(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 October 2006 (26.10.2006)

PCT

(10) International Publication Number
WO 2006/112899 A1

(51) International Patent Classification:
G06F 21/00 (2006.01)

(21) International Application Number:
PCT/US2006/000639

(22) International Filing Date: 9 January 2006 (09.01.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/106,181 13 April 2005 (13.04.2005) US

(71) Applicant *(for all designated States except US)*: **ORACLE INTERNATIONAL CORPORATION** [US/US]; 500 Oracle Parkway, Redwood Shores, CA 94065 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **WONG, Daniel Manhung** [US/US]; 7425 Durfee Way, Sacramento, CA 95831 (US). **LEI, Chon Hei** [US/US]; 352 Sweet Road, Alameda, CA 94502 (US).

(74) Agent: **PARK, Richard, A.**; 2820 Fifth Street, Davis, CA 95616 (US).

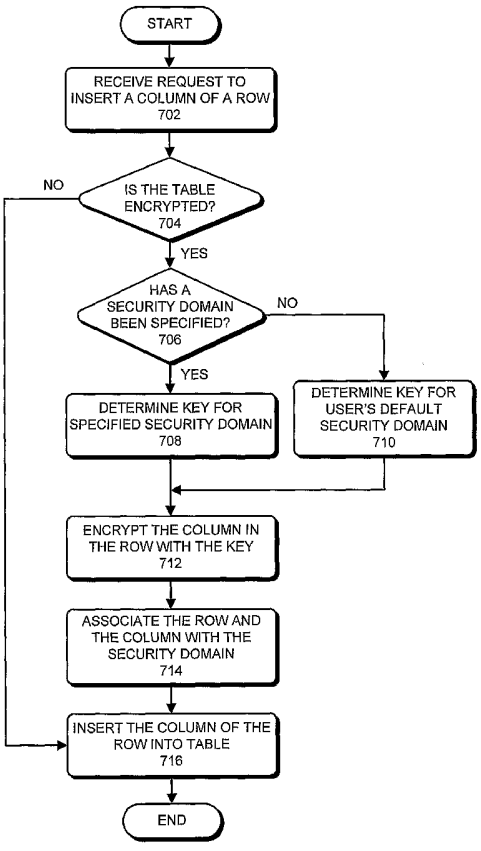
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

[Continued on next page]

(54) **Title:** METHOD AND APPARATUS FOR ENCRYPTING AND DECRYPTING DATA IN A DATABASE TABLE



(57) **Abstract:** One embodiment of the present invention provides a system that decrypts an encrypted column in a row. During operation, the system receives the encrypted column in the row. The system then determines a security domain associated with the encrypted column in the row, wherein the security domain represents a set of columns in rows encrypted using the same key. Next, the system determines a key associated with the security domain. The system then decrypts the encrypted column in the row using the key. Note that using a security domain to represent a set of columns in rows enables the database to grant access to data within the database at arbitrary levels of granularity.

WO 2006/112899 A1

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND APPARATUS FOR ENCRYPTING AND DECRYPTING DATA IN A DATABASE TABLE

Inventors: Daniel ManHung Wong and Chon Hei Lei

Field of the Invention

[0001] The present invention relates to database security. More specifically, the present invention relates to a method and an apparatus for encrypting columns in rows of a database table.

BACKGROUND

Related Art

[0002] As computer systems store ever-larger amounts of sensitive data, it is becoming increasingly important to protect this sensitive data from unauthorized accesses. The global costs incurred from such database security breaches can run into billions of dollars annually, and the cost to individual companies can be severe, sometimes catastrophic.

[0003] Database systems often use encryption to protect sensitive data from unauthorized accesses. Current database encryption techniques are suitable for system architectures with three tiers: an application tier, a mid-tier, and a database tier. In these architectures, the mid-tier usually enforces access control, i.e., the mid-tier decides whether a user can access a particular column of a particular row.

[0004] Unfortunately, mid-tiers often have security holes that can cause sensitive data to be compromised. Additionally, this approach for protecting sensitive data does not work in client-server based architectures that usually do not have a mid-tier.

[0005] Furthermore, in hosting environments, a table may be used for storing sensitive data that belongs to different users. In such situations, the database system needs to ensure that a user can only access rows that belong to him/her. Database systems typically use filters to prevent

a user from accessing sensitive data that belongs to other users. Unfortunately, a user may be able to access sensitive data that belongs to another user by evading these filters.

[0006] Additionally, database systems are increasingly being used to store "large objects" (LOBs) in a column of a row. In some situations, these LOBs contain sensitive information that needs to be accessible only to the LOB's owner. Unfortunately, present database systems do not allow data to be encrypted at such small granularities. Furthermore, the DBA can easily access these documents because the DBA typically has access to the encryption keys.

[0007] Furthermore, re-keying – encrypting data using a new key – poses a major performance problem in present database systems. This is because tables often contain millions of rows of data, and re-keying requires decrypting and encrypting a column in all rows.

[0008] Hence, what is needed is a method and an apparatus for database encryption without the above-described drawbacks.

SUMMARY

[0009] One embodiment of the present invention provides a system that decrypts an encrypted column in a row. During operation, the system receives the encrypted column in the row. The system then determines a security domain associated with the encrypted column in the row, wherein the security domain represents a set of columns in rows encrypted using the same key. Next, the system determines a key associated with the security domain. The system then decrypts the encrypted column in the row using the key. Note that using a security domain to represent a set of columns in rows enables the database to grant access to data within the database at arbitrary levels of granularity.

[0010] In a variation on this embodiment, the system determines the key associated with the security domain by: receiving a user-key; identifying an encrypted-key associated with the security domain; decrypting the encrypted-key using the user-key; performing an integrity check on the decrypted encrypted-key; and if the decrypted encrypted-key passes the integrity check, setting the key to be equal to the decrypted encrypted-key.

[0011] In a variation on this embodiment, the system identifies the encrypted-key associated with the security domain by determining an appropriate encrypted-key from a set of encrypted-keys based on the System Change Number (SCN).

[0012] One embodiment of the present invention provides a system that inserts a column of a row into a table. During operation, the system receives a request to insert the column of the row. The system then determines a security domain based on the request, wherein the security domain represents a set of columns in rows encrypted using the same key. Next, the system determines a key associated with the security domain. The system then encrypts the column of the row using the key. Next, the system inserts the encrypted column of the row into the table.

[0013] In a variation on this embodiment, while inserting the encrypted column of the row into the table, the system associates the encrypted column of the row with the security domain, thereby enabling the database to subsequently determine the appropriate key to decrypt the encrypted column of the row.

