SYSTEM AND METHOD FOR QUERYING AN ENCRYPTED DATABASE FOR DOCUMENTS SATISFYING AN EXPRESSIVE KEYWORD ACCESS STRUCTURE

Field of the Invention

This invention relates to a system and method for searching an encrypted database for documents satisfying an expressive keyword access structure. In particular, the invention enables encrypted documents stored in a database to be searched to determine if there are documents that contain public-key encrypted keywords that satisfy an expressive keyword access structure, which contains conjunctive, disjunctive or any monotonic Boolean formulas whereby the search process is carried out without disclosing the underlying plaintext of the keywords in the encrypted documents.

Summary of the Prior Art

Most users tend to store various types of documents, including documents that contain private and sensitive information, in remote data servers. Such data servers are typically managed and owned by an external entity. While contractual obligations would compel the external entity to take security measures to ensure that no unauthorized access of documents stored within the data servers occur, there may be little that prevents the external entity itself from accessing the sensitive information contained within these documents should they wish to do so. Further, although the external entity would have taken security measures to prevent unauthorized access to documents stored within the data servers, security breaches may still inadvertently occur resulting in access to the documents being granted accidentally.

This problem may be addressed by encrypting the information contained within the documents and storing the documents in the data servers in an encrypted format. By doing

so, only the owner of the document and authorized users are able to access the information contained within the documents as the user would be the only one who would possess the secret key to decrypt the document. In order to facilitate the use and sharing of data contained within the encrypted documents with multiple authorized users, it is highly desirable to have a searchable encryption scheme which allows the database provider to search through encrypted documents stored in its database on behalf of authorized users without gleaning information about the underlying plaintext contained therein.

Searchable encryption schemes that have been proposed thus far include a method that utilizes private-key encryption. This private-key encryption scheme only allows a single user to search and retrieve their data. As such, such a scheme is not suitable for use in cases whereby the encrypted documents are to be shared and/or has to be searchable by multiple authorized data users and/or data providers. Another approach that has been proposed involves the use of private information retrieval (PIR) protocols. These protocols allow users to retrieve certain data-items from a database and the database publicly stores the data without revealing the data-item to the database administrator. This approach is not ideal as it requires the data to be made publicly available and this increases the likelihood of leaks occurring. Other searchable encryption schemes that are commonly used in the art only support single or conjunctive keyword searches. Existing schemes that are able to perform expressive keyword searches are typically computationally inefficient as these schemes are based on bilinear pairings over the composite-order groups.

For the above reasons, those skilled in the art are constantly striving to come up with a system and method that allows a user to search an encrypted database for documents containing a particular expressive keyword access structure in a computationally efficient manner.

Summary of the Invention

We provide systems and methods to improve the efficiency of searchable encryption schemes for keyword search policies expressed in conjunction, disjunctive or monotonic Boolean formulas as set out by the embodiments in accordance with the invention.

A first advantage of embodiments of systems and methods in accordance with embodiments of the invention is that the proposed system and method results in a scheme that is much faster than existing solutions.

A second advantage of embodiments of systems and methods in accordance with embodiments of the invention is that encrypted documents containing an expressive keyword access structure may be retrieved from a cloud server without revealing the contents of the encrypted documents to the cloud server.

A third advantage of embodiments of systems and methods in accordance with embodiments of the invention is that keywords associated with the generated trapdoors will not be readable by unauthorized users even though the trapdoors are transmitted through public channels.

According to a first aspect of the invention, a method for querying an encrypted database for documents containing an expressive keyword access structure using a computer server comprises receiving public parameters from a trapdoor server , wherein the public parameter is constant; generating a public key g^{γ} and a private key γ based on the received public parameters where g is a generator for group G and γ is a random element; receiving a trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure, wherein the trapdoor $T_{M,\rho}$ comprises: trapdoor parameters, a Linear Secret Sharing Scheme (LSSS) matrix M having l rows and n columns, and a function ρ mapping the rows of the LSSS

matrix M to generic names in a keyword set W associated with the expressive keyword access structure, wherein the keyword set W comprises the generic names and keyword values, and; processing documents in the encrypted database that are pre-tagged with corresponding generic names N and corresponding ciphertext CT, wherein the processing the documents comprises: retrieving documents from the encrypted database tagged with at least one of the generic names contained in the keyword set W; computing a search token for each retrieved document, whereby each search token is computed based on parameters obtained from the ciphertext CT tagged to each document, parameters contained in the trapdoor $T_{M,\rho}$, the private key γ and the public parameters; determining, for each retrieved document, if the computed search token for the document matches with a first parameter in ciphertext CT tagged to the document, whereby if a match is determined, indicating the document contains the expressive keyword access structure. In the setup phase of construction, the public parameter is constant rather than linear to the number of keywords, so the method can be used to support the search of large number of keywords. This is very meaningful in a cloud storage system, where data are from different parties. If the keyword number allowed in the system is limited, then it is difficult cover all the keywords that might appear in the documents to meet the searching requirements from all kinds of users.

With reference to the first aspect, in a first possible implementation manner of the first aspect, the public parameters received from the trapdoor server is defined by:

$$\text{public parameters} = (\mathrm{H}, \mathrm{g}, \mathrm{u}, \mathrm{h}, \omega, \mathrm{g}_1, \mathrm{g}_2, \mathrm{g}_3, \mathrm{g}_4, \hat{\mathrm{e}}(\mathrm{g}, \mathrm{g})^{\alpha})$$

where H is a collision-resistant hash function mapping elements in G1 to elements in G, G is a group of prime order p with the generator g, $\hat{e}$: G x G → G1 is a symmetric bilinear pairing function, where $\mathrm{u}, \mathrm{h}, \omega$ are random group elements in G, and $\mathrm{g}_1 = \mathrm{g}^{\mathrm{d}_1}$, $\mathrm{g}_2 = \mathrm{g}^{\mathrm{d}_2}$, $\mathrm{g}_3 = \mathrm{g}^{\mathrm{d}_3}$, $\mathrm{g}_4 = \mathrm{g}^{\mathrm{d}_4}$, where $\alpha, \mathrm{d}_1, \mathrm{d}_2, \mathrm{d}_3, \mathrm{d}_4 \in \mathrm{Z}_{\mathrm{p}}^*$where $\mathrm{Z}_{\mathrm{p}}^*$ are non-zero residuals of modular p.

With reference to the first possible implementation manner of the first aspect, in a second possible implementation manner of the first aspect, the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M, \rho), T, T_0, \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i\in[1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$$T = g^r,\ T_0 = g^{r_0},\ T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}},\ T_{i,2} = H(\hat{e}(g^{r_0}, g^{\gamma})^r) g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$$

$$T_{i,3} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_2},\ T_{i,4} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,1}}\right)^{-d_1},\ T_{i,5} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,2}}\right)^{-d_4},$$

$$T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3},\ v_i = M_i \cdot \vec{y}$$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix M, $\vec{y} = (\alpha, y_2, \ldots, y_n)^{\perp}$ where $y_2, \ldots, y_n,\ r,\ r_0,\ t_{1,1},\ t_{1,2},\ \ldots,\ t_{l,1},\ t_{l,2} \in Z_p$ where Z_p are residuals of modular p.

With reference to the second possible implementation manner of the first aspect, in a third possible implementation manner of the first aspect, a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

$$\text{ciphertext } CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i\in[1,m]})$$

whereby m is the size of the keyword set of the document, and the first value C of the ciphertext CT is defined as:

$$C = \hat{e}(g, g)^{\alpha\mu}$$

where, $D = g^{\mu}$, $D_i = \omega^{-\mu}\left(u^{W_i}h\right)^{z_i}$,

$$E_{i,1} = g_1{}^{z_i - s_{i,1}},\ E_{i,2} = g_2^{s_{i,1}},\ \ F_{i,1} = g_3^{z_i - s_{i,2}},\ \ F_{i,2} = g_4^{s_{i,2}}$$

where W_i is the i-th keyword value in a keyword set of the document, $W_1, \ldots, W_m$ are the corresponding keyword values of the keyword set of the document, and $\mu, s_{1,1}, s_{1,2}, \ldots, s_{m,1}, s_{m,2}, z_1, \ldots, z_m \in Z_p$.

