server is the centre of communication between the client and the remote database. Fig. 1 displays an overview of the query scheme of an encrypted database.

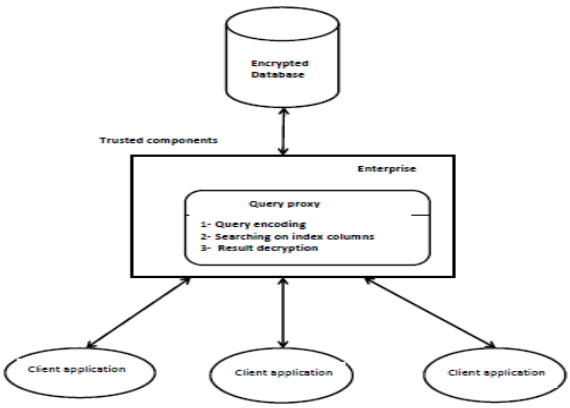


Fig. 1: Scheme of Querying Encrypted Database.

No direct communication occurs between the remote database and the clients. All the operations of the system (e.g. encoding, encryption, information hiding, decryption and building query) are only executed in the proxy server. The proxy server receives building queries from the clients and then forwards these queries to the database on a remote server. The proxy server creates the indexed fields in an encryption table by using hash functions and encoded values.
In other words, the proxy manages the communication between the database applications and the encrypted remote database. The clients represent the users who are authorised to view and query their information from the remote database via the proxy server. The remote database is the database server used to store encryption data. The database is isolated from any user, except the proxy server. The proposed system does not store any encryption data at the end of the system (users) because the encryption data is stored in the remote database server and only confined to provide a particular service to the users.

4.1. Encoded operation

This study suggested a technique for querying a remote encrypted database by using an index mechanism. Before applying the proposed technique, plaintext must be encoded in polynomial numbers of GF (2^8) by using an encoded dictionary method to provide an additional level of security when the proposed encryption approach is broken and decrease the embedding capacity when the random matrix of the proposed information hiding method is applied.
In the encoding operation, matching is performed every two encoded bytes with any of one word, one number or one symbol in the original database. The dictionary capacity contains up to 65,536 pairs of bytes, which are dynamically or manually stored. All data in the DB records are encoded in binary numbers as a form of polynomial numbers of GF (2^8). Each word, number (0……9) and symbol (#, $, /, () and *) is assigned to two bytes, and each byte represents one numeric value (0……255). The dictionary includes words, numbers and symbols that are dynamically added by allowing the accumulation of new words or deletion of old unused words. Furthermore, the words that previously entered manually are preserved. Table 1 presents an example of an encoding dictionary of data.

Table 1: Encoding Sensitive Data

No.	Sensitive words	Numerical cod		Binary cod	
		Byte 1	Byte 2	Byte 1	Byte 2
1	Abbas	2	60	00000010	00111100
2	Ahmed	200	60	11001000	00111100
3	Ahmad	150	2	10010110	00000010
4	Bassam	15	223	00001111	11011111
5	Hamid	55	78	00110111	01001110
6	Hazim	1	98	00000001	01100010
7	Hakim	28	251	00011100	11111011
8	Male	254	1	11111110	00000001
9	female	254	2	11111110	00000010
10	Unmarried	33	28	00100001	00011100
⋮	⋮	⋮	⋮		
448	/	44	67	00101100	001000011
500	7	170	165	10101010	10100101
501	9	185	196	10111001	11000100
⋮	⋮	⋮	⋮		
65536					

4.2. Information hiding operation

After the original data are encoded, such data are gathered as records in the temporary table, which has the same structure as the database. Then, the records will be encrypted by using the data encryption standard (DES) algorithm
The DES algorithm was elaborated by IBM in the early 1970s. This symmetric technique works on a 16-round Feistel cipher, which is a block of input length that encrypts 64 bits of plaintext bit-string, and the same key is used for decryption and encryption.
The encryption records are embedded into a random matrix (256 × 256) by using a linear equation. The elements of this matrix are polynomials in GF (2^8)
The procedure of embedding the encrypted record into a row of random matrix begins when encoded, encrypted records and random matrixes are available.
Embedding Equation (1) is used to merge an encrypted record of the database, in which one of the rows is randomly selected from the matrix. The results are replaced into cover file with the same location of the elements.

