

Searchable Encryption with Hybrid Index

Field of the Invention

This disclosure relates to a method and system for searching documents in a server. Particularly, this disclosure relates to a method and system for searching documents in a server having a database comprising separate indexes.

Summary of the Prior Art

We consider the problem of searching for a set of documents matching a given keyword. Without any pre-processing, a document can be viewed as a list of keywords which match the document. Collecting all documents form a dictionary called the forward index, which maps documents to their respective lists of keywords. To search for a keyword in this presentation of documents, one needs to check every single list of keywords to see whether the keyword being searched for is in the list. The search complexity is thus linear in the number of documents. One method of speeding up the search process is by an offline pre-processing stage, which creates another dictionary called the invert-index. Such index maps each keyword to a list of documents which contain the keyword. To search for a keyword, one only needs to check the inverted index on the given keyword, and outputs the list being mapped to. In this case, the search complexity is sublinear, i.e., linear in the number of matching documents only, which is optimal.

We consider the same keyword search problem in a setting with encrypted documents. Symmetric-key (resp. public-key) encryption schemes allow the secret (resp. public) encryption key holder to convert human or machine readable documents into seemingly random ciphertexts, which only the decryption key holder can comprehend. Ciphertexts created by ordinary encryption schemes loss most utility of the documents being encrypted. In particular, one cannot search for ciphertexts matching a keyword. Searchable encryption (SE) schemes are extensions of encryption schemes which allow searching over ciphertexts without decrypting all of them. Concretely, in SE, documents are encrypted with respect to keywords. These documents are typically stored by a cloud server in its database. To search for a keyword, the decryption key holder (reader) engages in a search protocol with the server. Upon termination, the client obtains those documents which match the keyword being searched for, while the server learns minimal information leaked about the documents and the keyword.

Dynamic SE schemes are SE schemes where it is not necessary to encrypt all documents when setting up the database. Instead, updates such as adding and removing documents can be made to the database dynamically. To update the database, the encryption key holder (writer) engages in an update protocol with the server. Upon termination, the server obtains an updated database which may contain new encrypted documents or have some existing documents removed. The server may not know what exactly is added to or removed from the database.

Prior SE schemes can be roughly classified into forward-index-based schemes and inverted-index-based schemes. Early symmetric-key SE (SKSE) and most public-key SE (PKSE) are forward-index-based. These schemes inherit the linear search complexity of searching over a forward index, but supports easy document additions by simply appending the forward index. Inverted-index-based SKSE schemes are designed to achieve sublinear search complexity. Unfortunately, inverted-index-based SKSE schemes tend to require complicated update procedures, which are generally less efficient and difficult to implement. The literature mostly focuses on designing SE schemes which are by themselves more efficient and more secure, without paying much attention to how SE schemes can be applied in practice most effectively.

Hence, those skilled in the art are striving to storing documents in a database that allow a more efficient and cost effective method of searching for documents.

Summary of the Invention

The above and other problems are solved and an advance in the state of art is made by systems and methods provided by embodiments in accordance with the disclosure. A first advantage of embodiments of systems and methods in accordance with the disclosure is that the systems and methods is more cost effective. This is because data stored in the plaintext database can be stored with high locality. Once a matching keyword-document pair is revealed by the relatively costly search protocol of the dynamic SE scheme, the pair is moved to the plaintext database, so that subsequent access to this pair is made more efficient and less costly. A second advantage of embodiments of systems and methods in accordance with the invention is that systems and methods are compatible with any dynamic SE scheme. Hence, the system and method can be implemented to existing systems employing dynamic SE schemes directly

The above advantages are provided by embodiments of a system and method for a secret encryption scheme having a first database and a second database. The first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data.