With reference to the third possible implementation manner of the first aspect, in a fourth possible implementation manner of the first aspect, the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{i \in I}(\hat{e}(D, T_{i,1})\hat{e}(D_i, \frac{T_{i,2}}{H(\hat{e}(T,T_0)^{\gamma})})\hat{e}(E_{i,1}, T_{i,3})\hat{e}(E_{i,2}, T_{i,4})\hat{e}\left(F_{i,1}, T_{i,5}\right)\hat{e}\left(F_{i,2}, T_{i,6}\right))^{\omega_i}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i \in I} w_i M_i = (1,0,\ldots,0)$.

With reference to the first aspect, in a fifth possible implementation manner of the first aspect the public parameters received from the trapdoor server is defined by:

$$\text{public parameters} = (\hat{g}, u, h, \omega, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, \hat{e}(g, \hat{g})^{\alpha})$$

where $\hat{G}$ is a group of prime order p with the generator $\hat{g}$, $\hat{e}$: G x $\hat{G}$ → G1 is an asymmetric bilinear pairing function, where u, h, ω are random group elements in G, and $\hat{g}_1 = \hat{g}^{d_1}$, $\hat{g}_2 = \hat{g}^{d_2}$, $\hat{g}_3 = \hat{g}^{d_3}$, $\hat{g}_4 = \hat{g}^{d_4}$ where $\alpha, d_1, d_2, d_3, d_4 \in Z_p^*$ wherein the public key g^{γ} is replaced by $\hat{g}^{\gamma}$.

With reference to the fifth possible implementation manner of the first aspect, in a sixth possible implementation manner of the first aspect, the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M, \rho), T, \left\{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\right\}_{i\in[1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$T = \hat{g}^{r}$, $T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1}+d_3 d_4 t_{i,2}}$, $T_{i,2} = \hat{g}^{\gamma r} \cdot \hat{g}^{d_1 d_2 t_{i,1}+d_3 d_4 t_{i,2}}$

$T_{i,3} = \left(\left(u^{W_{\rho(i)}}h\right)^{t_{i,1}}\right)^{-d_2}$, $T_{i,4} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,1}}\right)^{-d_1}$, $T_{i,5} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,2}}\right)^{-d_4}$,

$T_{i,6} = \left(\left(u^{W_{\rho(i)}}h\right)^{t_{i,2}}\right)^{-d_3}$, $v_i = M_i \cdot \vec{y}$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix M, $\vec{y} = (\alpha, y_2, \ldots, y_n)^{\perp}$ where $y_2, \ldots, y_n$, r, r_0, $t_{1,1}$, $t_{1,2}$, $\ldots$, $t_{l,1}$, $t_{l,2} \in Z_p$ where Zp are non-zero residuals of a modular positive integer p.

With reference to the sixth possible implementation manner of the first aspect, in a seventh possible implementation manner of the first aspect, a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

$$\text{ciphertext } CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i\in[1,m]})$$

whereby m is a size of the keyword set of the document, and a first parameter C of the ciphertext CT is defined as:

$$C = \hat{e}(g, \hat{g})^{\alpha\mu}$$

where, $D = \hat{g}^{\mu}$, $D_i = \omega^{-\mu}\left(u^{W_i}h\right)^{z_i}$, $E_{i,1} = \hat{g}_1{}^{z_i - s_{i,1}}$, $E_{i,2} = \hat{g}_2^{s_{i,1}}$, $F_{i,1} = \hat{g}_3^{z_i - s_{i,2}}$, $F_{i,2} = \hat{g}_4^{s_{i,2}}$

where W_i is the i-th keyword value in a keyword set of the document, $W_1, \ldots, W_m$ are the corresponding keyword values of the keyword set of the document, and

$\mu, s_{1,1}, s_{1,2}, \ldots, s_{m,1}, s_{m,2}, z_1, \ldots, z_m \in Z_p$where Zp are non-zero residuals of a modular positive integer p.

With reference to the seventh possible implementation manner of the first aspect, in an eighth possible implementation manner of the first aspect, the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{i\in I}(\hat{e}(T_{i,1}, D)\hat{e}(D_i, \frac{T_{i,2}}{T^{\gamma}})\hat{e}(T_{i,3}, E_{i,1})\hat{e}(T_{i,4}, E_{i,2})\hat{e}(T_{i,5}, F_{i,1})\hat{e}(T_{i,6}, F_{i,2}))^{\omega_i}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i\in I} w_i M_i = (1,0, \ldots,0)$.

With reference to the first aspect, in a ninth possible implementation manner of the first aspect, the public key g^{γ} is g and the private key γ is 1.

With reference to the fifth possible implementation of the first aspect, in a tenth possible implementation manner of the first aspect, the public key g^{γ} is $\hat{g}$ and the private key γ is 1.

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 a block diagram of a system for searching an encrypted database for an expressive keyword access structure in accordance with embodiments of the invention;

Figure 2 illustrating a block diagram representative of processing systems providing embodiments in accordance with embodiments of the invention;

Figure 3 illustrating an exemplary access tree for converting a monotonic Boolean formula into an equivalent Linear Secret Sharing Scheme (LSSS) Matrix in accordance with embodiments of the invention; and

Figure 4 illustrating a flow diagram of a process for querying an encrypted database of documents containing an expressive keyword access structure in accordance with embodiments of the invention.

Detailed Description

This invention relates to a system and method for searching an encrypted database for documents satisfying an expressive keyword access structure whereby the expressive keyword access structure may include conjunctive, disjunctive or any monotonic Boolean formulas of a set of keywords. In particular, the invention enables public-key encrypted documents stored in a database to be searched for such expressive keyword access structures whereby the search process, which may be initiated by any one of a plurality of authorized users, is carried out without disclosing the underlying plaintext of the words in the encrypted documents.

Figure 1 illustrates a block diagram of a system that includes modules and devices that execute processes to provide a method for searching an encrypted database for expressive keyword access structures in accordance with embodiments of the invention. System 100 as illustrated in Figure 1 comprises encryption module 110, cloud server 115, trapdoor server 120, and searching server 125. Module 110, servers 115, 120 and 125 may all be connected through wired connections or may be wirelessly connected to each other either through direct means such as fibre-network connections or indirect means – that is via the Internet. Module 110 may be provided within, but is not limited to, any device that is able

to carry out computing and wireless communicative functions such as a smart phone, a computer, a tablet computer, a mobile computer, a netbook, a wearable electronic device such as smart watch, smart plugs, or transceivers that may be found in smart devices or Internet of Things (IoT) enabled devices, and etc.

As for servers 115, 120 and 125, these servers may comprise secure cloud servers or remotely located secure servers which are able to communicate wirelessly with module 110 and multiple data owners and/or authorized users either through the Internet or through other communicative means. These communicative means may comprise wired networks or wireless networks such as, but are not limited to, cellular networks, satellite networks, telecommunication networks, Wide Area Networks (WAN), Wireless-Fidelity (Wi-Fi), Bluetooth, or Near Field Communication (NFC). In addition to being configured to carry out computing and wireless communicative functions, servers 115, 120 and 125 are also configured to store encrypted documents in a database and/or store private keys. In embodiments of the invention, for increased security, each of servers 115, 120 and 125 may be provided at different locations for increased security.