[0014] In a variation on this embodiment, the system can receive a request to re-key a second security domain using a new-key. The system then identifies an old set of columns in rows associated with the second security domain. Next, the system decrypts the old set of columns in rows. The system then encrypts the old set of decrypted columns in rows using the new-key to create a new set of columns in rows. Next, the system replaces the old set of columns in rows with the new set of columns in rows. Note that using a security domain to represent a set of columns in rows substantially improves re-keying performance because the database only needs to re-key data in the rows associated with a security domain, instead of re-keying all rows in the table.

[0015] In a variation on this embodiment, while replacing the old set of columns in rows, the system encrypts the new-key with the user-key and associates the encrypted new-key with the second security domain.

[0016] In a variation on this embodiment, the system determines the security domain by determining a default security domain based on a user identifier associated with the request to insert the column of the row.

BRIEF DESCRIPTION OF THE FIGURES

[0017] FIG. 1 illustrates a database system in accordance with an embodiment of the present invention.

[0018] FIG. 2 illustrates how a column in a row can be associated with a security domain in accordance with an embodiment of the present invention.

[0019] FIG. 3 presents a flowchart that illustrates a process for decrypting an encrypted column in a row in accordance with an embodiment of the present invention.

[0020] FIG. 4 presents a flowchart that illustrates a process of determining a key associated with the security domain in accordance with an embodiment of the present invention.

[0021] FIG. 5 illustrates how a system can encrypt keys associated with security domains in accordance with an embodiment of the present invention.

[0022] FIG. 6 illustrates a data structure that can be used to associate a security domain with an encrypted key in accordance with an embodiment of the present invention.

[0023] FIG. 7 presents a flowchart that illustrates a process for inserting a column of a row into a table in accordance with an embodiment of the present invention.

[0024] FIG. 8 presents a flowchart that illustrates a process for re-keying a security domain in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION

[0025] The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not limited to the embodiments shown, but is to be accorded the widest scope consistent with the principles and features disclosed herein.

[0026] The data structures and code described in this detailed description are typically stored on a computer-readable storage medium, which may be any device or medium that can store code and/or data for use by a computer system. This includes, but is not limited to, magnetic and optical storage devices such as disk drives, magnetic tape, CDs (compact discs) and DVDs (digital versatile discs or digital video discs), and computer instruction signals embodied in a transmission medium (with or without a carrier wave upon which the signals are modulated). For example, the transmission medium may include a communications network, such as the Internet.

Database System

[0027] FIG. 1 illustrates a database system in accordance with an embodiment of the present invention. This database system includes two users (or clients), namely, legitimate user 106 and malicious user 108. Database system also includes a set of application servers 110, a set of database servers 102, and a database administrator (DBA) 112. Users 106 and 108, application servers 110, and database servers 102 can communicate with one another via network 104.

[0028] Networks 104 can generally include any type of wire or wireless communication channel capable of coupling together computing nodes. This includes, but is not limited to, a local area network, a wide area network, or a combination of networks. In one embodiment of the present invention, network 104 can include the Internet. Network 104 can also be a private network. Note that in some configurations application servers 110 and database servers 102 can be located on the same physical device.

[0029] Database servers 102 can store data using a variety of data storage systems. This includes, but is not limited to, systems based upon magnetic, optical, and magneto-optical storage devices, as well as storage devices based on flash memory and/or battery-backed up memory.

[0030] Database administrator (DBA) 112 is responsible for the operation and maintenance of database servers 102, and typically has the privilege to modify data stored in the storage system.

[0031] Database clients allow a user, such as legitimate user 106, to enter data through a user interface, such as a web browser. This data can be sent to one of the application servers 110 across network 104. The application server then forms a database query using the data supplied by client 106 and forwards this query to the database servers 102 across network 108. Database servers 102 validate the query and, if valid, perform the requested operation.

[0032] Database systems are being used to store ever-larger amounts of sensitive data. As a result, it is becoming increasingly important to protect sensitive data from persons attempting to access or modify data beyond their authority. For example, a database system may want to allow legitimate users, such as user 106, to access data, but prevent malicious users, such as users 108, from accessing or modifying data. Furthermore, a database system also needs to prevent malicious DBAs from accessing private data.

[0033] To solve these security problems, database systems often use encryption. In these systems, the data is encrypted using a key and stored on a storage medium. Note that this technique provides a high level of security because even if a malicious user gains access to the encrypted data, the malicious user will not be able to decrypt the data without the key.

[0034] In present database systems, the key is often stored on a database server. Unfortunately, this can be a serious security problem because a DBA who has access to the key can decrypt private data.

[0035] Note that a database table is logically structured in terms of rows and columns. Specifically, a table can be viewed as a set of rows, wherein each row comprises a set of columns. In other words, "a column in a row" can be viewed as the smallest unit of data managed by the database system. Hence, "a set of columns in rows" represents an arbitrary portion of data stored in the database system.

[0036] Present database systems allow the user to encrypt the whole table or one or more columns in the table. Specifically, present database systems do not allow a user to encrypt different rows of the same table using different keys. In other words, present database systems do not allow users to encrypt an arbitrary portion of data in the database system.

[0037] Note that present approaches for encrypting data are suitable for system architectures with three tiers: an application tier, a mid-tier, and a database tier. In these architectures, the mid-tier usually enforces access control, i.e., the mid-tier decides whether a user can access a particular column of a particular row. Hence, these architectures are safe in as much as the mid-tier can be trusted with preventing a malicious user from accessing sensitive data.

[0038] Unfortunately, mid-tiers often have security holes that can cause sensitive data to be compromised. Additionally, this approach does not work in client-server based architectures that usually do not have a mid-tier.

[0039] Recall that, one of the problems with present database systems is that a DBA usually has access to the keys which allows the DBA to access sensitive data. Note that even if we store the keys in a secure location – thereby preventing a DBA from accessing the keys – we can still run into security problems. For example, in hosting environments, a table may be used for storing sensitive data that belongs to different users. In such situations, the database system needs to ensure that a user can only access rows that belong to him/her. Database systems

typically use filters to prevent a user from accessing sensitive data that belongs to other users. Unfortunately, a user may be able to access sensitive data that belongs to another user by evading these filters.

[0040] Furthermore, note that a column in a row typically stores standard data types, such as, integers, character strings, etc. But, database systems are increasingly being used to store non-standard data types in these data elements. For example, a column in a row can be used to store a "large object" (LOB), such as a document. In some situations, these LOBs store sensitive information that needs to be accessible only to the owner of the large object. Unfortunately, present database systems do not allow a user to encrypt data at such small granularities. Furthermore, the DBA can easily access these documents because the DBA typically has access to the encryption keys.