A first aspect of the disclosure describes an indexing method for a secret encryption scheme having a first database and a second database. The first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data. The method comprises: receiving a document containing a keyword; determining a type of keyword in the document; in response to the keyword being a search token, searching the keywords in the first database that matches the keyword in the document and retrieving data paired with a keyword in the first database that matches the keyword in the document; recording a new entry indicator data associated to the keyword in the first database that matches the keyword in the document as A; searching the labels in the second database, for new added entry indicator associated to the labels greater than A, with the keyword in the document and retrieving encrypted data associated to label in the second database that matches the keyword in the document; returning a search result containing the data and encrypted data retrieved from the first and second databases; and updating the first and the second databases.

In accordance with an embodiment of the first aspect, the step of updating the first and the second databases comprises: appending the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database; removing the label in the data-label pair this is revealed after the data is retrieved in the second database; and setting a new added entry indicator associated to the new keyword data pair to current value.

In accordance with an embodiment of the first aspect, the new added entry indicator is a counter and the current value is identical to the value associated to the data-label pair this is revealed after the data is retrieved in the second database. In accordance with another embodiment of the first aspect, the new added entry indicator is a timestamp and the current value is the current time.

In accordance with an embodiment of the first aspect, the indexing method further comprises: in response to the keyword being an encrypted data, adding a new data-label pair in the second database and setting a new entry added entry indicator associated to the new data-label pair with a current value. In accordance with an embodiment of this embodiment, the new added entry indicator is a counter and the current value is an

increment of the counter. In accordance with another embodiment of this embodiment, the new added entry indicator is a timestamp and the current value is the current time.

A second aspect of the disclosure describes an indexing system for a secret encryption scheme having a first database and a second database. The first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data. The system comprises: a processor, a non-transitory memory and instructions stored on the memory executable by the processor to: receive a document containing a keyword; determine a type of keyword in the document; in response to the keyword being a search token, search the keywords in the first database that matches the keyword in the document and retrieve data paired with a keyword in the first database that matches the keyword in the document; record a new entry indicator data associated to the keyword in the first database that matches the keyword in the document as A; search the labels in the second database, for new added entry indicator associated to the labels greater than A, with the keyword in the document and retrieving encrypted data associated to label in the second database that matches the keyword in the document; return a search result containing the data and encrypted data retrieved from the first and second databases; and update the first and the second databases.

In accordance with an embodiment of the second aspect, the instruction to update the first and the second databases comprises instructions to: append the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database; remove the label in the data-label pair this is revealed after the data is retrieved in the second database; and set a new added entry indicator associated to the new keyword data pair to current value.

In accordance with an embodiment of the second aspect, the new added entry indicator is a counter and the current value is identical to the value associated to the data-label pair this is revealed after the data is retrieved in the second database. In accordance with another embodiment of the second aspect, the new added entry indicator is a timestamp and the current value is the current time.

In accordance with an embodiment of the second aspect, the indexing system further comprises instructions to: in response to the keyword being an encrypted data, add a new data-label pair in the second database and set a new entry added entry indicator associated to the new data-label pair with a current value. In accordance with an embodiment of this embodiment, the new added entry indicator is a counter and the current value is an

increment of the counter. In accordance with another embodiment of this embodiment, the new added entry indicator is a timestamp and the current value is the current time.

A third aspect of the disclosure describes an indexing method for a secret encryption scheme having a first database and a second database. The first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data. The method comprises: receiving a document containing a search token computed by a keyword; searching the keywords in the first database that matches the keyword in the document and retrieving data paired with a keyword in the first database that matches the keyword in the document; in response to no data being retrieved from the first database, searching the labels in the second database with the keyword in the document, retrieving encrypted data associated to label in the second database that matches the keyword in the document and updating the first and the second databases; and returning a search result containing the data or encrypted data retrieved from the first and second databases.

In accordance with an embodiment of the third aspect, the step of updating the first and the second databases comprises: appending the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database; and removing the label in the data-label pair this is revealed after the data is retrieved in the second database.