E(x) = (ax + b) mod m (1)

Where:
x: encrypted byte value of the record.
m: an irreducible polynomial of GF (2^8) , ($x^8 + x^4 + x^3 + x + 1$) .
a: element of cover-file (z) .
b: secret key.
E(x): embedded element for sensitive data in the cover
Each row includes one record, row ID field and the indexed fields. The rows are gathered in a temporary file until sending the rows and stored in the DB table Fig. 2 show embedded process in the row of random matrix.

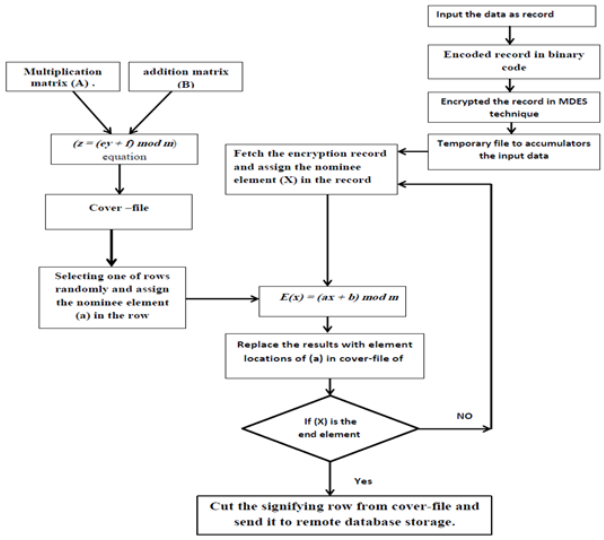


Fig. 2: Flow Chart Illustrating the Embedding Process.

5. Proposed querying method

Databases that use traditional encryption methods preserve data at the disk. The retrieval process involves decrypting the entire or a portion of the database to detect the data queried by a user. Thus, this process is time consuming.
Accordingly, we suggested a technique to overcome the drawbacks of traditional encryption methods in database systems. This technique processes SQL queries over encrypted data on a remote database without having to decrypt the data. Moreover, data decryption is only performed at the proxy site. This technique contains two phases. The first phase creates the index field of each record before sending it to the remote database.
The second phase builds the SQL query in the proxy server to be retrieved by the required records.

5.1. Construction of indexed fields

In this method, the index values are calculated and added as the pointer for each record before sending it to the database server. The index values are calculated on the basis of the combination of four element values via the XOR operation of the hash function (Equation 2). The first element represents the encoded value of the attribute (key), which is available in the encoding table. The three other elements correspond to the randomly selected values. Equation (2) is defined as follows:

$$y_i = x_i \oplus f_i \oplus f_i \oplus f_i$$

Where
y : The new value of index field.
x : The encoded value of the attribute (key).
f : is the element value that select randomly from the record.
i: is a polynomial number of GF (2^8) (0…….255)
To generate the indexing fields of the record, each word in the attribute (key) that is used in all the queries is added to the encoded table before encryption. The record elements represent the features that distinguish the record. Thus, the index fields are calculated by using the hash function to combine these randomly selected specific elements (columns). The results were then mapped into the index fields, where each value in the index is indicated to the specific record. The following example shows the calculation of the index fields.
Example
Compute index fields F257 and F258 of field F0 and F1 (Mohamed) in the 60001st row of Table 2 We assume that the encoded word ‘Mohammed’ in Table 3 of the database table is (102) and (57), and we select the values of files f4, f23, f66, f79, f213 and f244.

Table 2: Of Encryption Database Table

	Attribute (key)															Indexes				
	F0	F1	F3	F4	F5	-	F23	-	F66	-	F79	-	F213	F235	-	F244	F254	F255	F257	F258
0	63	123	213	4	62	-	7	-	107	-	126	-	240	10	-	115	13	26	91	39
1	211	45	100	89	11	-	207	-	141	-	183	-	105	15	-	176	27	83	89	177
2	56	12	150	231	177	-	16	-	140	-	225	-	104	14	-	61	33	128	249	96
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
60001	51	122	102	164	99	-	136	-	246	-	126	-	226	206	-	238	72	8	32	33
6002	123	134	160	164	48	-	123	-	214	-	183	-	254	244	-	225	218	12	68	253
6003	-	-	-	-	-	-		-	-	-		-	-	-	-		-	-	-	-
69678	34	129	188	190	81	-	297	-	109	-	259	-	169	19	-	235	94	13	55	75

Table 3: Temporary Database Table

	Name	father	Grand-f	surname	mother	Grand-m	gender	blood type	Issuer	Issuer Date	Effective Date	place of birth	date of Birth	Family number
1	Mohammed	Diaa	Ali	Bader	Noor	Hmeed	Male	O3	Adhamiya	2017/01/22	2027/01/21	Baghdad	1973/07/03	4578439
2	Huda	Ameer	Kreem	Alkaram	Hind	Salam	female	A+	Kadhimiya	2016/01/20	2027/01/21	Baghdad	1983/07/03	4578440

Sol.