[0041] Additionally, re-keying poses a major performance problem in present database systems. Recall that present database systems force the user to employ the same key for a column in all of the rows. Hence, re-keying involves decrypting a column in all of the rows, and then encrypting the column in all of the rows using the new key. Since tables often contain millions of rows of data, re-keying can require a substantial amount of computation in present database systems.

Overview

[0042] One embodiment of the present invention comprises a key management system that associates rows with keys. Whenever there is a reference to a row, the key management system provides an appropriate key to perform the database operation. Furthermore, one embodiment includes an extension to the command execution system which performs database operations on encrypted database objects by interacting with the key management system.

[0043] Note that the relationship between rows and keys can be a many to one relationship. This means that each row only maps to one key, but multiple rows can map to the same key to facilitate sharing of keys. Optionally, we can enhance this mapping by including additional fields such as column, user identifier, etc. For example, a column in a row can be

associated with a key. (Note that, in this example, a row can be associated with multiple keys – one for each encrypted column in the row.)

[0044] The extension to the command execution system typically includes a capability to recognize that the rows in the table may have different keys, which allows the system to derive a modified query plan. Specifically, the modified query plan forces a parallel query slave to run for each unique encryption key. For example, if three distinct encryption keys are used for rows in a table, the modified query plan issues three query slaves, each with its own key. Since each query slave performs its operation with the appropriate key, it eliminates any confusion about which key to use or to cache for the entire operation, thereby making the operation transparent to the rest of the engine.

Security Domain

[0045] A security domain represents a set of columns in rows encrypted using the same key. Recall that a database table is logically structured in terms of rows and columns. Specifically, a table can be viewed as a set of rows, wherein each row comprises a set of columns. Hence, "a set of columns in rows" represents an arbitrary set of data elements stored in the database system. Consequently, a security domain represents an arbitrary set of data elements encrypted using the same key in a database.

[0046] FIG. 2 illustrates how a column in a row can be associated with a security domain in accordance with an embodiment of the present invention.

[0047] Data structure 200 associates row 202 and column 204 with security domain 206. Similarly, row 212 and column 214 are associated with security domain 216. Note that data structure 200 can be used to find a security domain based on a given row and column. Similarly, data structure 200 can also be used to find the set of rows and columns that belong to a particular security domain.

[0048] Data structure 250 associates user identifiers 252 and 262 with security domains 254 and 264, respectively. Note that this data structure can be used to identify a "default security domain" associated with a user.

[0049] It will be apparent to one skilled in the art that a security domain can also be associated with other system parameters. Furthermore, various embodiments of these data structures will be apparent to practitioners in the art. For example, data structure 200 can be implemented as a table in a database.

[0050] In general, these data structures help the system to identify an appropriate security domain for a given set of encrypted data elements, which enables the system to identify an appropriate key to decrypt the set of encrypted data elements. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. In particular, the above-described data structures are not intended to limit the present invention.

Process for Decrypting an Encrypted Column in a Row

[0051] FIG. 3 presents a flowchart that illustrates a process for decrypting an encrypted column in a row in accordance with an embodiment of the present invention.

[0052] The process typically begins by receiving an encrypted column in a row (step 302).

[0053] Next, the system determines a security domain associated with the encrypted column in the row (step 304). In another embodiment, the system determines a security domain based on the user, instead of the specific row and column that needs to be decrypted.

[0054] The system then determines a key associated with the security domain (step 306).

[0055] FIG. 4 presents a flowchart that illustrates a process of determining a key associated with the security domain in accordance with an embodiment of the present invention.

[0056] Specifically, the process of determining a key can begin by receiving a user-key (step 402).

[0057] Next, the system identifies an encrypted-key associated with the security domain (step 404).

[0058] Note that, storing encrypted keys in the database can prevent a DBA from decrypting encrypted data because the user-key is usually not accessible to the DBA. As a result, the DBA cannot decrypt the encrypted key which prevents the DBA from decrypting the encrypted data.

[0059] FIG. 5 illustrates how a system can encrypt keys associated with security domains in accordance with an embodiment of the present invention.

[0060] Encrypted key 512 is associated with security domain 502, and can be encrypted using a user-key that belongs to user 522. Encrypted key 514 is associated with security domain 502, and encrypted keys 516 and 518 are associated with security domain 504. Moreover, encrypted key 514 is encrypted using user 524's key, and encrypted keys 516 and 518 are encrypted using user 522's key and user 526's key, respectively.

[0061] Note that a security domain has only one key, but this key can be encrypted using different user keys. Further, the user-key can be a symmetric key or the private key of an asymmetric encryption technique. The system can use asymmetric encryption as follows. The system first encrypts a security domain's key using a user's public key. The system then uses the user's private key to decrypt the encrypted key.

[0062] FIG. 6 illustrates a data structure that can be used to associate a security domain with an encrypted key in accordance with an embodiment of the present invention.

[0063] Data structure 600 associates user identifier 602 and security domain 604 with encrypted key 606, and user identifier 612 and security domain 614 with encrypted key 616. It will be apparent that other parameters can also be used to associate a security domain with an encrypted key. For example, in one embodiment, the system identifies an encrypted key based on three parameters: a user identifier, a security domain, and a System Change Number (SCN), which is a monotonically increasing counter that can be used to determine the order in which transactions are processed by the database. Note that over time, the system may re-key a security domain. As a result, the system may need to identify the appropriate encrypted-key associated with the security domain based on the SCN. In another embodiment, the system can use a timestamp to determine the appropriate encrypted-key associated with the security domain. Note that various embodiments of these data structures will be apparent to practitioners in the art. For example, data structure 600 can be a table in a database.

[0064] Continuing with the flowchart of FIG. 4, the system then decrypts the encrypted key using the user key (step 406).

[0065] Next, the system performs an integrity check on the decrypted encrypted-key (step 408). Note that the integrity check ensures the validity of the user key.

[0066] If the decrypted encrypted-key passes the integrity check, the system sets the key to be equal to the decrypted encrypted-key (step 410).

[0067] On the other hand, if the decrypted encrypted-key does not pass the integrity check, the system can report an error (step 412).

[0068] Continuing with the flowchart of FIG. 3, the system then decrypts the encrypted column in the row using the key (step 308).

[0069] Note that using a security domain to represent a set of columns in rows enables the database to grant access to data within the database at arbitrary levels of granularity.

Process for Inserting a Column of a Row into a Table

[0070] FIG. 7 presents a flowchart that illustrates a process for inserting a column of a row into a table in accordance with an embodiment of the present invention.

[0071] The process typically begins by receiving a request to insert a column of a row into a table (step 702). Note that, in one embodiment, the system may decide to insert a whole row even though it receives a request to insert only a few columns of the row.