A fourth aspect of the disclosure describes an indexing system for a secret encryption scheme having a first database and a second database. The first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data. The system comprises: a processor, a non-transitory memory and instructions stored on the memory executable by the processor to: receive a document containing a search token computed by a keyword; search the keywords in the first database that matches the keyword in the document and retrieve data paired with a keyword in the first database that matches the keyword in the document; in response to no data being retrieved from the first database, search the labels in the second database with the keyword in the document, retrieve encrypted data associated to label in the second database that matches the keyword in the document and update the first and the second databases; and return a search result containing the data or encrypted data retrieved from the first and second databases.

In accordance with an embodiment of the fourth aspect, the instruction to update the first and the second databases comprises instructions to: append the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database;

and remove the label in the data-label pair this is revealed after the data is retrieved in the second database.

Brief Description of the Drawings

The above advantages and features in accordance with this invention are described in the following detailed description and are shown in the following drawings:

Figure 1 illustrating an overview of a system for applying SE technology based on the new indexing approach in accordance with the disclosure;

Figure 2 illustrating a processing system in a server that executes the instructions to perform the processes for providing a method and/or system in accordance with the disclosure;

Figure 3 illustrating a flow diagram of a process of searching and updating a first database and a second database upon receiving a document in accordance with an embodiment of this disclosure;

Figure 4 illustrating a flow diagram of another process of searching and updating a first database and a second database upon receiving a document in accordance with an embodiment of this disclosure;

Figure 5 illustrating an example of the databases where the first database is empty;

Figure 6 illustrating an example of the databases after a search is performed by the server in addition to the example shown in figure 5;

Figure 7 illustrating an example of the databases after a new document is added in addition to the example shown in figure 6; and

Figure 8 illustrating an example of the databases after a search is performed by the server in addition to the example shown in figure 7.

Detailed Description

This disclosure relates to a method and system for searching documents in a server. Particularly, this disclosure relates to a method and system for searching documents in a server having a database comprising separate indexes.

In current SE schemes, the ciphertexts of the documents that are revealed in a search are kept in the ciphertext database. Thus, in a subsequent search for the same

keyword, the server repeats most of its previous computation in the previous search, assuming not many modifications has been made to the documents matching the keyword. This repetition of computation is unnecessary in terms of security, since the documents which are already revealed to the server in a previous search are no longer protected by the SE scheme. Keep storing these documents in SE-encrypted form is a waste of resource.

Further, searching the ciphertext database, even of an inverted-index-based SE scheme, is costly compared to searching a plaintext invert-index. This is due to the inherently low locality of memory access of SE schemes. The physical data of the SE encrypted documents matching a keyword are typically stored sparsely in the memory for security. In contrast, the physical data of each list of documents in a plaintext inverted-index are stored in close proximity in the memory.

This invention proposes a hybrid indexing approach which is built upon existing SE schemes. It achieves the balance of efficient searching and efficient addition/deletion of data, without changing the underlying SE schemes.

The system and method in accordance with embodiments of this disclosure is based on an elegant process of applying a new indexing approach which is based on current SE technology. The new indexing approach is able to achieve the balance of efficient searching and efficient addition and/or deletion of data without changing the underlying SE schemes.

Figure 1 illustrates an overview of the system 100 of applying SE technology based on the new indexing approach. The system 100 comprises a server 110 having a first database 120 and a second database 130. The first database 120 consists of a data structure for inverted indexes, which allows sublinear keyword search. This data structure stores keyword-document pairs which are revealed in previous searches in plaintext. Essentially, the data structure of the first database includes a list of keywords and a list of data where data associated to a keyword in the list are paired together. In short, the first database comprises keyword-data pairs. The second database 130 is the database of a dynamic SE scheme. Particularly, the data structure of the second database includes a list of encrypted data and a list of label labels computed by keywords where each encrypted data is paired with labels computed by keywords that are relevant to the encrypted data. In short, the second database comprises data-label pairs. All data, the set of (data, keyword/label) pairs, will be stored in either the first database or the second database. Examples of the data structure of the first and second databases are shown in figures 5-8.

