

CIPHERTEXT DATABASE QUERY METHOD AND SYSTEM BASED ON SEARCHABLE ENCRYPTION

TECHNICAL FIELD

[0001] The present invention relates to the field of data processing, and in particular, relates to a ciphertext database query method and system based on searchable encryption.

BACKGROUND

[0002] With the in-depth application of the Internet and the advent of the era of big data, enterprises and even individuals have a large amount of data to be stored. At present, many cloud service providers provide outsourcing storage services with high cost performance for data storage, so enterprises or individuals usually buy these storage services to store their own data. In order to protect the security or privacy of these data, people tend to encrypt these data and then store them in the cloud server. However, the data in the form of ciphertext has brought great obstacles to the use of data. For example, in common data query scenarios, the data owner wants to query the related data (the data may be a document, audio, video, etc.) stored on the cloud server by a certain keyword, but the common encryption algorithm does not support querying on the data ciphertext. If the data to be queried is downloaded to the local area for decryption and then queried, the communication cost and decryption time are often unacceptable. In order to solve the query problem of ciphertext data, searchable encryption technology came into being. This technology encrypts plaintext data into ciphertext data with index structure by designing a special encryption algorithm. These ciphertext data are formed into a ciphertext database, in

which the index structure is searchable. During the query, a specific key is used to generate a query trapdoor T for the keyword of the query, and then T is used to search the ciphertext index to find out the data matching the keyword, and finally the query result is output.

[0003] Depending on whether the key used to generate the ciphertext index is the same as the key used to query the trapdoor, searchable encryption technologies are specifically divided into two categories: Symmetric Searchable Encryption (SSE) and Public Key Encryption with Keyword Search (PEKS).

[0004] However, the existing searchable encryption algorithms need to scan the full text of the database when searching, and the search time is linearly related to the size of the database, so the efficiency of the algorithm is low. In the data sharing scenario, the security and efficiency are not taken into account, and the ciphertext query request cannot be effectively controlled, so it is difficult to interfere with the decryption of ciphertext data and perceive high-value data resources, which affects the data sharing revenue of data providers.

SUMMARY

[0005] Based on the above description, an embodiment of the present application provides a ciphertext database query method and system based on searchable encryption, which solve the problems existing in the current searchable encryption algorithm for ciphertext database query.

[0006] In a first aspect, a ciphertext database query method based on searchable encryption is provided, the method comprises:

acquiring plaintext source data, the plaintext source data comprising data file names and data content; performing data encryption on the data content based on a

data encryption key to obtain ciphertext data content;

performing keyword extraction on the data content to obtain a keyword list, obtaining a ciphertext index library based on an index generation key according to the data file name, and storing the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly;

acquiring a target query keyword, and processing the target query keyword through a trapdoor generation key to generate a query ciphertext;

querying the ciphertext index library based on the query ciphertext to obtain a ciphertext query result and obtain the corresponding ciphertext data content;

performing data decryption on the ciphertext data content based on a data decryption key to obtain a plaintext data content.

[0007] The method further comprises the following operation before the operation of performing data decryption on the ciphertext data content based on a data decryption key to obtain a plaintext data content: generating a data decryption key of the plaintext source data in response to a decryption authorization request of the plaintext source data provider.

[0008] Preferably, the operation of performing data encryption on the data content based on a data encryption key to obtain ciphertext data content comprises:

performing encryption on the data content by a block encryption algorithm, wherein the block encryption algorithm at least comprises an AES algorithm and an SM4 algorithm.

[0009] Preferably, the data encryption key and the index generation key comprise:

generating a random value as the seed of key generation, and then using this seed and a key derivation function to generate the corresponding encryption key for each file.

[0010] Preferably, the operation of storing the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly further comprises:

establishing a data deletion table in the ciphertext database, wherein the data deletion table is used for recording deleted data, and when the query result contains deleted data, the decryption operation of the deleted data is skipped by accessing this table during decryption, so that the returned query result does not contain deleted data.

[0011] Preferably, the method comprises the following operation before the operation of processing the target query keyword through a trapdoor generation key to generate a query ciphertext:

obtaining the trapdoor generation key based on the trapdoor generation algorithm of token/nonce.