[0072] Next, the system determines whether the table is encrypted or not (step 704).

[0073] If the table is not encrypted, the system inserts the column of the row into the table (step 716).

[0074] On the other hand, if the table is encrypted, the system determines whether a security domain is specified in the request (step 706).

[0075] If a security domain is specified, the system determines a key associated with the specified security domain (step 708).

[0076] On the other hand, if a security domain is not specified, the system determines a key associated with the user's default security domain (step 710). For example, the system can use data structure 250 to determine a default security domain associated with the user identifier. (Note that the request to insert a column in a row usually specifies the user identifier of the user who invoked the request.)

[0077] The system then encrypts the column of the row with the key (step 712).

[0078] Next, the system associates the row and column with the security domain (step 714).

[0079] Note that associating the encrypted column of the row with the security domain enables the database to subsequently determine the appropriate key to decrypt the encrypted column of the row. Specifically, the system can use data structure 200 to associate the row and column with the security domain.

[0080] Finally, the system inserts the column of the row into the table (step 716).

Process of Re-Keying a Security Domain

[0081] FIG. 8 presents a flowchart that illustrates a process for re-keying a security domain in accordance with an embodiment of the present invention.

[0082] The process typically begins by receiving a request to re-key a security domain using a new key (step 802).

[0083] Next, the system identifies a set of columns in rows associated with the security domain (step 804).

[0084] The system then decrypts the set of columns in rows (step 806).

[0085] Next, the system encrypts the set of decrypted columns in rows using the new key to create a new set of columns in rows (step 808).

[0086] The system then replaces the set of columns in rows with the new set of columns in rows (step 810).

[0087] Note that the system can encrypt the new-key with a user-key and associate the encrypted new-key with the security domain. This enables the database to subsequently determine the appropriate key to decrypt the encrypted column of the row. Moreover, if the system uses asymmetric encryption, the system can use a locally stored copy of the user's public key to encrypt the new key. On the other hand, if the system uses symmetric encryption, the system may require the user to provide the user key so that the system can encrypt the new key.

[0088] Furthermore, using a security domain to represent a collection of columns in rows substantially improves re-keying performance because the database only needs to re-key data in the rows associated with a security domain, instead of re-keying all rows in the table.

[0089] The foregoing descriptions of embodiments of the present invention have been presented only for purposes of illustration and description. They are not intended to be exhaustive or to limit the present invention to the forms disclosed. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. Additionally, the above disclosure is not intended to limit the present invention. The scope of the present invention is defined by the appended claims.

What Is Claimed Is:

1. A method for decrypting an encrypted column in a row, the method comprising:

receiving the encrypted column in the row;

determining a security domain associated with the encrypted column in the row, wherein the security domain represents a set of columns in rows encrypted using the same key;

determining a key associated with the security domain; and

decrypting the encrypted column in the row using the key;

wherein using a security domain to represent a set of columns in rows enables the database to grant access to data within the database at arbitrary levels of granularity.

2. The method of claim 1, wherein determining the key associated with the security domain involves:

receiving a user-key;

identifying an encrypted-key associated with the security domain;

decrypting the encrypted-key using the user-key;

performing an integrity check on the decrypted encrypted-key; and

if the decrypted encrypted-key passes the integrity check, setting the key to be equal to the decrypted encrypted-key.

3. The method of claim 2, wherein identifying the encrypted-key associated with the security domain involves determining an appropriate encrypted-key from a set of encrypted-keys based on a timestamp.

4. A method to insert a column of a row into a table, the method comprising:

receiving a request to insert the column of the row;

determining a security domain based on the request, wherein the security domain represents a set of columns in rows encrypted using the same key;

determining a key associated with the security domain;

encrypting the column of the row using the key; and

inserting the encrypted column of the row into the table.

5. The method of claim 4, wherein determining the key associated with the security domain involves:

receiving a user-key;

identifying an encrypted-key associated with the security domain;

decrypting the encrypted-key using the user-key;

performing an integrity check on the decrypted encrypted-key; and

if the decrypted encrypted-key passes the integrity check, setting the key to be equal to the decrypted encrypted-key.

6. The method of claim 4, wherein inserting the encrypted column of the row into the table involves associating the encrypted column of the row with the security domain, thereby enabling the database to subsequently determine an appropriate key to decrypt the encrypted column of the row.

7. The method of claim 4, comprising:

receiving a request to re-key a second security domain using a new-key;

identifying an old set of columns in rows associated with the second security domain;

decrypting the old set of columns in rows;

encrypting the old set of decrypted columns in rows using the new-key to create a new set of columns in rows; and

replacing the old set of columns in rows with the new set of columns in rows;

wherein using a security domain to represent a set of columns in rows substantially improves re-keying performance because the database only needs to re-key data in the rows associated with a security domain, instead of re-keying all rows in the table.

8. The method of claim 7, wherein replacing the old set of columns in rows involves encrypting the new-key with the user-key and associating the encrypted new-key with the second security domain.

9. The method of claim 5, wherein identifying the encrypted-key associated with the security domain involves determining an appropriate encrypted-key from a set of encrypted-keys based on a timestamp.

10. The method of claim 4, wherein determining the security domain involves determining a default security domain based on a user identifier associated with the request to insert the column of the row.

11. A computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for decrypting an encrypted column in a row, the method comprising:

receiving the encrypted column in the row;

determining a security domain associated with the encrypted column in the row, wherein the security domain represents a set of columns in rows encrypted using the same key;

determining a key associated with the security domain; and

decrypting the encrypted column in the row using the key;

wherein using a security domain to represent a set of columns in rows enables the database to grant access to data within the database at arbitrary levels of granularity.

12. The computer-readable storage medium of claim 11, wherein determining the key associated with the security domain involves:

receiving a user-key;

identifying an encrypted-key associated with the security domain;

decrypting the encrypted-key using the user-key;

performing an integrity check on the decrypted encrypted-key; and

if the decrypted encrypted-key passes the integrity check, setting the key to be equal to the decrypted encrypted-key.

13. The computer-readable storage medium of claim 12, wherein identifying the encrypted-key associated with the security domain involves determining an appropriate encrypted-key from a set of encrypted-keys based on a timestamp.

14. A computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method to insert a column of a row into a table, the method comprising:

receiving a request to insert the column of the row;

determining a security domain based on the request, wherein the security domain represents a set of columns in rows encrypted using the same key;

determining a key associated with the security domain;

encrypting the column of the row using the key; and

inserting the encrypted column of the row into the table.