As shown in figure 1, the system 100 starts with receiving a document 150, where each document is associated with a set of keywords. Upon receiving the document 150, the server 110 searches the first database 120 for the given keyword, to retrieve previously revealed matching documents. Thereafter, the server 110 engages in a search protocol with the user, to search for the keyword in the second database 130. This reveals matching documents which are added after the previous search for the keyword. The server 110 then merges the search results revealed via the first and second database searches and returns the search result 140 to the user. Subsequently, the server engages in an update protocol to delete keyword-document pairs which are revealed in the second database search from the underlying dynamic SE scheme and adds the keyword-document pairs to the first database.

The programming languages that may be used for writing the programme in accordance with embodiments of this disclosure include Scala, Java and Haskell. Nevertheless, one skilled in the art will recognise that other programming languages may be used without departing from the invention. The processes in accordance with embodiments of this disclosure are executable on the various platforms, namely, Windows, Linux and Mac. Nevertheless, one skilled in the art will recognise that the processes in accordance with embodiments of this disclosure may also be executable on other platforms without departing from this disclosure.

Processes stored as instructions in a media that are executed by a processing system or a virtual machine running on processing system in the system provide the method and/or system in accordance with this invention. The instructions may be stored as firmware, hardware, or software. Figure 2 illustrates a processing system 200 in the server 110 that executes the instructions to perform the processes for providing a method and/or system in accordance with this disclosure. One skilled in the art will recognize that the exact configuration of each processing system may be different and the exact configuration of the processing system in each device may vary. Thus, processing system 200 shown in Figure 2 is given by way of example only.

Processing system 200 includes Central Processing Unit (CPU) 205. CPU 205 is a processor, microprocessor, or any combination of processors and microprocessors that execute instructions to perform the processes in accordance with the present disclosure. CPU 205 connects to memory bus 210 and Input/ Output (I/O) bus 215. Memory bus 210 connects CPU 205 to memories 220 and 225 to transmit data and instructions between the memories and CPU 205. I/O bus 215 connects CPU 205 to peripheral devices to transmit data between CPU 205 and the peripheral devices. One skilled in the art will recognize that

I/O bus 215 and memory bus 210 may be combined into one bus or subdivided into many other busses and the exact configuration is left to those skilled in the art.

A non-volatile memory 220, such as a Read Only Memory (ROM), is connected to memory bus 210. Non-volatile memory 220 stores instructions and data needed to operate various sub-systems of processing system 200 and to boot the system at start-up. One skilled in the art will recognize that any number of types of memory may be used to perform this function.

A volatile memory 225, such as Random Access Memory (RAM), is also connected to memory bus 210. Volatile memory 225 stores the instructions and data needed by CPU 205 to perform software instructions for processes such as the processes required for providing a system in accordance with this invention. One skilled in the art will recognize that any number of types of memory may be used as volatile memory and the exact type used is left as a design choice to those skilled in the art.

I/O device 230, keyboard 235, display 240, memory 245, network device 250 and any number of other peripheral devices connect to I/O bus 215 to exchange data with CPU 205 for use in applications being executed by CPU 205. I/O device 230 is any device that transmits and/or receives data from CPU 205. Keyboard 235 is a specific type of I/O that receives user input and transmits the input to CPU 205. Display 240 receives display data from CPU 205 and display images on a screen for a user to see. Memory 245 is a device that transmits and receives data to and from CPU 205 for storing data to a media. Network device 250 connects CPU 205 to a network for transmission of data to and from other systems.

A system and method for applying SE technology based on the new indexing approach in accordance with embodiments of the disclosure will now be described.