[0012] In a second aspect, a ciphertext database query system based on searchable encryption is provided, and the system comprises:

a data encryption module, being configured to acquire plaintext source data which comprises data file names and data content; perform data encryption on the data content based on a data encryption key to obtain ciphertext data content;

a ciphertext index library construction module, being configured to perform keyword extraction on the data content to obtain a keyword list, obtain a ciphertext index library based on an index generation key according to the data file name, and store the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly;

a ciphertext database storage module, being configured to store the ciphertext data content and the ciphertext index library;

a query trapdoor generation module, being configured to acquire a target query

keyword, and process the target query keyword through a trapdoor generation key to generate a query ciphertext;

a ciphertext index library query module, being configured to query the ciphertext index library based on the query ciphertext to obtain a ciphertext query result and obtain the corresponding ciphertext data content;

a data decryption module, being configured to perform data decryption on the ciphertext data content based on a data decryption key to obtain a plaintext data content.

[0013] Preferably, the system further comprises:

a key management and authorization module, being configured to generate a data decryption key of the plaintext source data in response to a decryption authorization request of the plaintext source data provider.

[0014] Preferably, the data encryption module comprises:

performing encryption on the data content by a block encryption algorithm, wherein the block encryption algorithm at least comprises an AES algorithm and an SM4 algorithm.

[0015] Preferably, the ciphertext database storage module further comprises:

establishing a data deletion table, wherein the data deletion table is used for recording deleted data, and when the query result contains deleted data, the decryption operation of the deleted data is skipped by accessing this table during decryption, so that the returned query result does not contain deleted data.

[0016] The beneficial effects generated by the technical solution provided by the embodiments of the present application at least comprise:

(1) an efficient and safe ciphertext query method and system are designed by adopting the high-intensity encryption method of one key for one data, technologies

based on symmetric cryptographic primitives and inverted index search or the like, and the method and system are especially suitable for the safe sharing of data and the circulation scene of data value.

(2) it supports ciphertext query under dynamic changes of data such as data addition or deletion, adapts to the updateability of data, and meets the needs of data changes in practical application scenarios.

(3) a token mechanism is used to ensure the immediacy and legality of the data query request, so as to prevent the adversary from launching illegal query or replay attack, thereby further improving the business security of data query.

(4) the data provider can count the frequently queried data according to the authorization information, so as to be capable of perceiving the data value and flexibly adjusting the data price. In the actual scenario, this is beneficial for data providers to obtain more reasonable revenue in data sharing.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] In order to explain the embodiments of the present invention or the technical solutions in the prior art more clearly, the following description will briefly introduce the drawings required in the description of the embodiments or the prior art. Obviously, the drawings in the following description are only exemplary, and other implementation drawings can be obtained by extension according to the provided drawings by those of ordinary skill in the art without any creative labor.

[0018] FIG. 1 is a flowchart diagram of a ciphertext database query method based on searchable encryption provided according to an embodiment of the present application.

[0019] FIG. 2 is a schematic view illustrating encryption of plaintext source data

provided according to an embodiment of the present application.

[0020] FIG. 3 is a schematic view of a ciphertext index database EDB provided according to an embodiment of the present application.

[0021] FIG. 4 is a schematic view of a query result in form of plaintext data provided according to an embodiment of the present application.

[0022] FIG. 5 is a block diagram of a ciphertext database query system based on searchable encryption provided according to an embodiment of the present application.

DETAILED DESCRIPTION

[0023] In order to make the purpose, technical solutions and advantages of the present application more clear, the present application will be further explained in detail below with reference to the attached drawings and embodiments. It shall be appreciated that, the specific embodiments described here are only used to explain the present application, and are not intended to limit the present application.

[0024] In the description of the present invention, the terms “including” and “having” and any variants thereof are intended to cover non-exclusive inclusion. For example, processes, methods, systems, products or equipments that comprise a series of steps or units are not necessarily limited to those steps or units that have been explicitly listed, but may also comprise other steps or units which are not explicitly listed but are inherent to these processes, methods, products or equipments, or steps or units added based on further optimization scheme of the inventive concept.