15. The computer-readable storage medium of claim 14, wherein determining the key associated with the security domain involves:

receiving a user-key;

identifying an encrypted-key associated with the security domain;

decrypting the encrypted-key using the user-key;

performing an integrity check on the decrypted encrypted-key; and

if the decrypted encrypted-key passes the integrity check, setting the key to be equal to the decrypted encrypted-key.

16. The computer-readable storage medium of claim 14, wherein inserting the encrypted column of the row into the table involves associating the encrypted column of the row with the security domain, thereby enabling the database to subsequently determine an appropriate key to decrypt the encrypted column of the row.

17. The computer-readable storage medium of claim 14, comprising:

receiving a request to re-key a second security domain using a new-key;

identifying an old set of columns in rows associated with the second security domain;

decrypting the old set of columns in rows;

encrypting the old set of decrypted columns in rows using the new-key to create a new set of columns in rows; and

replacing the old set of columns in rows with the new set of columns in rows;

wherein using a security domain to represent a set of columns in rows substantially improves re-keying performance because the database only needs to re-key data in the rows associated with a security domain, instead of re-keying all rows in the table.

18. The computer-readable storage medium of claim 17, wherein replacing the old set of columns in rows involves encrypting the new-key with the user-key and associating the encrypted new-key with the second security domain.

19. The computer-readable storage medium of claim 15, wherein identifying the encrypted-key associated with the security domain involves determining an appropriate encrypted-key from a set of encrypted-keys based on a timestamp.

20. The computer-readable storage medium of claim 14, wherein determining the security domain involves determining a default security domain based on a user identifier associated with the request to insert the column of the row.