Figure 2 illustrates a block diagram representative of components of module 200 that may be provided within module 110 and servers 115, 120 and 125 for implementing embodiments in accordance with embodiments of the invention. One skilled in the art will recognize that the exact configuration of each module provided within the electronic devices or the servers may be different and the exact configuration of module 200 may vary and Figure 2 is provided by way of example only.

In embodiments of the invention, module 200 comprises controller 201 and user interface 202. User interface 202 is arranged to enable manual interactions between a user and module 200 and for this purpose includes the input/output components required for the

user to enter instructions to control module 200. A person skilled in the art will recognize that components of user interface 202 may vary from embodiment to embodiment but will typically include one or more of display 240, keyboard 235 and track-pad 236.

Controller 201 is in data communication with user interface 202 via bus 215 and includes memory 220, processor 205 mounted on a circuit board that processes instructions and data for performing the method of this embodiment, an operating system 206, an input/output (I/O) interface 230 for communicating with user interface 202 and a communications interface, in this embodiment in the form of a network card 250. Network card 250 may, for example, be utilized to send data from electronic device 200 via a wired or wireless network to other processing devices or to receive data via the wired or wireless network. Wireless networks that may be utilized by network card 250 include, but are not limited to, Wireless-Fidelity (Wi-Fi), Bluetooth, Near Field Communication (NFC), cellular networks, satellite networks, telecommunication networks, Wide Area Networks (WAN) and etc.

Memory 220 and operating system 206 are in data communication with CPU 205 via bus 210. The memory components include both volatile and non-volatile memory and more than one of each type of memory, including Random Access Memory (RAM) 220, Read Only Memory (ROM) 225 and a mass storage device 245, the last comprising one or more solid-state drives (SSDs). Memory 220 also includes secure storage 246 for securely storing secret keys, or private keys. It should be noted that the contents within secure storage 246 are only accessible by a super-user or administrator of module 200 and may not be accessed by any user of module 200. One skilled in the art will recognize that the memory components described above comprise non-transitory computer-readable media and shall be taken to comprise all computer-readable media except for a transitory, propagating signal.

Typically, the instructions are stored as program code in the memory components but can also be hardwired. Memory 220 may include a kernel and/or programming modules such as a software application that may be stored in either volatile or non-volatile memory.

Herein the term “processor” is used to refer generically to any device or component that can process such instructions and may include: a microprocessor, microcontroller, programmable logic device or other computational device. That is, processor 205 may be provided by any suitable logic circuitry for receiving inputs, processing them in accordance with instructions stored in memory and generating outputs (for example to the memory components or on display 240). In this embodiment, processor 205 may be a single core or multi-core processor with memory addressable space. In one example, processor 205 may be multi-core, comprising—for example—an 8 core CPU.

With reference to Figure 1, before any documents may be encrypted by module 110 and uploaded into cloud server 115, trapdoor server 120 will first generate public parameters *pars* and master private key *msk*. Trapdoor server 120 then provides public parameters *pars* to encryption module 110 and searching server 125. It should be noted that the master private key *msk* is only kept by trapdoor server 120 and is not transmitted to any other entity.

In order to generate public parameters *pars* and master private key *msk*, trapdoor server 120 first takes the security parameter 1^{λ} as its input and then randomly chooses a group G of prime order p, a generator g and random group elements $u, h, w \in G$. In addition, trapdoor server 120 also randomly chooses $\alpha, d_1, d_2, d_3, d_4 \in Z_p$, and computes $g_1 = g^{d_1}$, $g_2 = g^{d_2}$, $g_3 = g^{d_3}$, $g_4 = g^{d_4}$. Finally, based on the above, trapdoor server 120 then generates the master private key $msk = (\alpha, d_1, d_2, d_3, d_4)$ and publishes the public parameter $pars = (H, g, u, h, w, g_1, g_2, g_3, g_4, \hat{e}(g,g)^{\alpha} \hat{e}(g,g)^{\alpha})$, where H is a collision-resistant hash function that maps an element belonging G_1 to an element belonging G.

It should be noted at this stage, that the following computational assumptions are used in embodiments of the invention. For example, when the bracket "[ ]" is used to encapsulate a positive integer *n*, it shall be understood that [*n*] denotes the set of all positive integers less or equal to *n*, i.e., [*n*]={1, 2,...*n*}. It shall also be understood that the term "Z_p" represents all residuals of a modular positive integer *p*, i.e., Z_p= {0, 1, 2, …, p-1}, while Z_p^* denotes the set of all non-zero residuals of modular *p*, i.e. Z_p^*={1, 2, …, p-1}.

Bilinear Pairing Functions:

For this function, both *G* and G_1 are understood to comprise two groups of prime order *p*, and *g* is a generator of *G*. Hence, the function $\hat{e}: G \times G \to G_1$ is understandably identified as a bilinear map if it has the following properties:

- Computable: For all $f, h \in G$, $\hat{e}(f,h)$ can be computed efficiently.
- Bilinear: For all a, b, ∈ Z_p, $\hat{e}(g^a, g^b) = \hat{e}(g,g)^{ab}$.
- Non-degenerate: $\hat{e}(g,g) \neq 1$, where 1 is the identity of group G_1.

In this example, *G* is also identified as a bilinear group if the group operation in *G* is efficiently computable and there exists a group G_1 and an efficiently computable bilinear map $\hat{e}: G \times G \to G_1$ as above.

Decisional Bilinear Diffie-Hellman Assumption:

For any probabilistic polynomial-time algorithm, given (g, g^a, g^b, g^c), it is difficult to distinguish $\hat{e}(g,g)^{abc}$ from *Z*, where *g* is a generator of *G*, and $Z \in G_1$, and a, b, c ∈ Z_p, are chosen independently and uniformly at random, while $\hat{e}: G \times G \to G_1$ is a computable bilinear map as defined above.

Decisional (q-2) Assumption:

For any probabilistic polynomial-time algorithm, given $g, g^x, g^y, g^z, g^{(xz)^2}$, and $\{g^{xzb_i/b_j}, g^{yb_i/b_j^2}, g^{xyzb_i/b_j}, g^{(xz)^2 b_i/b_j} \mid \forall i, j \in [q]\}$, it is difficult to distinguish $\hat{e}(g,g)^{abc}$ from Z, where q is a given integer, g is a generator of G, and $Z \in G_1$, and x, y, z, $b_1,\ldots,b_q \in Z_p$, are chosen independently and uniformly at random, while $\hat{e}: G \times G \to G_1$ is a computable bilinear map as defined above.

Decisional Linear Assumption:

For any probabilistic polynomial-time algorithm, given $g, g^{x_1}, g^{x_2}, g^{x_1 x_3}$ and $g^{x_2 x_4}$, it is difficult to distinguish $g^{x_3+x_4}$ from Z, where q is a given integer, g is a generator of G, and $Z \in G_1$, and $x_1, x_2, x_3, x_4 \in Z_p$ are chosen independently and uniformly at random, while $\hat{e}: G \times G \to G_1$ is a computable bilinear map as defined above.

Upon receiving the public parameter *pars* from trapdoor server 120, searching server 125 will then randomly choose $\gamma \in Z_p$, and based on thus selected parameter, search server 125 will then compute the public and private key pair $(\mathrm{pk_s}, \mathrm{sk_s})\ as\ (g^{\gamma}, \gamma)$. Once the key pair has been computed, this key pair is then provided to trapdoor server 120 from searching server 125. Trapdoor server 120 will utilize these parameters later on in the generation of trapdoors for expressive keyword access structures as requested by authorized data users.