[0025] Aiming at a data sharing scenario with multiple data providers, the present application designs a ciphertext database query method and system architecture based on searchable encryption, and solves the following technical problems:

1. query efficiency: the ciphertext database query methods designed circumvent the public key encryption technology from the architecture design, all of which are designed with symmetric cryptographic primitives; the index structure of plaintext data and the generation of ciphertext data are completed by using system-level key sharing, and the inverted index technology is adopted in the search, which ensures the efficiency of data encryption and decryption and the search efficiency of ciphertext database.

2. query security: the algorithms adopt different keys to generate index structure and data ciphertext, and the indexes and data stored in the ciphertext database are all in form of ciphertext. In addition, when querying, keywords will also be encrypted and converted into query trapdoors at first, and the ciphertext database cannot recover keywords from the query trapdoors. This ensures the security of original data and query keywords.

3. query controllability: in the algorithm, query requests are effectively controlled by a query token or a random number (nonce), and the generated query trapdoors are different even for query requests with the same keyword. The replay query request can also be accurately detected.

4. decryption requires authorization, and the data value can be sorted: after the data inquirer obtains the query result in ciphertext form, he/she initiates a decryption request, the system requests the data provider for decryption authorization, and after the authorization is passed, the system sends the decrypted plaintext data to the data inquirer. Data providers can make statistics according to the authorization records of data decryption, and find out high-value data resources with high query frequency, so that they can adjust the pricing of data to improve the revenue of data sharing.

[0026] Specifically, please refer to FIG. 1, which shows a flowchart diagram of a

ciphertext database query method based on searchable encryption provided according to an embodiment of the present application, and the method may comprise the following steps:

S1: acquiring plaintext source data, and performing data encryption on the data content based on a data encryption key to obtain ciphertext data content.

[0027] The plaintext source data comprises data file names and data content. This step corresponds to arrows 1 and 2 in the figure. Step S1 involves the encryption algorithm PlainEnc of plaintext source data. Specifically:

the encryption algorithm of plaintext source data encrypts the plaintext source data provided by the data provider to generate corresponding ciphertext data.

[0028] Algorithm input: plaintext source data M , encryption key K ;

[0029] Algorithm output: ciphertext data $C = \text{Enc}(M, K)$, wherein Enc is a block encryption algorithm, such as AES algorithm and SM4 algorithm or the like. Optionally, the symmetric encryption and decryption algorithm of the present application is not limited to block cryptographic algorithms such as AES and SM4, but can be actually replaced by any other block cryptographic algorithm or stream cryptographic algorithm.

[0030] Detailed description of algorithm is as follows.

[0031] The case where the plaintext source data is a file is taken as an example, and it is assumed that there are total three data providers A, B, C, and file lists provided by them are respectively $\{A-01, A-02, A-03\}$, $\{B-01, B-02\}$, $\{C-01\}$. Encryption keys $\{K_{A01}, K_{A02}, K_{A03}\}$, $\{K_{B01}, K_{B02}\}$, $\{K_{C01}\}$ of plaintext data are generated respectively for these files. Contents of these files are encrypted respectively by adopting the block cryptographic algorithm Enc, and for example, the data content of plaintext A-01 is encrypted to get the ciphertext file $\text{Cipher_A01} = \text{Enc}(A01, K_{A01})$. Similarly, the

corresponding ciphertext files $\{\text{Cipher_A01}, \text{Cipher_A02}, \text{Cipher_A03}\}$, $\{\text{Cipher_B01}, \text{Cipher_B02}\}$ and $\{\text{Cipher_C01}\}$ are obtained after encrypting all the plaintext source data, as shown in FIG. 2.

[0032] During the specific implementation, in order to reduce the storage amount of plaintext data encryption keys, only one random value may be generated for each data provider as the seed of key generation, and then this seed and a key derivation function KDF are used to generate the corresponding encryption key for each file of each data provider. For example, a key seed seed_A is set for a user A, and KDF is used to generate three plaintext data encryption keys for the user A as follows:

$$K_{A0i} = \text{KDF}(\text{seed_A}, A-0i), \text{ wherein } i=1, 2, 3.$$

S2: performing keyword extraction on the data content to obtain a keyword list, obtaining a ciphertext index library based on an index generation key according to the data file name, and storing the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly.