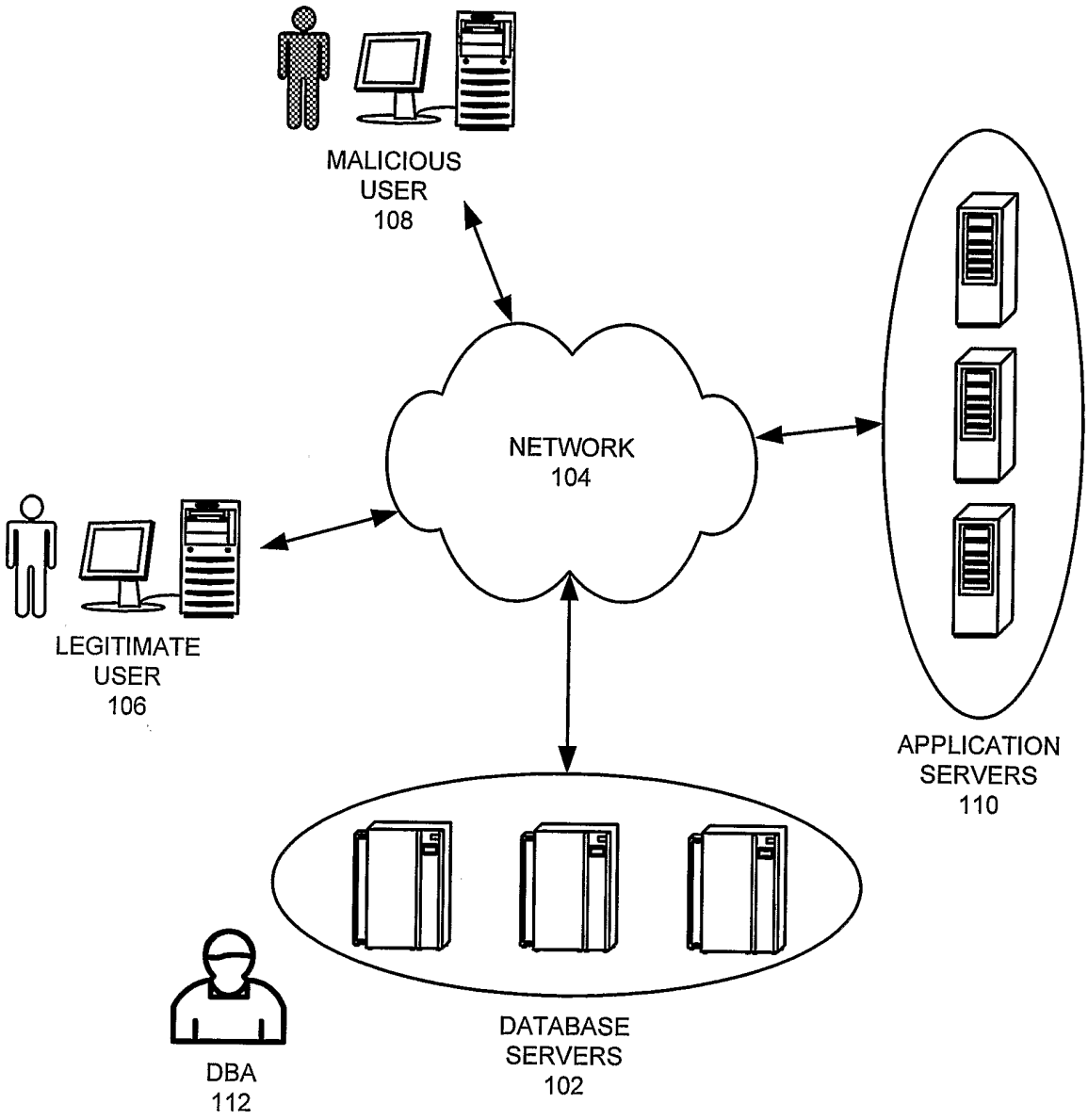

MALICIOUS
USER
108
NETWORK
104
LEGITIMATE
USER
106
APPLICATION
SERVERS
110
DATABASE
SERVERS
102
DBA
112

FIG. 1

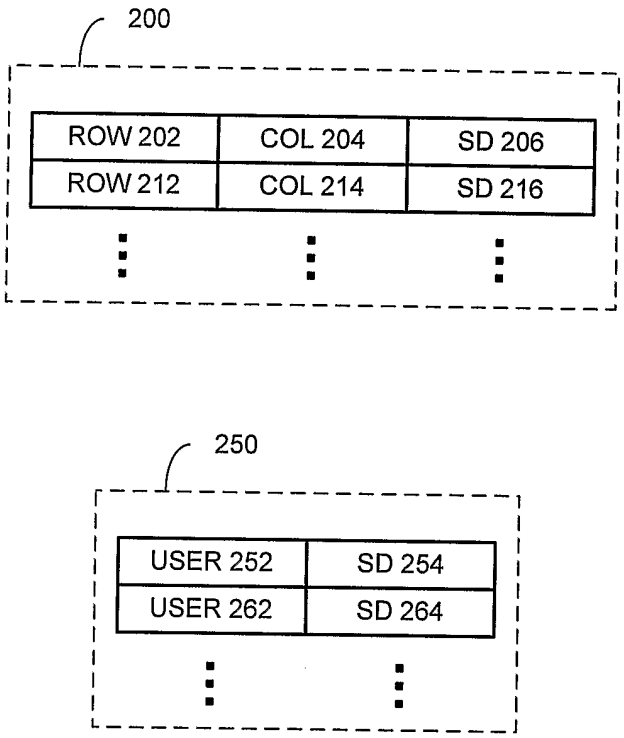

200
ROW 202
COL 204
SD 206
ROW 212
COL 214
SD 216
250
USER 252
SD 254
USER 262
SD 264

FIG. 2

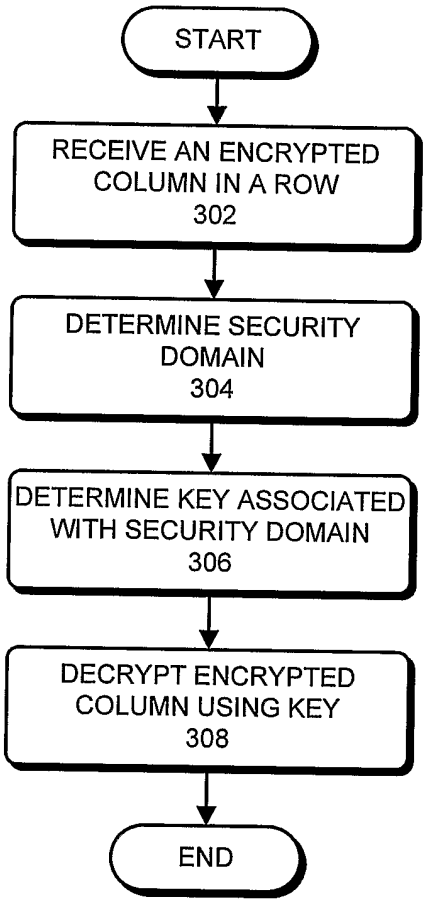

START
RECEIVE AN ENCRYPTED COLUMN IN A ROW
302
DETERMINE SECURITY DOMAIN
304
DETERMINE KEY ASSOCIATED WITH SECURITY DOMAIN
306
DECRYPT ENCRYPTED COLUMN USING KEY
308
END

FIG. 3

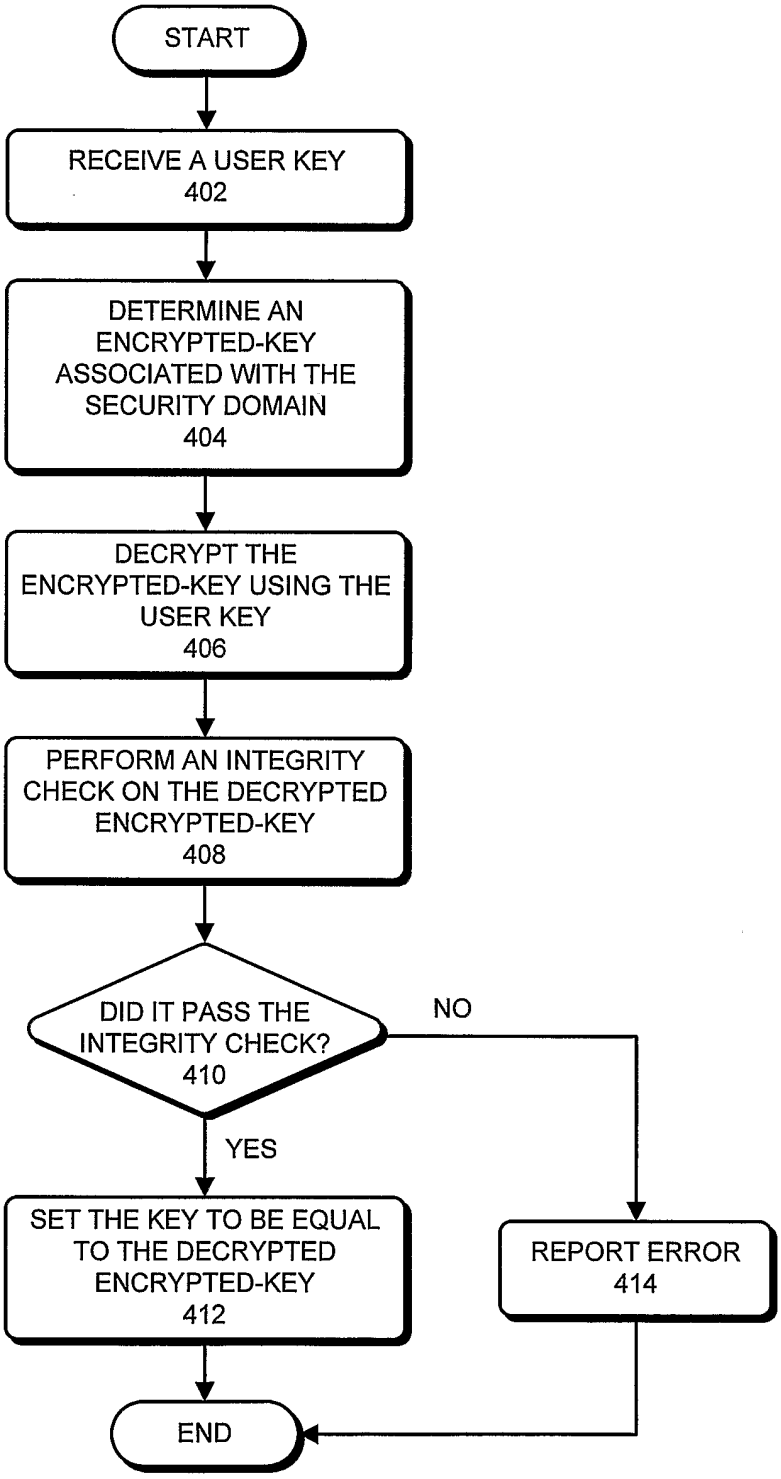

START
RECEIVE A USER KEY
402
DETERMINE AN ENCRYPTED-KEY ASSOCIATED WITH THE SECURITY DOMAIN
404
DECRYPT THE ENCRYPTED-KEY USING THE USER KEY
406
PERFORM AN INTEGRITY CHECK ON THE DECRYPTED ENCRYPTED-KEY
408
DID IT PASS THE INTEGRITY CHECK?
410
NO
YES
SET THE KEY TO BE EQUAL TO THE DECRYPTED ENCRYPTED-KEY
412
REPORT ERROR
414
END