From the equation (2) $y_i = x_i \oplus f_i \oplus f_i \oplus f_i$ the results of values of F257, F258 are :
F257= 102 ⊕ 164 ⊕246 ⊕226 = 32
F258= 57 ⊕ 136⊕ 126 ⊕ 238 = 33

F257 = 32
F258 = 33

5.2. Construction of SQL query

The query building inside the proxy server includes all elements of the record values that contribute to achieving the index (record indicators). The index values represent the array of record elements compressed by the hash function and encoding word of the attribute (key) available from the encoding table to increase safety and prevent attackers from speculating the index values.
Thus, when the SQL query is executed on the database table, it begins from the physical start of the table and continues with each record in the table. If a record matches the criterion, then such record will be included in the results. The structure of the encrypted DB is based on the fact that each byte in the record is located under one field. Thus, the pointers of each record may contain more than one field (byte) based on the number of bytes that represent the values of the attribute (key).
When an authorized user wants to query some records in the encrypted database table, he sends a query via the SQL server application to the proxy server, where the proxy server obtains the encoded word as a pair of bytes from the encoding table. The proxy server that builds the query sends it to the remote encrypted database, and then the searching mechanism begins. When the searching mechanism finds that the result matches the index field values, all the records that satisfy the user query are returned to the proxy server and decrypted before sending it to the user. The querying operation of the proposed algorithm is illustrated in the following example.
Example:

From the temporary database Table 2. Suppose a user need retrieves all records has the First Name word "Mohammad" in the encrypted database in Table 3.

SELECT #
FROM Encrypted Data Table
WHERE Name = Mohammed

The proposed algorithm interprets this query in the proxy server by performing an encoding mechanism where the word "Mohammed" is encoded as a pair of bytes (102), (57) and transforms it as follows:

SELECT #
FROM Encrypted Data Table
WHERE 102 ⊕f4⊕ f66 ⊕ f213 = f257 AND 57 ⊕f23 ⊕f79⊕f244 = f258

F257= 102 ⊕ 164 ⊕246 ⊕226 = 32

F258= 57 ⊕ 136 ⊕ 126 ⊕ 238 = 33

In this example, the user attempts to retrieve all records with the name 'Mohammed'. The proposed algorithm performs two steps. Firstly, in the encoding table, the binary value that corresponds to 'Mohammed' is searched. Secondly, a search is performed at randomly selected specific columns in the encrypted data table (f4, f66, f213, f23, f79 and 244) to compute the result of the index via the hash function. When the searching mechanism verifies that the result matches the values of the index field of the attribute name (F257, f258), all the records that correspond to the user's query are returned to the proxy server and decrypted before it is sent to the user.

5.2.1. Algorithmic outline

The proposed query algorithm is presented as follows:

1) The user presents a query to the proxy server.
2) The requested encoded word or number is fetched from the encoding table.
3) The query is built.
4) The query is sent to the encryption database table.
5) The SQL query is performed the searching operation on encrypted records.
   If (the search does not match all the values of the index columns)
   Go to step 9
   Else Go to step 6
6) The record has a matching value as the index, so it retrieved.
7) A proxy server decrypts all required records.
8) The requested data are sent to the client applications,
9) Go to step10.
10) "Search is unsuccessful".
11) Exit.

6. Results and testing

The proposed algorithm is evaluated on an information system database. The encryption database table includes 1,000,000 records for testing. In practice, the retrieval of these records is generally fast when the large database table is divided into smaller subtables. Moreover, the proposed method is suitable when a large amount of data is retrieved. Thus, the proposed algorithm is prompt when the table size is small, but takes a considerable amount of time when the database table is large. Figure 3 shows the time consumed by the proposed method, as compared with the conventional methods of different data samples.