Before a document owner uploads an encrypted document to be stored in cloud server 115, the document owner will first have to append each encrypted document with encrypted keyword sets. This is accomplished with the owner first identifying key terms/phrases or index terms that may be used to identify the content of a document. These key/index terms or phrases are then appended to the document as a keyword set. In Figure

1, documents 105 illustrate a set of documents, from Document 1 – n, that have been appended with their respective keyword sets. It should be noted that each keyword in a keyword set is divided into a generic name N = {N(1),…N(i)} and a keyword value W = {W(1),…,W(i)} whereby the value (i) illustrates the relationship between a particular generic name with a particular keyword value. For example, the keyword set_1 for Document 1 may be taken to contain “Affiliation = City Hospital”, “Department = Medicine”, “Illness = Diabetes”, and “Gender = Male”. The generic names N for this keyword set_1 would be N(1) = Affiliation, N(2) = Department, N(3) = Illness and N(4) = Gender while the keyword values W for this keyword set_1 would be W(1) = City Hospital, W(2) = Medicine, W(3) = Diabetes and W(4) = Male. Further, in embodiments of the invention, each keyword value in W(i) may be represented by an numerical value such as, but not limited to, the keyword’s equivalent ASCII value.

By dividing each keyword contained within a keyword set into a generic name and a keyword value, an authorized user of system 100 may then later on search and subsequently retrieve all encrypted documents containing certain keywords that satisfy a certain keyword access structure. An example of a monotone type keyword access structure is “Affiliation = City Hospital AND (Department = Medicine OR (Illness = Diabetes AND Gender = Male))”.

In this description, any references made to a keyword access structure, an access policy or a search predicate all refer to monotone type access structures. However, it should be noted that other types of general access structures may be realized in embodiments of this invention if the attribute universe were to be split in half and by treating the attributes of one half as the negated (NOT) version of the attributes in the other half. A person skilled in the art will also note that the keyword access structures may be described in terms of

monotonic Boolean formulas and these formulas may be converted into a corresponding Linear Secret Sharing Scheme (LSSS) matrix. This will be described in greater detail with reference to Figures 3 and 4 in the later sections.

Once all the keyword sets containing keywords (and their corresponding generic names and keyword values) have been appended to each respective document, documents 105 comprising Documents 1-n with their respective appended keyword sets are then provided to encryption module 110. To recap, as the initial setup procedures have been completed as previously discussed, encryption module 110 would have received and stored the public parameters *pars* that were provided by trapdoor server 120. At this stage, it should be noted that the content or plaintext in Documents 1-n may be encrypted using any encryption scheme known to a person skilled in the art and that ciphertext contained in Documents 1-n may subsequently be decrypted using a corresponding decryption scheme known to a person skilled in the art. The encryption of the content of Documents 1-n may take place within encryption module 110.

Upon receiving documents 105 (comprising Documents 1-n with their respective appended keyword sets), encryption module 110 will then select a first document together with its appended keyword set, i.e. Document 1 that has been appended with Keyword Set_1. As previously discussed, each keyword contained within each keyword set is divided into a generic name and a keyword value. Hence, Keyword Set_1 comprises of two groups of plaintext - a first group comprising of generic names and a second group comprising of the generic names' corresponding keyword values. In embodiments of the invention, only the plaintext contained in the keyword values are encrypted into ciphertext using encryption module 110.

In order to encrypt the keyword values in Keyword Set_1 into ciphertext CT_1, module 110 will initiate the process by retrieving the public parameter *pars* from its records and keyword values W = {W(1),…,W(m)} from Keyword Set_1 whereby "m" represents the m^{th} keyword value of W in Keyword Set_1. As mentioned above, each individual keyword value may be represented by a numerical value. An integer m is then used to denote the overall size of keyword value W, and $W_1, \dots, W_m \in Z_p$ is used to represent the values of W. Module 110 then randomly chooses values for elements $\mu, s_{1,1}, s_{1,2}, \dots, s_{m,1}, s_{m,2}, z_1, \dots, z_m \in Z_p$, and computes a ciphertext CT_1 for the keyword values W using the following equation:

$$CT = (C , D , \{D_i , E_{i,1} , E_{i,2} , F_{i,1} , F_{i,2}\}_{i\in[1,m]})$$

$$\mathrm{CT} = (\mathrm{C}, \mathrm{D}, \{(\mathrm{D_i}, \mathrm{E_{i,1}}, \mathrm{E_{i,2}}, \mathrm{F_{i,1}}, \mathrm{F_{i,2}})_{\mathrm{i}\in[1,\mathrm{m}]})$$

where $\mathrm{C} = \hat{\mathrm{e}}(\mathrm{g},\mathrm{g})^{\alpha\mu}$, $\mathrm{D} = \mathrm{g}^{\mu}$, $\mathrm{D_i} = \mathrm{w}^{-\mu}(\mathrm{u}^{\mathrm{W_i}}\mathrm{h})^{\mathrm{z_i}}$, $\mathrm{E_{i,1}} = \mathrm{g_1}^{\mathrm{z_i}-\mathrm{s_{i,1}}}$, $\mathrm{E_{i,2}} = \mathrm{g_2}^{\mathrm{s_{i,1}}}$, $\mathrm{F_{i,1}} = \mathrm{g_3}^{\mathrm{z_i}-\mathrm{s_{i,2}}}$, $\mathrm{F_{i,2}} = \mathrm{g_4}^{\mathrm{s_{i,2}}}$.

Once ciphertext CT_1 has been computed for Keyword Set_1, module 110 will then replace all the keyword values in Keyword Set_1 with ciphertext CT_1. Document 1 will then be transmitted to cloud server 115 at step 132 and it should be noted that at this step, the contents of Document 1 would have been encrypted and that this Document 1 is appended with Keyword Set_1, whereby Keyword Set_1 includes the generic names, which remain un-encrypted, and ciphertext CT_1. This process is then repeated for all the documents contained within documents 105 until all the documents have been uploaded into cloud server 115 whereby each encrypted document stored in cloud server 115 would be appended with its tagged or appended keyword set whereby each keyword set contains a list of generic names and its ciphertext CT.

When an authorized user wishes to obtain all encrypted documents stored in cloud server 115 that contains a set of keywords *SW* that satisfy expressive keyword access structure 133, access structure 133 that contains keywords *SW* will first have to be converted into a Linear Secret Sharing Scheme (LSSS) matrix M. In this embodiment of the invention, access structure 133 comprises a monotonic Boolean formula and this Boolean formula is converted into an equivalent LSSS matrix by the authorized user.

In order to do so, it shall be assumed that the Boolean formula may be represented as an access tree, of which the interior nodes comprise “AND” and “OR” logic operators or gates, and the leaf nodes represent the corresponding attributes. The values in the vector (1, 0, ..., 0) are denoted as the sharing vector for the LSSS matrix. Further, it is also assumed that the root node of the tree is labeled with a vector of length 1, which is vector (1), and that each node is labeled with a vector determined by the vector assigned to its parent node. In addition, there also exists a global counter variable c which is preset to 1.

If the parent node is an OR gate labelled by a vector $\vec{v}$, then its children will be labelled by $\vec{v}$ as well, and the value of its global counter variable c remains the same. Alternatively, if the parent node is an AND gate labelled by a vector $\vec{v}$, then v will be padded with 0s at the end, if required, in order to increase its length to c. One of its children will then be labelled with the vector $\vec{v} \mid 1$ and the other with the vector $(0, ..., 0 \mid -1)$, where (0, ..., 0) denotes the zero vector of length c. Note that these two vectors when summed equate to $\vec{v}|0$. Once this is done, the value of c is increased by one i.e. $c = c + 1$. Once the labelling of the entire tree is finished, the vectors labelling the leaf nodes form the rows l of the LSSS matrix. If these vectors forming these rows are of different lengths, the shorter vectors will be padded with 0s at the end so that all the vectors will be equal in length.