Figure 3 illustrates a flow diagram of process 300 performed by the processor in processing system 200 in accordance with an embodiment of this disclosure. Process 300 is a process for searching and updating the first and second databases upon receiving a document. Process 300 begins with step 305 by receiving a document containing search tokens, T, computed by keywords. The search token may be via a trapdoor method as shown in figure 5 below.

In step 310, process 300 searches the first database for the given keywords, to retrieve previously revealed matching documents. This stage can be completed in sublinear

time due to the inverted-index data structure. Matched documents are retrieved and stored in the search result.

In step 312, process 300 determines whether a matched document is available. If a matched document is not available, process 300 proceeds to step 315. If a matched document is available, process 300 proceeds to step 320 and return the search result.

In step 315, process 300 engages in the search protocol with the user, to search for the keyword in the second database. An example of the search protocol is that the second database stores the “encrypted data/document ID” together with “labels computed by keywords” as shown in figures 5-8. When the user searches for a keyword, the user input is a “search tokens computed by keywords”. Then the server compares the “search tokens” with all the “labels” one by one. If the comparison returns “true” (it can be a simple identical match, or some complicated computation), the data/document ID is returned. The server computation cost for this stage depends on the search complexity of the underlying dynamic SE scheme. Matched documents are retrieved and appended to the search result.

In step 320, process 300 returns the search result to the user.

In step 325, process 300 updates the first and second databases. Particularly, process 300 appends the first database with the keyword-document pairs which are revealed in step 315 and remove the keyword-document pairs which are revealed in step 315 from the second database. Step 325 may be performed concurrently with step 320. Alternatively, step 325 may be performed after step 315 and prior to step 320.

Process 300 ends after step 320.

From time to time, new entries would be added to the database. These new entries would be added to the second database since the keywords are not revealed. Essentially, a new entry would include an “encrypted data/document ID” (as shown as Enc in figures 5 to 8) together with “labels computed by keywords” (as shown as F_k in figures 5-8). Figure 4 illustrates a flow diagram of process 300a performed by the processor in processing system 200 in accordance with an embodiment of this disclosure. Process 300a is a process for searching and updating the first and second databases upon receiving a document where the document contains keyword or keywords. The keyword may be provided as search tokens or encrypted data. Process 300a is a modification of process 300 where the second database is appended to include new entries. Same reference numerals are used for similar steps in processes 300 and 300a. Process 300a begins with step 405 by receiving a

document. If the document contains encrypted data, process 300a proceeds to step 430. If the document contains search tokens, process 300a proceeds to step 310.

In step 430, process 400 appends the second database to add a new entry containing the “encrypted data” (as shown as Enc in figures 5 to 8) together with “labels computed by keywords (as shown as F_K in figures 5-8). A new added entry indicator is marked against the new entry. The new added entry indicator may be a counter or time stamp. If the counter is used, the new added entry indicator would be counter+1 (i.e. an increment of counter). If timestamp is used, the current time would be used. One skilled in the art will recognise that counter would be initialised to zero at the beginning of process 300a should counter be used as the new added entry indicator.

In step 310, process 300a searches the first database for the given keywords, to retrieve previously revealed matching documents. This stage can be completed in sublinear time due to the inverted-index data structure. Matched documents are retrieved and stored in the search result. The new added entry indicator is also noted. If the new added entry indicator is available, the new added entry indicator is recorded as ‘A’.

In step 315, only the entries where the new added entry indicator is greater than A are considered. Particularly, process 300a only searches the same keywords with counter or timestamp that is larger than ‘A’. If all entries are equal or less than ‘A’, process 300a engages in the search protocol with the user, to search for the keyword in the second database. As mentioned above, one example of the search protocol is that the second database stores the “encrypted data/document ID” together with “labels computed by keywords” as shown in figures 5-8. When the user searches for a keyword, the user input is a “search tokens computed by keywords”. Then the server compares the “search tokens” with all the “labels” one by one. If the comparison returns “true” (it can be a simple identical match, or some complicated computation), the data/document ID is returned. Matched documents are retrieved and appended to the search result. The server computation cost for this stage depends on the search complexity of the underlying dynamic SE scheme. Further, as only the entries greater than ‘A’ are searched, step 312 of process 300 would not be required in process 300a.