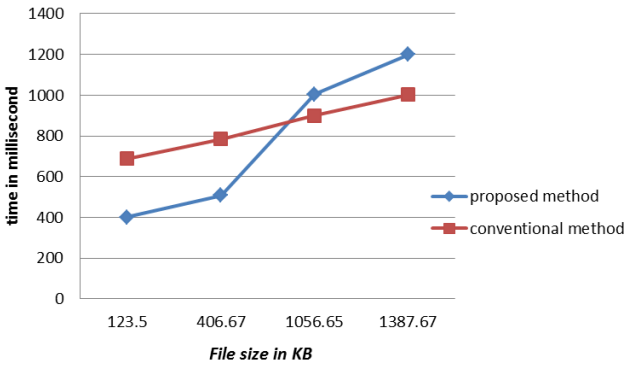


Fig. 3: Comparison Analysis of Querying Over an Encrypted Database.

This figure shows that our proposed technique consumes less time during record retrieval than conventional systems when the data size is small. Such problem can be addressed by partitioning the large tables into different sub-tables to decrease the time consumed in retrieving records.

7. Conclusion

This study proposed an effective algorithm for querying encrypted data. The data retrieval process is the main objective of this research and not the encryption process. Thus, a simple encryption operation was conducted to perform database encryption using the retrieval method from the encrypted database. In conventional systems, the query process from a large encrypted database takes a long time because the database needs to be decrypted entirely or partially before the results are sent to the client .
The proposed system improved the performance of the retrieval algorithm by decreasing the time consumed when sub-tables were used. The proposed system used the encoded query without needing an encryption query to match the encrypted data and only acquired the records requested by the user. The proposed system addressed the problem in conventional systems by retrieving the required encrypted query through indexing query and by returning the original record requests to the users.

References

[1] M. Sharma, A. Chaudhary, S. Kumar. "Query Processing Performance and Searching over Encrypted Data by using an Efficient Algorithm" International Journal of Computer Applications (0975 – 8887) Volume 62– No.10, January 2013. https://doi.org/10.5120/10114-4781.

[2] Kaur, Gurleen. *"A Review on Database Security."* International Journal of Engineering and Management Research (IJEMR) 7.3 (2017): 269-272.

[3] Morkel,Tayana, Jan HP Eloff, and Martin S. Olivier. *"An overview of image steganography."* Information Security South Africa Conference ISSA. 2005.

[4] Sharma, Manish, Atul Chaudhary, and Santosh Kumar. *"Query processing performance and searching over encrypted data by using an efficient algorithm."* arXiv preprint arXiv:1308.4687 (2013). https://doi.org/10.5120/10114-4781.

[5] J. H. Cheon, M. Kim, and M. Kim. Optimized search-and-compute circuits and their application to query evaluation on encrypted data. IEEE Trans. Information Forensics and Security, 11(1):188–199, 2016. https://doi.org/10.1109/TIFS.2015.2483486.

[6] J. Li and E.R. Omiecinski, "Efficiency and Security Trade-Off in Supporting Range Queries on Encrypted Databases," Technical Report, pp. 69-83, 2005. https://doi.org/10.1007/11535706_6.

[7] P. Sudharaka. "Homomorphic encryption and database query privacy" Diss. Memorial University of Newfoundland, 2016.

[8] H. Hacigümüs, B.R.I., C. Li and S. Mehrotra, "*Executing SQL over encrypted data in Database-Service-Provider Model*" ACM SIGMOD Madison, Wisconsin, USA, pp. 216-227, June 2002. https://doi.org/10.1145/564691.564717.

[9] E. Damiani, S. De Capitani Vimercati, Sushil Ja jodia, S.Paraboschi, and P.Samarati, "Balancing confidentiality and efficiency in untrusted relational dbmss", In Proceedings of CCS'03, pages 93{102, 2003.]. https://doi.org/10.1145/948109.948124.

[10] Wikipedia, the free encyclopedia that anyone can edit http://en.wikipedia.org/wiki/Hash_table , an article on Hash table

[11] Alhanjouri, Mohammed A., A. L. Derawi, and M. Ayman. "A New Method of Query over Encrypted Data in Database using Hash Map." A New Method of Query over Encrypted Data in Database using Hash Map 41.4 (2012). https://doi.org/10.5120/5533-7580.