For example, assume that keyword access structure 133 comprises the following: "**Affiliation** = City Hospital AND (**Department** = Medicine OR (**Illness** = Diabetes AND = **Gender** = Male))". With reference to Figure 3 and using the method described above, the LSSS matrix for this access structure 133 is generated as follows. As the parent node 302 is an AND gate labelled with the vector $\vec{v}$=1 , the root AND node 302 of this tree will be labelled as (1) while its global counter variable c = 1. For the left and right child of node 302, the global counter variable c is increased to 2. Its left child 305, the leaf node corresponding to Affiliation = City Hospital, is then labelled as (1, 1) because this node is labelled with ($\vec{v}$ | 1), while its right child 310, the OR node, is labelled as (0, −1) because this node is labelled with (0 | −1) when c = 2.

For the left and right child of node 310, as the node 310 is an OR gate labelled with (0,-1), then its children will be labelled by (0,-1) as well, and the value of its global counter variable c remains the same at c = 2. The left child 315 of the OR node 310 corresponds to Department = Medicine and is labelled as (0, −1) while its right child 320 is an AND node that is then labelled as (0, −1).

For the left and right child of node 320, as node 320 is an AND gate labelled with (0-1), the global counter variable c will be increased to 3. The left child 325 of the AND node 320 corresponding to Illness = Diabetes, is then labelled as (0, −1, 1) because this node is labelled with (0,-1 | 1), and the right child 330, corresponding to Gender = Male, is then labelled as (0, 0, −1) because this node is labelled with (0,0 | −1).

The labelling of the entire tree is now finished and the vectors labelling the leaf nodes, i.e. nodes 305, 315, 325, 330, form the rows l of the LSSS matrix, which in this example l = 4. For nodes 305 and 315, as the vectors forming these rows are of different lengths from nodes 325 and 330, the vectors in nodes 305 and 315 will be padded with 0s at the end and

this results in node 305 being represented as (1, 1, 0) and node 315 being represented as (0, -1, 0). Hence, the resulting LSSS matrix M for this example is:

$$\mathrm{M} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

Once access structure 133 containing keywords *SW* has been converted into LSSS matrix *M*, the access structure 133 may then be represented as LSSS access structure $(M, \rho, W_{\rho(i)})$ instead where, ρ is a function that associates the rows of M to the generic names in keywords *SW*, and $\{\mathrm{W}_{\rho(\mathrm{i})}\}$ are the corresponding keyword values in keywords *SW*. For example, for row 1 in LSSS matrix M, $\rho(1)$= "Affiliation" and $W_{\rho(1)}$= (1, 1, 0).

The LSSS access structure is then transmitted from the authorized user to trapdoor server 120 at step 134. Once trapdoor server 120 confirms that the request is from a user that has been authorized to use trapdoor server 120, trapdoor server 120 shall use the public parameter pars, the server public key $\mathrm{pk_s} = \mathrm{g}^{\gamma}$, the master private key msk and the LSSS access structure $(\mathrm{M}, \rho, \{\mathrm{W}_{\rho(\mathrm{i})}\})$ to generate the trapdoor associated with this LSSS access structure, where M is an $\mathrm{l} \times \mathrm{n}$ matrix over $\mathrm{Z_p}$, the function ρ maps the rows of M to generic keyword names, and $\{\mathrm{W}_{\rho(\mathrm{i})}\}$ are the corresponding keyword values as previously discussed. To recap, $\mathrm{M_i}$ is the i-th row of M for $\mathrm{i} \in \{1, \ldots, \mathrm{l}\}$, and $\rho(\mathrm{i})$ is the generic keyword name associated with this row by the mapping ρ. Trapdoor server 120 randomly chooses a vector $\vec{\mathrm{y}} = (\alpha, \mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{n})^{\perp}$ where $\mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{n} \in \mathrm{Z_p}$, $\mathrm{r}, \mathrm{r}_0 \in \mathrm{Z_p}$, $\mathrm{t}_{1,1}, \mathrm{t}_{1,2}, \ldots, \mathrm{t}_{\mathrm{l},1}, \mathrm{t}_{\mathrm{l},2} \in \mathrm{Z_p}$, and based on these parameters, trapdoor server 120 compute the trapdoor $\mathrm{T}_{\mathrm{M},\rho}$ using the following equation:

$$\mathrm{T}_{\mathrm{M},\rho} = ((\mathrm{M}, \rho), \mathrm{T}, \mathrm{T}_0, \{\mathrm{T}_{\mathrm{i},1}, \mathrm{T}_{\mathrm{i},2}, \mathrm{T}_{\mathrm{i},3}, \mathrm{T}_{\mathrm{i},4}, \mathrm{T}_{\mathrm{i},5},$$

$$\mathrm{T}_{\mathrm{i},6}\}_{\mathrm{i}\in[1,\mathrm{l}]})\mathrm{T}_{\mathrm{M},\rho} = \left((\mathrm{M}, \rho), \mathrm{T}, \mathrm{T}_0, \{\mathrm{T}_{\mathrm{i},1}, \mathrm{T}_{\mathrm{i},2}, \mathrm{T}_{\mathrm{i},3}, \mathrm{T}_{\mathrm{i},4}, \mathrm{T}_{\mathrm{i},5}, \mathrm{T}_{\mathrm{i},6}\}_{\mathrm{i}\in[1,\mathrm{l}]}\right)$$

where $T = g^{r}$, $T_0 = g^{r_0}$, $T_{i,1} = g^{v_i} w^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$, $T_{i,2} = H(\hat{e}(g^{r_0}, g^{\gamma})^{r}) g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$,

$T_{i,3} = \left(\left(u^{(W_{\rho(i)})} h\right)^{t_{i,1}}\right)^{-d_2}$, $T_{i,4} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_1}$, $T_{i,5} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_4}$, $T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3}$,

where $v_i = M_i \cdot \vec{y}$ is the value associated with the row M_i of M. It should be noted that when the trapdoor $T_{M,\rho}$ is computed, trapdoor $T_{M,\rho}$ only includes the partial hidden access structure (M, ρ) and the actual names of the keyword values are omitted.

The trapdoor $T_{M,\rho}$ is then transmitted at step 136 from trapdoor server 120 to the authorized user. The authorized user then transmits the trapdoor $T_{M,\rho}$ for LSSS access structure $(M, \rho, \{W_{\rho(i)}\})$ to searching server 125 at step 138 so that searching server 125 may search through the encrypted documents and their respective appended keyword sets for documents that contain the set of keywords *SW* that satisfy expressive keyword access structure 133.

Upon receiving trapdoor $T_{M,\rho}$ for LSSS access structure $(M, \rho, \{W_{\rho(i)}\})$, searching server 125 will retrieve documents appended with keyword sets that contain at least one of the generic names as found in the function ρ of the received trapdoor $T_{M,\rho}$. Such an initial matching of the generic keyword names greatly reduces the overall searching time of the documents.

Searching server 125 will then use the public parameter $pars$, the private key sk_s, and the received trapdoor $T_{M,\rho}$ for the LSSS access structure $(M, \rho, \{W_{\rho(i)}\})$ to test each retrieved document to determine if the document contains the set of keywords *SW* that satisfies the expressive keyword access structure 133 sought by the authorized user. Searching server 125 will initiate this process by selecting a first retrieved document together with its

appended keyword set containing its ciphertext CT. A search token is then computed for this first retrieved document whereby the search token is computed using the following equation:

$$\sum_{i\in I}(\hat{e}(D,T_{i,1})\hat{e}(D_i,\frac{T_{i,2}}{H(\hat{e}(T,T_0)^{\gamma})})\hat{e}(E_{i,1},T_{i,3})\hat{e}(E_{i,2},T_{i,4})\hat{e}\left(F_{i,1},T_{i,5}\right)\hat{e}\left(F_{i,2},T_{i,6}\right))^{w_i}=C$$

where the search token equals to a "0" value if this equation is not satisfied and conversely, the search token equals to a "1" if this equation is satisfied. It should be noted that the parameters used to compute the search token are obtained from the parameters contained in the ciphertext CT that is contained in the keyword set appended to the first retrieved document. Further, it should be noted that $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M,ρ) and $\sum_{i\in I} w_i M_i = (1,0,\dots,0)$.