FIG. 4

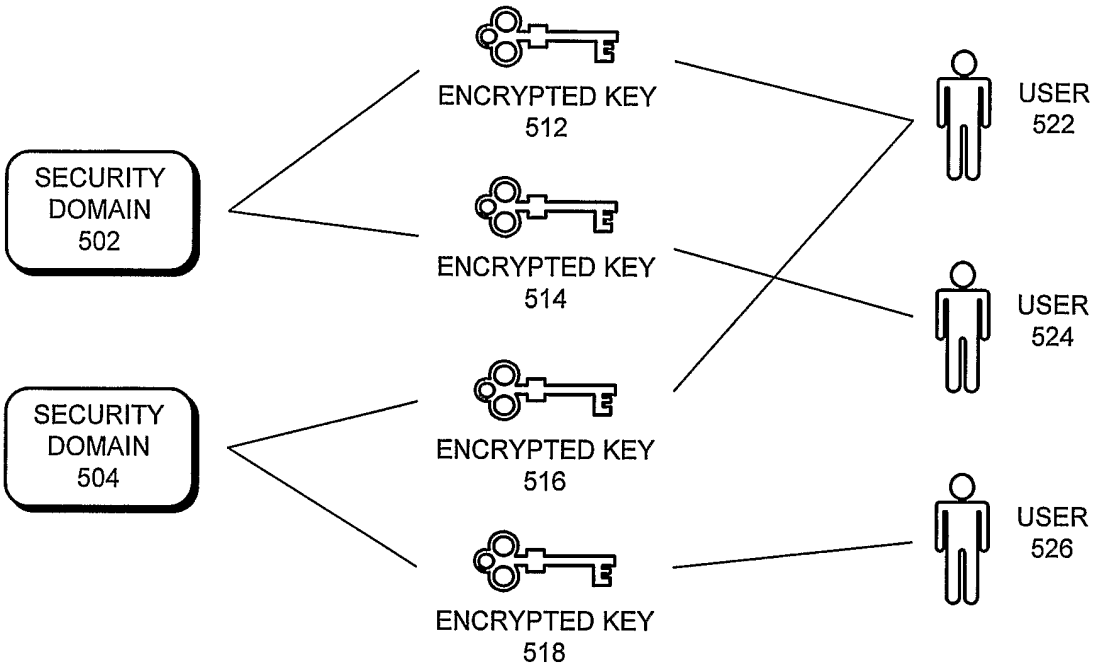

SECURITY DOMAIN 502
SECURITY DOMAIN 504
ENCRYPTED KEY 512
ENCRYPTED KEY 514
ENCRYPTED KEY 516
ENCRYPTED KEY 518
USER 522
USER 524
USER 526