In step 320, process 300a returns the search result to the user.

In step 325, process 300a updates the first and second databases. Particularly, process 300a appends the first database with the keyword-data pairs which are revealed in step 315 and removes the label(s) in the data-label pairs which are revealed in step 315

from the second database. Further, the updated rows in the first database will have its new added entry indicator such as counter or timestamp set as the current value. If counter is used, the new added entry indicator would be the same value as the counter of the data-label pair which was revealed in step 315 from the second database (i.e. current counter value). If timestamp is used, the current time would be used. Step 325 may be performed concurrently with step 320. Alternatively, step 325 may be performed after step 315 and prior to step 320.

Process 300a ends after step 430, 320 or 325.

Figures 5 to 8 illustrate examples of updating of the first and second databases in response to receiving a document where the document may be a request for a search or a request to add document to the second database.

Figure 5 illustrates an example where the first database is empty which is applicable at the initial stage. Particularly, the initialised database 510 contains a first database 591 which is empty and the second database 592 contains 3 documents with respective encrypted keywords. Further, counter is initialised to zero. In this example, if the document received contains a search token, T, which contains the keyword “tomato” (as illustrated with $T = \text{Trapdoor}_K(\text{tomato})$), process 300 proceeds in the following sequence, 305 → 310 → 312 → 315 → 320 and 325, while process 300a proceeds in the following sequence 405 → 310 → 315 → 320 and 325. The resultant database 520 after process 300 and process 300a would cause the first database 591 to include a new keyword, “tomato”, with the respective document ID while the keyword, “tomato” in the second database 592 would be removed.

Figure 6 illustrates a follow up search after the example shown in figure 5. Particularly, a user request for a search for a keyword “onion” which is subsequent to the request for search for keyword, “tomato”. In this circumstance, process 300 proceeds in the following sequence, 305 → 310 → 312 → 315 → 320 and 325 while process 300a proceeds in the following sequence 405 → 310 → 315 → 320 and 325. The resultant database 530 after process 300 and process 300a would cause the first database 591 to include a new keyword, “onion”, with the respective document ID while the keyword, “onion” in the second database 592 would be removed.

Figure 7 illustrates adding a new document after the example shown in figure 6. In this circumstance, process 300a proceeds in the following sequence, 405 → 310 → 315 → 320 and 325. The resultant database 540 after process 300a would cause the second

database 592 to include a new entry, recipe ID₄, with respective keywords and a counter and/or a timestamp.

Figure 8 illustrates a search for keyword, “tomato”, after the example shown in figure 7. In this circumstance, process 300a proceeds in the following sequence, 405 → 310 → 315 → 320 → 325. The resultant database 550 after process 400 would cause the first database 591 to update keyword, “tomato” to include the new document ID while the keyword, “tomato” in the second database would be removed. The updated rows in the first database 591 will have its counter or timestamp set as the current value. In this instance, the counter is set to 1 which is identical to the counter for recipe ID₄ in the second database 592.

While the first database as illustrated above are in plaintext, one skilled in the art will recognise that the first database can be further encrypted by the server, using its own secret key. This is used to provide data protection for data-at-rest.

The main application scenario is that the databases are stored in a cloud server. The data owner wants to store some data in the cloud database for some data users to retrieve them. However, they do not want the cloud server to learn about the content, as well as the keyword used to search the data.

Benefits

Data stored in the plaintext database can be stored with high locality. Once a matching keyword-document pair is revealed by the relatively costly search protocol of the dynamic SE scheme, the pair is moved to the plaintext database, so that subsequent access to this pair is made more efficient. Advantageously, this new approach is compatible with any dynamic SE scheme. Furthermore, this new approach does not introduce new application interface. That is, the new approach can be implemented to existing systems employing dynamic SE schemes directly.