[0033] This step corresponds to arrows 3 and 4 in the figure. Step S2 involves an algorithm BuildIndex for constructing the ciphertext index database, specifically:

the algorithm for constructing the ciphertext index database is used to process the keywords contained in the plaintext data of all data providers to generate ciphertext indexes, which together form the ciphertext index database, as shown in FIG. 3.

[0034] Algorithm input: all plaintext source data $\{M_1, M_2, \dots, M_n\}$, keyword list $\{W_{i1}, W_{i2}, \dots, W_{im}\}$ extracted from each source data M_i , that is, it is assumed that there are n pieces of source data in total, and the i th source data contains im keywords.

[0035] Algorithm output: ciphertext index generation key K , ciphertext index database EDB.

[0036] Detailed description of algorithm is as follows:

(a) let the keyword set of all data be W and the security parameter be λ , and a keyword ciphertext index with λ bit length is randomly selected to generate the key K^* ;

(b) for each keyword $w \in W$ in the keyword set, $DB(w)$ is used to indicate the set of the file names containing the keyword w , and two keys $K1$ and $K2$ are calculated by using the key K^* and a pseudo-random function $F: \{0,1\}^\lambda \times \{0,1\}^* \rightarrow \{0,1\}^\lambda$:

$$K1 = F(K^*, w || \text{key1}), K2 = F(K^*, w || \text{key2}),$$

wherein the symbol $||$ represents the cascading of w and the character string “key1” or “key2”.

[0037] First, the ciphertext index database EDB is initialized to be empty, and the following loop is executed:

- 1) initializing the counter $c = 0$;
- 2) calculating the tag $= F(K1, c)$, selecting $id \in DB(w)$, and calculating the ciphertext thereof $d = Enc(K2, id)$
- 3) incrementing the counter by $c++$;
- 4) adding the tag and ciphertext pair (l, d) to the ciphertext index database $(l, d) \rightarrow EDB$.

(c) outputting the ciphertext index generation key K^* and the ciphertext index database EDB.

S3: acquiring a target query keyword, and processing the target query keyword through a trapdoor generation key to generate a query ciphertext.

[0038] This step corresponds to arrows 5 and 6 in the figure, and this step comprises the query trapdoor generation algorithm TrapGen, specifically:

the query trapdoor generation algorithm uses the trapdoor generation key (which is the same as the ciphertext index generation key K^*) to generate query conditions (i.e. query trapdoors) of ciphertext for the query keywords w of data inquirers, and submits them to the ciphertext index database EDB for query.

[0039] In the present application, a trapdoor generation algorithm based on token/nonce is designed, so that the query trapdoor obtained by a data inquirer is different for each time, and the trapdoors obtained in two times of query are also different even for the same keyword w . For the convenience of marking, the present application uniformly uses token to represent query token or random number (nonce).

[0040] Algorithm input: querying keyword and token pair $(w, token)$;

[0041] Algorithm output: querying trapdoor T .

[0042] Algorithm description is as follows:

(a) calculating $K1 = F(K^*, w || key1)$, $H = F(K^*, token)$ by using index generation key K^* ;

(b) calculating $Kt = K1 \oplus H$, and the symbol $\oplus$ representing XOR operation;

(c) outputting the query trapdoor $T = (Kt, H)$.

S4: querying the ciphertext index library based on the query ciphertext to obtain a ciphertext query result and obtain the corresponding ciphertext data content.

[0043] The ciphertext index is only used to find out the corresponding ciphertext data, finally the ciphertext data is decrypted to obtain the plaintext data, and only the final plaintext data is required by the data inquirer, so in the present application, it is unnecessary to return the plaintext index to the data inquirer, and it is only required to

return the plaintext data content.

[0044] In this embodiment, this step corresponds to arrows 7, 8 and 9 in the figure. This step involves the search algorithm Search for the ciphertext index database, specifically:

the search algorithm for the ciphertext index database searches the ciphertext index database by using the query trapdoor to obtain the set of ciphertext file names of the corresponding tags.