In another embodiment of the invention, in order to improve the performance efficiency of system 100, the modules in system 100 may be adapted for an asymmetric bilinear pairing $\hat{e}: G \times \hat{G} \rightarrow G_1$, in which bilinear pairings may be calculated faster than the symmetric bilinear pairings used in the previously described embodiment. In particular, system 100 may be transformed from symmetric bilinear maps to asymmetric bilinear maps.

In this embodiment of the invention, trapdoor server 120 performs the similar steps as described above except that the previously used parameters g_1, g_2, g_3, g_4 will be replaced by $\hat{g}_1$, $\hat{g}_2$, $\hat{g}_3$, $\hat{g}_4$ whereby $\hat{g}_1 = \hat{g}^{d_1}$, $\hat{g}_2 = \hat{g}^{d_2}$, $\hat{g}_3 = \hat{g}^{d_3}$, $\hat{g}_4 = \hat{g}^{d_4}$, and H is removed from the public parameter *pars*.

In this embodiment of the invention, searching server 125 will repeat the similar steps as described above except that the public and private key pair (pk_s,sk_s) is computed as $(\hat{g}^{\gamma}.\gamma)$.

Similarly, when trapdoor server 120 computes the trapdoor $T_{M,\rho}$ using the equation described above, in this embodiment of the invention, the parameters from the above equation are replaced with the following: $T = \hat{g}^r$, $T_{i,2} = pk_s^r \hat{g}^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$ and the parameter T_0 is removed from the trapdoor $T_{M,\rho}$.

In this embodiment of the invention, when module 110 computes the ciphertext CT, the similar equation as previously described is utilized except that the term "g" is replaced with the term "$\hat{g}$". This replacement takes place in parameters D, $E_{i,1}$, $E_{i,2}$, $F_{i,1}$, $F_{i,2}$ during the computation of ciphertext CT.

Finally, in this embodiment of the invention, searching server 125 will compute the search token as described above except that the term $H(\hat{e}(T, T_0)^{\gamma})$ will be replaced by T^{γ}.

In accordance with an embodiment of the invention, a method for querying an encrypted database for documents containing an expressive keyword access structure using a computer server, comprises the following four steps:

Step 1, receiving public parameters from a trapdoor server;

Step 2, generating a public key g^{γ} and a private key γ based on the received public parameters where g is a generator for group G and γ is a random element;

Step 3, receiving a keyword set W for the expressive keyword access structure, wherein the keyword set W comprises generic names and keyword values, and receiving a trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure, wherein the trapdoor $T_{M,\rho}$ comprises;

trapdoor parameters, a Linear Secret Sharing Scheme (LSSS) matrix M having l rows and *n* columns, and a function ρ mapping the rows of the LSSS matrix M to the generic names in the keyword set W;

Step 4, processing documents in the encrypted database that are pre-tagged with corresponding generic names N and corresponding ciphertext CT, wherein the processing the documents comprises:

retrieving documents from the encrypted database tagged with at least one of the generic names contained in the keyword set W;

computing a search token for each retrieved document, whereby each search token is computed based on parameters obtained from the ciphertext CT tagged to each document, parameters contained in the trapdoor $T_{M,\rho}$, the private key γ and the public parameters;

determining, for each retrieved document, if the computed search token for the document matches with a first parameter in ciphertext CT tagged to the document, whereby if a match is determined, indicating the document contains the expressive keyword access structure.

In order to provide such a system or method, a process is needed for querying an encrypted database for documents containing an expressive keyword access structure. The following description and Figure 4 describes embodiments of processes that provide processes in accordance with this invention.

Figure 4 illustrates process 400 that is performed by a module or computer processor installed within an electronic device or server to query an encrypted database for documents containing an expressive keyword access structure in accordance with embodiments of this invention. Process 400 begins at step 405 whereby process 400 receives the public parameters from a server. The public parameters *pars* are stored in a secure memory or a secure database and are only accessible by process 400. Process 400 then proceeds to

generate public and private key pairs at step 410 using the parameters received at step 405. Process 400 then receives the trapdoor at step 420 for the expressive keyword access structure. A first document is then retrieved from the encrypted database by process 400 at step 425. If the generic names contained within the trapdoor matches with the generic names contained in the keyword set appended to the retrieved document at step 420, process 400 will add the retrieved document to a document list at step 435. Conversely, if the generic names contained within the trapdoor do not match with the generic names contained in the keyword set appended to the retrieved document at step 430, process 400 will proceed to step 440 instead and select a next document at this step. If at step 440 process 400 determines there is another document that has not yet been selected, process 400 will proceed to step 430 whereby steps 430 to 440 will repeat until all documents have been selected. Process 400 then proceeds to step 445.

However, at step 430, if process 400 determines that a retrieved document has been added to a document list at step 435, process 400 will then proceed to step 440 to select the next document. Process 400 repeats steps 435 to 440 until all documents have been processed by process 400.

At step 445, a first document from the retrieved document list is selected by process 400. Process 400 then proceeds to compute a search token for the selected document. If process 400 determines at step 455 that the search token matches with the first parameter in the ciphertext appended or tagged to the selected document, process 400 will proceed to step 460 where it will indicate that the selected document contains the expressive keyword access structure. Process 400 then proceeds to step 465 to select another document.

Conversely, if process 400 determines at step 455 that the search token does not match with the first parameter C, in ciphertext CT, process 400 will proceed to step 465

instead. At step 465, process 400 will select the next document and if there is another document to select, process 400 repeats from step 465 to 455. Alternatively, if all the documents have been selected, process 400 then ends.

The above is a description of embodiments of a system and process in accordance with the present invention as set forth in the following claims. It is envisioned that others may and will design alternatives that fall within the scope of the following claims.

CLAIMS:

1. A method for querying an encrypted database for documents containing an expressive keyword access structure using a computer server, the method comprising:

receiving public parameters from a trapdoor server, wherein the public parameter is constant;

generating a public key g^{γ} and a private key γ based on the received public parameters where g is a generator for a group G and γ is a random element;

receiving a trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure, wherein the trapdoor $T_{M,\rho}$ comprises:

trapdoor parameters, a Linear Secret Sharing Scheme (LSSS) matrix M having l rows and *n* columns, and a function ρ mapping the rows of the LSSS matrix M to generic names in a keyword set W associated with the expressive keyword access structure, wherein the keyword set W comprises the generic names and keyword values, and;

processing documents in the encrypted database that are pre-tagged with corresponding generic names N and corresponding ciphertext CT, wherein the processing the documents comprises:

retrieving documents from the encrypted database tagged with at least one of the generic names contained in the keyword set W;

computing a search token for each retrieved document, whereby each search token is computed based on parameters obtained from the ciphertext CT tagged to each document, parameters contained in the trapdoor $T_{M,\rho}$, the private key γ and the public parameters;

determining, for each retrieved document, if the computed search token for the document matches with a first parameter in ciphertext CT tagged to the document, whereby if a match is determined, indicating the document contains the expressive keyword access structure.

2. The method according to claim 1 wherein the public parameters received from the trapdoor server is defined by:

$$public\ parameters = (H, g, u, h, \omega, g_1, g_2, g_3, g_4, \hat{e}(g,g)^{\alpha})$$

where *H* is a collision-resistant hash function mapping elements in G_1 to elements in *G*, *G* is a group of prime order *p* with the generator *g*, $\hat{e}$: *G x G* → G_1 is a symmetric bilinear pairing function, where u, h, ω are random group elements in *G*, and $g_1 = g^{d_1}$, $g_2 = g^{d_2}$, $g_3 = g^{d_3}$, $g_4 = g^{d_4}$, where $\alpha, d_1, d_2, d_3, d_4 \in Z_p{}^*$ where Z_p* are non-zero residuals of modular *p*.