The above is a description of embodiments of a method and system of searching and updating two databases upon receiving a document where one of the two databases is based on existing SE schemes. It is foreseeable that those skilled in the art can and will design alternative method and system based on this disclosure that infringe upon this invention as set forth in the following claims.

CLAIMS:

1. An indexing method for a secret encryption scheme having a first database and a second database, the first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data, the method comprising:
 - receiving a document containing a keyword;
 - determining a type of keyword in the document;
 - in response to the keyword being a search token,
 - searching the keywords in the first database that matches the keyword in the document and retrieving data paired with a keyword in the first database that matches the keyword in the document;
 - recording a new entry indicator data associated to the keyword in the first database that matches the keyword in the document as A;
 - searching the labels in the second database, for new added entry indicator associated to the labels greater than A, with the keyword in the document and retrieving encrypted data associated to label in the second database that matches the keyword in the document;
 - returning a search result containing the data and encrypted data retrieved from the first and second databases; and
 - updating the first and the second databases.
2. The indexing method according to claim 1 wherein the step of updating the first and the second databases comprises:
 - appending the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database;
 - removing the label in the data-label pair this is revealed after the data is retrieved in the second database; and
 - setting a new added entry indicator associated to the new keyword data pair to current value.
3. The indexing method according to claim 2 wherein the new added entry indicator is a counter and the current value is identical to the value associated to the data-label pair this is revealed after the data is retrieved in the second database.
4. The indexing method according to claim 2 wherein the new added entry indicator is a timestamp and the current value is the current time.

5. The indexing method according to claim 1 further comprising:
in response to the keyword being an encrypted data, adding a new data-label pair in the second database and setting a new entry added entry indicator associated to the new data-label pair with a current value.
6. The indexing method according to claim 5 wherein the new added entry indicator is a counter and the current value is an increment of the counter.
7. The indexing method according to claim 5 wherein the new added entry indicator is a timestamp and the current value is the current time.
8. An indexing system for a secret encryption scheme having a first database and a second database, the first database comprises a list of keyword-data pairs and the second database comprises encrypted data-label pairs where the labels computed by keywords are associated to the encrypted data, the system comprising:
a processor, a non-transitory memory and instructions stored on the memory executable by the processor to:
receive a document containing a keyword;
determine a type of keyword in the document;
in response to the keyword being a search token,
search the keywords in the first database that matches the keyword in the document and retrieve data paired with a keyword in the first database that matches the keyword in the document;
record a new entry indicator data associated to the keyword in the first database that matches the keyword in the document as A;
search the labels in the second database, for new added entry indicator associated to the labels greater than A, with the keyword in the document and retrieving encrypted data associated to label in the second database that matches the keyword in the document;
return a search result containing the data and encrypted data retrieved from the first and second databases; and
update the first and the second databases.
9. The indexing system according to claim 8 wherein the instruction to update the first and the second databases comprises instructions to:
append the first database with a new keyword-data pair that is revealed after the data is retrieved in the second database;

remove the label in the data-label pair this is revealed after the data is retrieved in the second database; and

set a new added entry indicator associated to the new keyword data pair to current value.

10. The indexing system according to claim 9 wherein the new added entry indicator is a counter and the current value is identical to the value associated to the data-label pair this is revealed after the data is retrieved in the second database.

11. The indexing system according to claim 9 wherein the new added entry indicator is a timestamp and the current value is the current time.

12. The indexing system according to claim 8 further comprising instructions to:
in response to the keyword being an encrypted data, add a new data-label pair in the second database and set a new entry added entry indicator associated to the new data-label pair with a current value.

13. The indexing system according to claim 12 wherein the new added entry indicator is a counter and the current value is an increment of the counter.

14. The indexing system according to claim 12 wherein the new added entry indicator is a timestamp and the current value is the current time.