[0045] Algorithm input: query trapdoor T , ciphertext index database EDB;

[0046] Algorithm Output: the set Cipher_ID of ciphertext file names;

[0047] Algorithm description is as follows:

(a) initializing calculator $c=0$, Cipher_ID=empty;

(b) calculating the key $K1=Kt \oplus H$ by using the query trapdoor T , and executing the following loop:

1) calculating the tag $l = F(K1, c)$, and querying the ciphertext index of the corresponding tag l in EDB, and the query result $d=find(EDB, l)$;

2) if the query result d of EDB in the previous step is not empty, adding d to the Cipher_ID, and incrementing the counter by $c++$; if the query result d is empty, exiting the loop.

(c) outputting the set Cipher_ID of ciphertext file names.

[0048] In an optional embodiment of the present application, the step S4 further comprises:

generating a data decryption key of the plaintext source data in response to a decryption authorization request of the plaintext source data provider. In this embodiment, this step corresponds to the arrow 10 in the figure.

S5: performing data decryption on the ciphertext data content based on the data

decryption key to obtain a plaintext data content.

[0049] In this embodiment, this step corresponds to arrows 11 and 12 in the figure. This step involves ciphertext index decryption algorithm DecryptIndex and ciphertext data decryption algorithm CipherDec, specifically:

ciphertext index decryption algorithm decrypts the set Cipher_ID of ciphertext file names to obtain the set Plain_ID of corresponding plaintext file names.

[0050] Algorithm input: the current query information $(w, token, H, Cipher_ID)$;

[0051] Algorithm output: set Plain_ID of plaintext file names.

[0052] Algorithm description is as follows:

(a) verifying whether the current query is valid or not by using the token and H in the query information, calculating $F(K^*, token)$, and if it is equal to H in the current query information and the token is different from the token values queried in the previous query history, then the query is valid; and otherwise, an invalid query is output, and the algorithm exits;

(b) initializing the Plain_ID to be empty, and the decryption key $K2 = F(K^*, w || key2)$ is recovered;

(c) for each $d \in Cipher_ID$, the following calculation is executed:

1) decrypting the plaintext file name $id = Dec(K2, d)$, wherein Dec represents the decryption algorithm corresponding to Enc.

2) adding id to the set Plain_ID of plaintext file names.

(d) outputting the set Plain_ID of plaintext file names.

[0053] The ciphertext data decryption algorithm decrypts ciphertext data content, firstly, the source of plaintext data is confirmed according to the file name in Plain_ID, and then the corresponding decryption key is acquired through the authorization of the

data provider to decrypt the plaintext file content, and the process is shown in FIG. 4.

[0054] Algorithm input: the set $Plain_ID$ of plaintext file names;

[0055] Algorithm output: the plaintext data set $PlainData$.

[0056] Algorithm description is as follows:

(a) initializing $PlainData$ to be empty, and for each $id \in Plain_ID$, the following operation is executed:

1) acquiring the decryption key DK corresponding to the id , for example, if the id is A-02, then the data comes from a provider A, and the decryption key of the ciphertext data content is $DK = KDF(seed_A, A-02)$.

2) decrypting to obtain plain data content $Plain = Dec(Cipher, DK)$, wherein $Cipher$ represents the ciphertext data content corresponding to the current id .

3) adding plaintext data $Plain$ to the set $PlainData$.

(b) outputting the plaintext data set $PlainData$.

[0057] As shown in FIG. 5, using the six algorithms given above, the present application designs a ciphertext database query system based on searchable encryption, and the system consists of seven modules, namely, a key management and authorization module, a data encryption module, a ciphertext index library construction module, a query trapdoor generation module, a ciphertext index library query module, a data decryption module and a ciphertext database storage module or the like.

[0058] (1) Key management and authorization module: responsible for generating and managing data encryption and decryption keys and ciphertext index generation keys, and providing ciphertext index generation keys and data decryption keys on the premise of obtaining authorization from data providers.

[0059] (2) Data encryption module: responsible for acquiring the data encryption

key from the key management and authorization module, and running PlainEnc algorithm to encrypt the plaintext data content to generate the ciphertext data content.