3. The method according to claim 2 wherein the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M,\rho), T, T_0, \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i\in[1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$$T = g^r, T_0 = g^{r_0}, T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}, T_{i,2} = H(\hat{e}(g^{r_0}, g^{\gamma})^r) g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$$

$$T_{i,3} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_2}, T_{i,4} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,1}}\right)^{-d_1}, T_{i,5} = \left(\left(u^{W_{\rho(i)}}\ h\right)^{t_{i,2}}\right)^{-d_4},$$

$$T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3}, v_i = M_i \cdot \vec{y}$$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix

M , $\vec{y} = (\alpha, y_2, \dots, y_n)^{\perp}$ where $y_2, \dots, y_n, r, r_0, t_{1,1}, t_{1,2}, \dots, t_{l,1}, t_{l,2} \in Z_p$ where Z_p are residuals of modular *p*.

4. The method according to claim 3 wherein a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

ciphertext $CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i\in[1,m]})$

whereby m is a size of the keyword set of the document, and the first parameter in ciphertext CT comprises a first value C that is defined as:

$$C = \hat{e}(g,g)^{\alpha\mu}$$

where, $D = g^{\mu}, D_i = \omega^{-\mu}(u^{W_i}h)^{z_i}$,

$$E_{i,1} = g_1{}^{z_i - s_{i,1}}, E_{i,2} = g_2^{s_{i,1}}, \ F_{i,1} = g_3^{z_i - s_{i,2}}, \ F_{i,2} = g_4^{s_{i,2}}$$

where W_i is the *i-th* keyword value in a keyword set of the document, $W_1, \dots, W_m$ are the corresponding keyword values of the keyword set of the document, and $\mu, s_{1,1}, s_{1,2}, \dots, s_{m,1}, s_{m,2}, z_1, \dots, z_m \in Z_p$.

5. The method according to claim 4 wherein the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{i\in I}(\hat{e}(D, T_{i,1})\hat{e}(D_i, \frac{T_{i,2}}{H(\hat{e}(T,T_0)^{\gamma})})\hat{e}(E_{i,1}, T_{i,3})\hat{e}(E_{i,2}, T_{i,4})\hat{e}\left(F_{i,1}, T_{i,5}\right)\hat{e}\left(F_{i,2}, T_{i,6}\right))^{\omega_i}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i\in I} w_i M_i = (1,0,\dots,0)$.

6. The method according to claim 1 wherein the public parameters received from the trapdoor server is defined by:

$$public\ parameters = (\hat{g}, u, h, \omega, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, \hat{e}(g,\hat{g})^{\alpha})$$

where $\hat{G}$ is a group of prime order *p* with the generator $\hat{g}$, $\hat{e}$: *G x* $\hat{G}$ → G_1 is an asymmetric bilinear pairing function, where u, h, ω are random group elements in *G*, and $\hat{g}_1 = \hat{g}^{d_1}$, $\hat{g}_2 = \hat{g}^{d_2}$, $\hat{g}_3 = \hat{g}^{d_3}$, $\hat{g}_4 = \hat{g}^{d_4}$ where $\alpha, d_1, d_2, d_3, d_4 \in Z_p{}^*$ where $Z_p{}^*$ denotes the set of all non-zero residuals of modular *p*, and wherein the public key $g^{\gamma}$is replaced by $\hat{g}^{\gamma}$.

7. The method according to claim 6 wherein the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M, \rho), T, \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i\in[1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$$T = \hat{g}^{r}, T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}, T_{i,2} = \hat{g}^{\gamma r} \cdot \hat{g}^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$$

$$T_{i,3} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_2}, T_{i,4} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_1}, T_{i,5} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_4},$$

$$T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3}, v_i = M_i \cdot \vec{y}$$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix M, $\vec{y} = (\alpha, y_2, \ldots, y_n)^{\perp}$ where $y_2, \ldots, y_n, r, r_0, t_{1,1}, t_{1,2}, \ldots, t_{l,1}, t_{l,2} \in Z_p$ where Z_p are non-zero residuals of a modular positive integer *p*.

8. The method according to claim 7 wherein a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

$$\text{ciphertext } CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i\in[1,m]})$$

whereby m is a size of the keyword set of the document, and a first parameter C of the ciphertext CT is defined as:

$$C = \hat{e}(g, \hat{g})^{\alpha\mu}$$

where, $D = \hat{g}^{\mu}, D_i = \omega^{-\mu}(u^{W_i}h)^{z_i}$,

$$E_{i,1} = \hat{g}_1^{\,z_i - s_{i,1}}, E_{i,2} = \hat{g}_2^{s_{i,1}}, \; F_{i,1} = \hat{g}_3^{z_i - s_{i,2}}, \; F_{i,2} = \hat{g}_4^{s_{i,2}}$$

where W_i is the *i-th* keyword value in a keyword set of the document, $W_1, \ldots, W_m$ are the corresponding keyword values of the keyword set of the document, and $\mu, s_{1,1}, s_{1,2}, \ldots, s_{m,1}, s_{m,2}, z_1, \ldots, z_m \in Z_p$ where Z_p denotes the set of all residuals of a modular positive integer *p*.

9. The method according to claim 8 wherein the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{\mathrm{i}\in\mathrm{I}}(\hat{\mathrm{e}}(\mathrm{T_{i,1}}, \mathrm{D})\hat{\mathrm{e}}(\mathrm{D_i}, \frac{\mathrm{T_{i,2}}}{\mathrm{T^{\gamma}}})\hat{\mathrm{e}}(\mathrm{T_{i,3}}, \mathrm{E_{i,1}})\hat{\mathrm{e}}(\mathrm{T_{i,4}}, \mathrm{E_{i,2}})\hat{\mathrm{e}}\big(\mathrm{T_{i,5}}, \mathrm{F_{i,1}}\big)\hat{\mathrm{e}}\big(\mathrm{T_{i,6}}, \mathrm{F_{i,2}}\big))^{\omega_{\mathrm{i}}}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i\in I} w_i M_i = (1,0,\ldots,0)$.

10. The method according to claim 1 wherein the public key g^{γ} is g and the private key γ is 1.

11. The method according to claim 6 wherein the public key $\hat{\mathrm{g}}^{\gamma}$ is $\hat{\mathrm{g}}$ and the private key γ is 1.

12. A computer server for querying an encrypted database for documents containing an expressive keyword access structure, the computer server comprising:

a processing unit; and

a non-transitory media readable by the processing unit, the media storing instructions that when executed by the processing unit, cause the processing unit to:

receive public parameters from a trapdoor server , wherein the public parameter is constant;

generate a public key g^{γ} and a private key γ based on the received public parameters where g is a generator for a group G and γ is a random element; receiving a trapdoor

$T_{M,\rho}$ associated with the expressive keyword access structure, wherein the trapdoor $T_{M,\rho}$ comprises:

trapdoor parameters, a Linear Secret Sharing Scheme (LSSS) matrix M having l rows and *n* columns, and a function ρ mapping the rows of the LSSS matrix M to generic names in a keyword set W associated with the expressive keyword access structure, wherein the keyword set W comprises the generic names and keyword values, and;

process documents in the encrypted database that are pre-tagged with corresponding generic names N and corresponding ciphertext CT by,

retrieving documents from the encrypted database tagged with at least one of the generic names contained in the keyword set W;

computing a search token for each retrieved document, whereby each search token is computed based on parameters obtained from the ciphertext CT tagged to each document, parameters contained in the trapdoor $T_{M,\rho}$, the private key γ and the public parameters;

determining, for each retrieved document, if the computed search token for the document matches with a first parameter in ciphertext CT tagged to the document, whereby if a match is determined, indicating the document contains the expressive keyword access structure.