FIG. 5

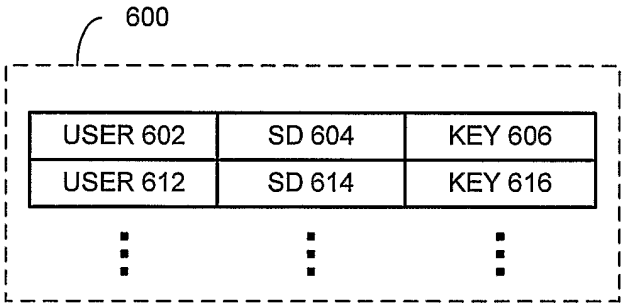

600
USER 602
SD 604
KEY 606
USER 612
SD 614
KEY 616

FIG. 6

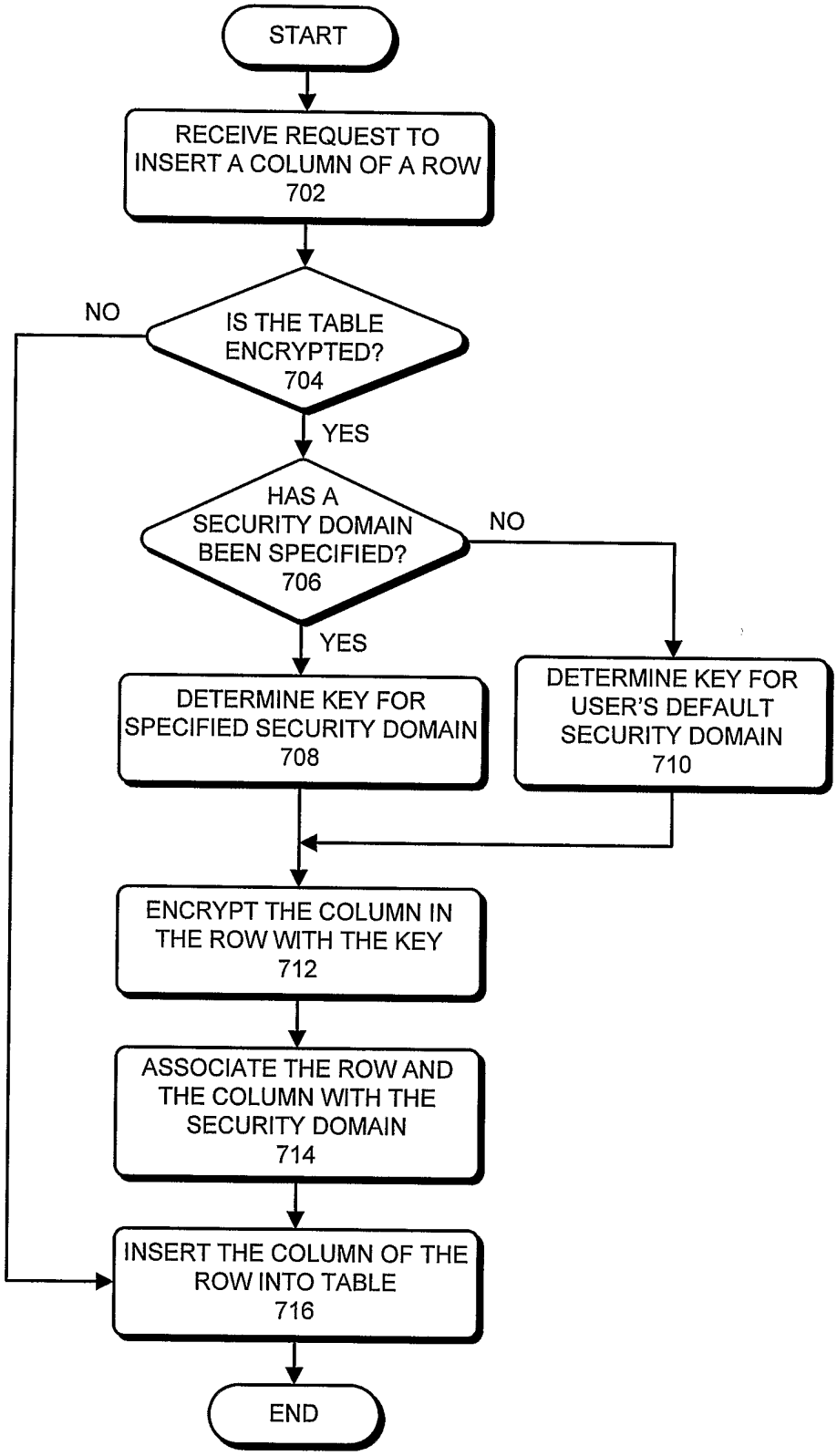

START
RECEIVE REQUEST TO INSERT A COLUMN OF A ROW
702
IS THE TABLE ENCRYPTED?
704
NO
YES
HAS A SECURITY DOMAIN BEEN SPECIFIED?
706
NO
YES
DETERMINE KEY FOR SPECIFIED SECURITY DOMAIN
708
DETERMINE KEY FOR USER'S DEFAULT SECURITY DOMAIN
710
ENCRYPT THE COLUMN IN THE ROW WITH THE KEY
712
ASSOCIATE THE ROW AND THE COLUMN WITH THE SECURITY DOMAIN
714
INSERT THE COLUMN OF THE ROW INTO TABLE
716
END

FIG. 7

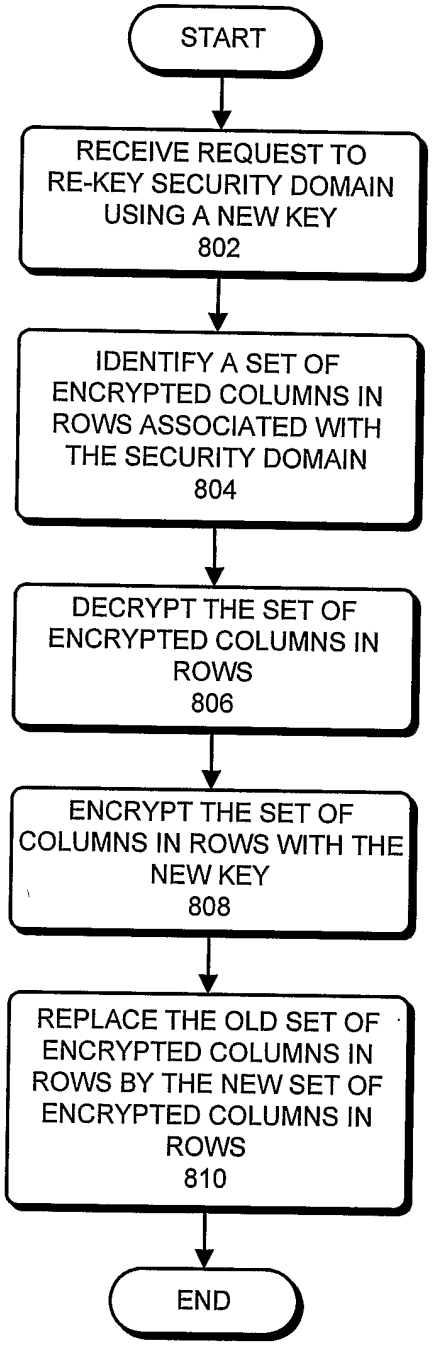

START
RECEIVE REQUEST TO RE-KEY SECURITY DOMAIN USING A NEW KEY
802
IDENTIFY A SET OF ENCRYPTED COLUMNS IN ROWS ASSOCIATED WITH THE SECURITY DOMAIN
804
DECRYPT THE SET OF ENCRYPTED COLUMNS IN ROWS
806
ENCRYPT THE SET OF COLUMNS IN ROWS WITH THE NEW KEY
808
REPLACE THE OLD SET OF ENCRYPTED COLUMNS IN ROWS BY THE NEW SET OF ENCRYPTED COLUMNS IN ROWS
810
END

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2006/000639

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F21/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	IKRAM N ET AL: "A cryptographically secure EW database with selective random access" MILCOM 97 PROCEEDINGS MONTEREY, CA, USA 2-5 NOV. 1997, NEW YORK, NY, USA,IEEE, US, vol. 3, 2 November 1997 (1997-11-02), pages 1407-1411, XP010260677 ISBN: 0-7803-4249-6 section System implementation ----- -/--	1-20

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
12 June 2006	23/06/2006

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Veillas, E

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2006/000639

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WILSON J: "Views as the security objects in a multilevel secure relational database management system" SECURITY AND PRIVACY, 1988. PROCEEDINGS., 1988 IEEE SYMPOSIUM ON OAKLAND, CA, USA 18-21 APRIL 1988, WASHINGTON, DC, USA,IEEE COMPUT. SOC. PR, US, 18 April 1988 (1988-04-18), pages 70-84, XP010012327 ISBN: 0-8186-0850-1 page 73, paragraph 5-7	1-20
Y	JINGRNIN HE ET AL: "Cryptography and relational database management systems" DATABASE ENGINEERING & APPLICATIONS, 2001 INTERNATIONAL SYMPOSIUM ON. JUL. 16-18, 2001, PISCATAWAY, NJ, USA,IEEE, 16 July 2001 (2001-07-16), pages 273-284, XP010554391 ISBN: 0-7695-1140-6 Sections 5.3 and 5.4	2,5,12, 15
Y	WO 01/35226 A (PROTEGRITY RESEARCH & DEVELOPMENT; VALFRIDSSON, THOMAS; MATTSSON, ULF) 17 May 2001 (2001-05-17) see whole document in particular page 4, lines 8-11 and page 4, line 22 - page 5, line 27 the whole document	3,7-9, 17,18
X	US 2001/019614 A1 (MADOUKH ASHRAF) 6 September 2001 (2001-09-06) paragraphs [0005] - [0012], [0017] paragraphs [0052] - [0054], [0059] figures 3,4,13	1-20
P,A	HAKAN HACIGUMU, SHARAD MEHROTA: "Efficient key updates in Encrypted databases" LECTURE NOTES IN COMPUTER SCIENCE, vol. 3674, August 2005 (2005-08), pages 1-15, XP002384598 Heidelberg, Springer Berlin Sections 3.1, 5, 5.1	3,7-9, 17,18

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2006/000639

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 0135226 A	17-05-2001	AU 1322701 A	06-06-2001
		EP 1236117 A1	04-09-2002
US 2001019614 A1	06-09-2001	AU 1320602 A	06-05-2002
		CA 2426419 A1	02-05-2002
		CN 1481525 A	10-03-2004
		EP 1397735 A2	17-03-2004
		TW 523682 B	11-03-2003
		WO 0235329 A2	02-05-2002
		US 2003021417 A1	30-01-2003