[0060] (3) Ciphertext index library construction module: responsible for acquiring ciphertext index generation key from the key management and authorization module, running BuildIndex algorithm to generate a ciphertext index library for all data names and keyword lists, and storing it in the ciphertext database storage module. In addition, it can be seen that the given ciphertext index database construction algorithm BuildIndex can cope with the dynamic changes (additions or deletions) of data, for the newly added data, it is only needed to add the data file name to the file name list DB(w) having the keyword w contained therein, and then calculate a new index pair (tag, file name ciphertext). It shall be noted that, the counter c in the algorithm at this point is the existing maximum plus 1. The simplest way to process data deletion is to add a data deletion table to record the deleted data, and if the query result contains the deleted data, the decryption operation of the deleted data can be skipped simply by accessing this table during decryption, so that the returned query result does not contain the deleted data.

[0061] (4) Query trapdoor generation module: responsible for receiving query keywords of data inquirers, acquiring trapdoor generation keys from key management and authorization module, and running TrapGen algorithm to generate corresponding query trapdoors.

[0062] (5) Ciphertext index library query module: responsible for receiving the query trapdoor from the query trapdoor generation module, and running Search algorithm to query from the ciphertext index library in the ciphertext storage module to obtain query results in ciphertext form.

[0063] (6) Data decryption module: responsible for receiving the ciphertext query

result from the ciphertext index library query module, acquiring the index generation key from the key management and authorization module, and running the DecryptIndex algorithm to decrypt to obtain the file name list of the plaintext. The file name list is parsed, a content decryption authorization request is initiated, CipherDec algorithm is executed to acquire data content decryption key from the key management and authorization module after the request is passed, and the plaintext data content is decrypted and returned to the data inquirer.

[0064] (7) Ciphertext database storage module: responsible for uniformly storing the ciphertext index library and the ciphertext data content for querying by the ciphertext index library query module and providing ciphertext data content for the data decryption module.

[0065] As shown in FIG. 1, it is the process flow for implementing the above system (indicated by the number on the flow arrow). The processing is started from the data and keyword list provided by the data provider, firstly, the ciphertext data content and ciphertext index library are generated and stored in the ciphertext storage module. Then the data inquirer initiates a query by using the query keyword, the query trapdoor generation module generates the query trapdoor, and the ciphertext index library query module uses the query trapdoor to perform ciphertext query to obtain the ciphertext result. The data decryption module decrypts the ciphertext query result, initiates authorization to the corresponding data provider, and decrypts the plaintext result and returns it to the inquirer.

[0066] (1) All the algorithms are designed with symmetric cryptographic primitives, and avoid the use of complex and time-consuming public key ciphers such as bilinear pairings or lattice ciphers, thus ensuring the high efficiency of algorithm execution.

[0067] (2) The ciphertext index library search algorithm adopts the inverted index

technology, that is, the keyword is used as the key value, and its corresponding data value is the file name id containing the keyword, which can improve the search efficiency. Because the forward index (file name, keyword) can only be searched according to the file name in search, the search time is linear complexity of the total number of files, while using the inverted index technology, the number of cycles in search is the number of files containing the keyword, and the search time is sublinear time complexity, which improves the search efficiency.

[0068] At the same time, different encryption keys (one key for one data) are adopted for different data files of different data providers, and the security of data content is highly protected.

[0069] What stored in the ciphertext storage module are all index libraries and data contents in the form of ciphertext, and even if the ciphertext storage module is breached, the adversary cannot decrypt it.

[0070] The query keywords provided by the data inquirer will be encrypted by the trapdoor generation module, and neither the ciphertext index library query module nor the ciphertext storage module can know the query keywords, thereby ensuring the security of keywords.

[0071] The ciphertext database query system based on searchable encryption provided according to the embodiment of the present application is used to realize the above-mentioned ciphertext database query method based on searchable encryption. Please refer to the definition of the ciphertext database query method based on searchable encryption described above for the specific definition of the ciphertext database query system based on searchable encryption, and this will not be further described herein. Each part of the ciphertext database query system based on searchable encryption described above may be realized in whole or in part by

software, hardware and combination thereof. The above modules may be embedded in or independent of the processor in the equipment in the form of hardware, or they may be stored in the memory in the equipment in the form of software so as to be called by the processor to execute the operations corresponding to the modules described above.

[0072] The technical features of the above-mentioned embodiments may be arbitrarily combined, and for briefness of the description, not all possible combinations of the technical features in the above-mentioned embodiments are described; however, the combinations of these technical features shall all be considered as within the scope recorded in this specification as long as there is no contradiction in the combinations.