13. The computer server according to claim 12 wherein the public parameters received from the trapdoor server is defined by:

$$public\ parameters = (H, g, u, h, \omega, g_1, g_2, g_3, g_4, \hat{e}(g,g)^{\alpha})$$

where *H* is a collision-resistant hash function mapping elements in G_1 to elements in *G*, *G* is a group of prime order *p* with the generator *g*, $\hat{e}$: *G x G* → G_1 is a symmetric bilinear

pairing function, where u, h, ω are random group elements in *G*, and $g_1 = g^{d_1}$, $g_2 = g^{d_2}$, $g_3 = g^{d_3}$, $g_4 = g^{d_4}$, where $\alpha, d_1, d_2, d_3, d_4 \in Z_p{}^*$ where Z_p* are non-zero residuals of modular *p*.

14. The computer server according to claim 13 wherein the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M,\rho), T, T_0, \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i\in[1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$$T = g^r, T_0 = g^{r_0}, T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}, T_{i,2} = H(\hat{e}(g^{r_0}, g^{\gamma})^r) g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$$

$$T_{i,3} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_2}, T_{i,4} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_1}, T_{i,5} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_4},$$

$$T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3}, v_i = M_i \cdot \vec{y}$$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix M, $\vec{y} = (\alpha, y_2, \ldots, y_n)^{\perp}$ where $y_2, \ldots, y_n, r, r_0, t_{1,1}, t_{1,2}, \ldots, t_{l,1}, t_{l,2} \in Z_p$ where Z_p are residuals of modular *p*.

15. The computer server according to claim 14 wherein a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

ciphertext $CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i\in[1,m]})$

whereby m is a size of the keyword set of the document, and the first parameter in ciphertext CT comprises a first value C that is defined as:

$$C = \hat{e}(g,g)^{\alpha\mu}$$

where, $D = g^{\mu}, D_i = \omega^{-\mu}(u^{W_i}h)^{z_i}$,

$$E_{i,1} = g_1{}^{z_i - s_{i,1}}, E_{i,2} = g_2^{s_{i,1}}, \ F_{i,1} = g_3^{z_i - s_{i,2}}, \ F_{i,2} = g_4^{s_{i,2}}$$

where W_i is the *i-th* keyword value in a keyword set of the document, $W_1, \ldots, W_m$ are the corresponding keyword values of the keyword set of the document, and $\mu, s_{1,1}, s_{1,2}, \ldots, s_{m,1}, s_{m,2}, z_1, \ldots, z_m \in Z_p$.

16. The computer server according to claim 15 wherein the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{i \in I}(\hat{e}(D, T_{i,1})\hat{e}(D_i, \frac{T_{i,2}}{H(\hat{e}(T,T_0)^{\gamma})})\hat{e}(E_{i,1}, T_{i,3})\hat{e}(E_{i,2}, T_{i,4})\hat{e}\left(F_{i,1}, T_{i,5}\right)\hat{e}\left(F_{i,2}, T_{i,6}\right))^{\omega_i}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i \in I} w_i M_i = (1,0, \ldots,0)$.

17. The computer server according to claim 12 wherein the public parameters received from the trapdoor server is defined by:

$$public\ parameters = (\hat{g}, u, h, \omega, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, \hat{e}(g, \hat{g})^{\alpha})$$

where $\hat{G}$ is a group of prime order p with the generator $\hat{g}$, $\hat{e}$: $G \times \hat{G} \rightarrow G_1$ is an asymmetric bilinear pairing function, where u, h, ω are random group elements in G, and $\hat{g}_1 = \hat{g}^{d_1}$, $\hat{g}_2 = \hat{g}^{d_2}$, $\hat{g}_3 = \hat{g}^{d_3}$, $\hat{g}_4 = \hat{g}^{d_4}$ where $\alpha, d_1, d_2, d_3, d_4 \in Z_p{}^*$ where $Z_p{}^*$ denotes the set of all non-zero residuals of modular p, and wherein the public key g^{γ} is replaced by $\hat{g}^{\gamma}$.

18. The computer server according to claim 17 wherein the received trapdoor $T_{M,\rho}$ associated with the expressive keyword access structure is defined by the following equation:

$$T_{M,\rho} = \left((M, \rho), T, \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i \in [1,l]}\right)$$

whereby the trapdoor parameters are defined as:

$$T = \hat{g}^{r}, T_{i,1} = g^{v_i}\omega^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}, T_{i,2} = \hat{g}^{\gamma r} \cdot \hat{g}^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$$

$$T_{i,3} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_2}, T_{i,4} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,1}}\right)^{-d_1}, T_{i,5} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_4},$$

$$T_{i,6} = \left(\left(u^{W_{\rho(i)}} h\right)^{t_{i,2}}\right)^{-d_3}, v_i = M_i \cdot \vec{y}$$

where $i \in \{1, \ldots, l\}$ whereby l is the total number of rows in LSSS matrix M, v_i is the value associated with a row M_i of the LSSS matrix M, M_i is the i-th row of LSSS matrix M, $\vec{y} = (\alpha, y_2, \ldots, y_n)^{\perp}$ where $y_2, \ldots, y_n, r, r_0, t_{1,1}, t_{1,2}, \ldots, t_{l,1}, t_{l,2} \in Z_p$ where Z_p are non-zero residuals of a modular positive integer ***p***.

19. The computer server according to claim 18 wherein a ciphertext CT pre-tagged to a document in the encrypted database is defined by the following equation:

ciphertext $CT = (C, D, \{D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2}\}_{i \in [1,m]})$

whereby m is a size of the keyword set of the document, and a first parameter C of the ciphertext CT is defined as:

$$C = \hat{e}(g, \hat{g})^{\alpha\mu}$$

where, $D = \hat{g}^{\mu}, D_i = \omega^{-\mu}(u^{W_i} h)^{z_i}$,

$$E_{i,1} = \hat{g}_1^{\,z_i - s_{i,1}}, E_{i,2} = \hat{g}_2^{s_{i,1}}, F_{i,1} = \hat{g}_3^{z_i - s_{i,2}}, F_{i,2} = \hat{g}_4^{s_{i,2}}$$

where W_i is the *i-th* keyword value in a keyword set of the document, $W_1, \ldots, W_m$ are the corresponding keyword values of the keyword set of the document, and $\mu, s_{1,1}, s_{1,2}, \ldots, s_{m,1}, s_{m,2}, z_1, \ldots, z_m \in Z_p$ where Z_p denotes the set of all residuals of a modular positive integer ***p***.

20. The computer server according to claim 19 wherein the search token for a retrieved document is computed based on the following equation:

search token = $\sum_{i\in I}(\hat{e}(T_{i,1}, D)\hat{e}(D_i, \frac{T_{i,2}}{T^{\gamma}})\hat{e}(T_{i,3}, E_{i,1})\hat{e}(T_{i,4}, E_{i,2})\hat{e}(T_{i,5}, F_{i,1})\hat{e}(T_{i,6}, F_{i,2}))^{\omega_i}$

whereby $I \in I_{M,\rho}$ where $I_{M,\rho}$ is the set of all minimum subsets satisfying (M, ρ) and $\sum_{i\in I} w_i M_i = (1,0,\dots,0)$.

21. The computer server according to claim 12 wherein the public key g^{γ} is g and the private key γ is 1.

22. The computer server according to claim 17 wherein the public key $\hat{g}^{\gamma}$ is $\hat{g}$ and the private key γ is 1.