[0073] The above-mentioned embodiments only represent several implementations of the present application, and they are described more specifically and in detail, but these embodiments should not thus be understood as limiting the scope of the present application for patent. It shall be noted that, several modifications and improvements can be made by those of ordinary skill in the art without departing from the concept of the present application, and all of these modifications and improvements are within the scope claimed in the present application. Therefore, the scope claimed in the present application for the patent shall be governed by the appended claims.

Claims

1. A ciphertext database query method based on searchable encryption, being characterized in that, the method comprising:

acquiring plaintext source data, the plaintext source data comprising data file names and data content; performing data encryption on the data content based on a data encryption key to obtain ciphertext data content;

performing keyword extraction on the data content to obtain a keyword list, obtaining a ciphertext index library based on an index generation key according to the data file name, and storing the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly;

acquiring a target query keyword, and processing the target query keyword through a trapdoor generation key to generate a query ciphertext;

querying the ciphertext index library based on the query ciphertext to obtain a ciphertext query result and obtain the corresponding ciphertext data content;

performing data decryption on the ciphertext data content based on a data decryption key to obtain a plaintext data content.

2. The method according to Claim 1, being characterized in that, the method further comprises the following operation before the operation of performing data decryption on the ciphertext data content based on a data decryption key to obtain a plaintext data content:

generating a data decryption key of the plaintext source data in response to a decryption authorization request of the plaintext source data provider.

3. The method according to Claim 1, being characterized in that, the operation of performing data encryption on the data content based on a data encryption key to

obtain ciphertext data content comprises:

performing encryption on the data content by a block encryption algorithm, wherein the block encryption algorithm at least comprises an AES algorithm and an SM4 algorithm.

4. The method according to Claim 1, being characterized in that, the data encryption key and the index generation key comprise:

generating a random value as the seed of key generation, and then using this seed and a key derivation function to generate the corresponding encryption key for each file.

5. The method according to Claim 1, being characterized in that, the operation of storing the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly further comprises:

establishing a data deletion table in the ciphertext database, wherein the data deletion table is used for recording deleted data, and when the query result contains deleted data, the decryption operation of the deleted data is skipped by accessing this table during decryption, so that the returned query result does not contain deleted data.

6. The method according to Claim 1, being characterized in that, the method comprises the following operation before the operation of processing the target query keyword through a trapdoor generation key to generate a query ciphertext:

obtaining the trapdoor generation key based on the trapdoor generation algorithm of token/nonce.

7. A ciphertext database query system based on searchable encryption, being characterized in that, the system comprising:

a data encryption module, being configured to acquire plaintext source data

which comprises data file names and data content; perform data encryption on the data content based on a data encryption key to obtain ciphertext data content;

a ciphertext index library construction module, being configured to perform keyword extraction on the data content to obtain a keyword list, obtain a ciphertext index library based on an index generation key according to the data file name, and store the ciphertext data content and the ciphertext index library into a ciphertext database correspondingly;

a ciphertext database storage module, being configured to store the ciphertext data content and the ciphertext index library;

a query trapdoor generation module, being configured to acquire a target query keyword, and process the target query keyword through a trapdoor generation key to generate a query ciphertext;

a ciphertext index library query module, being configured to query the ciphertext index library based on the query ciphertext to obtain a ciphertext query result and obtain the corresponding ciphertext data content;

a data decryption module, being configured to perform data decryption on the ciphertext data content based on the data decryption key to obtain a plaintext data content.

8. The system according to Claim 7, being characterized in that, the system further comprises:

a key management and authorization module, being configured to generate a data decryption key of the plaintext source data in response to a decryption authorization request of the plaintext source data provider.

9. The system according to Claim 7, being characterized in that, the data encryption module comprises:

performing encryption on the data content by a block encryption algorithm, wherein the block encryption algorithm at least comprises an AES algorithm and an SM4 algorithm

10. The system according to Claim 7, being characterized in that, the ciphertext database storage module further comprises:

establishing a data deletion table, wherein the data deletion table is used for recording deleted data, and when the query result contains deleted data, the decryption operation of the deleted data is skipped by accessing this table during decryption, so that the returned query result does not contain deleted data.