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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 January 2018 (25.01.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/017168 A2

(51) International Patent Classification:
H04L 9/08 (2006.01) *H04L 29/06* (2006.01)

(21) International Application Number:
PCT/US2017/027221

(22) International Filing Date:
12 April 2017 (12.04.2017)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

201610252306.5	21 April 2016 (21.04.2016)	CN
201610298847.1	06 May 2016 (06.05.2016)	CN
15/481,338	06 April 2017 (06.04.2017)	US

(71) Applicant: **ALIBABA GROUP HOLDING LIMITED** [—/US]; Fourth Floor, One Capital Place, P. O. Box 847, George Town, Grand Cayman (KY).

(72) Inventors: **YUAN, Peng**; Alibaba Group Legal Department, 5/F, Building 3, No. 969 West Wen Yi Road, Yu Hang District, Hangzhou, Zhejiang 311121 (CN). **WANG, Zhiqiang**; Alibaba Group Legal Department, 5/F, Building 3, No. 969 West Wen Yi Road, Yu Hang District, Hangzhou, Zhejiang 311121 (CN).

(74) Agent: **YAO, Shun**; 2800 Fifth Street, Suite 110, Davis, California 95618 (US).

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

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

(54) Title: SYSTEM AND METHOD FOR ENCRYPTION AND DECRYPTION BASED ON QUANTUM KEY DISTRIBUTION

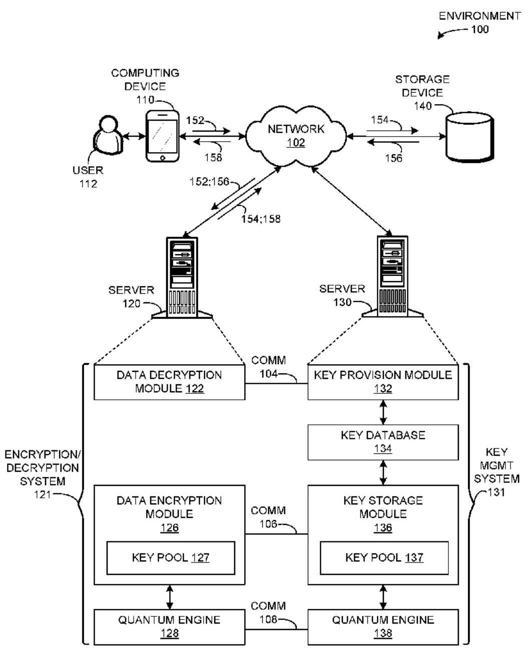

FIG. 1

(57) Abstract: One embodiment of the present invention provides a system for facilitating storage encryption and decryption. During operation, the system receives a first request to encrypt data which is to be stored on a remote device, wherein the first request indicates the data. The system updates a key based on a dynamic key refreshment protocol. The system determines a key label for the updated key. The system encrypts the data based on the updated key, and transmits the encrypted data and the key label to the remote device, thereby facilitating secure encryption and decryption of data on the remote device.

WO 2018/017168 A2

[Continued on next page]

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

SYSTEM AND METHOD FOR ENCRYPTION AND DECRYPTION BASED ON QUANTUM KEY DISTRIBUTION

BACKGROUND

Field

[0001] This disclosure is generally related to the field of data encryption. More specifically, this disclosure is related to a method and system for encryption and decryption based on quantum key distribution.

Related Art

[0002] The proliferation of the Internet and e-commerce continues to create a vast amount and types of digital content. Sensitive data, such as company data and customer data, may be vulnerable to security leaks. One solution to protect such sensitive data is based on data storage encryption technology, which generally involves applying a specific technology to encrypt data before writing the data onto a storage device, and to subsequently decrypt the data when reading the stored data.

[0003] Features of a typical data storage encryption system may include, e.g., host software encryption, an encrypted storage security switch, an embedded special encryption device, and an encryption mechanism based on the storage device itself. A typical logical architecture for such features may include an encryption/decryption unit and a key management center. The encryption/decryption unit can encrypt data to be stored using a key (which is typically fixed or infrequently updated), and can also decrypt stored data with the fixed key. The key management center can store the fixed key, and can also perform other key management-related functions, such as selecting new keys and deleting unused or previously used keys.

[0004] Replacing or updating a fixed (“original”) key may occur only under specific circumstances, e.g., when establishing a secure transmission channel based on an asymmetric encryption algorithm via the key management center and transmitting the new key to the encryption/decryption unit, or by a specific administrative user who physically visits the actual site to manually update the key, e.g., by copying a new key to the system. In addition, when replacing or updating a key, the system must retain access to both the original key and the original encrypted data in order to retain the ability to decrypt the original encrypted data.

[0005] However, this typical data storage encryption system may result in several problems. First, because of the fixed key, the stored encrypted data may be vulnerable to a brute force attack. If the fixed key is discovered, all of the stored data encrypted based on that fixed key may face the risk of exposure. Second, the key distribution process (e.g., selecting or replacing a key) may pose a security risk. For example, adopting an asymmetric encryption algorithm for a key distribution process requires transmissions between two entities. These transmissions may be susceptible to eavesdropping and algorithm hacking. Furthermore, a manual update may face the risk of malicious disclosure. Third, in order for the system to retain access to the original key and the original encrypted data upon updating a key, the system may require administrative users (e.g., management and maintenance personnel) to perform additional and potentially computationally complex work to handle the original key and the original encrypted data. These problems may decrease the efficiency and security of the data storage encryption system.

SUMMARY

[0006] One embodiment of the present invention provides a system for facilitating storage encryption and decryption. During operation, the system receives a first request to encrypt data, wherein the first request indicates the data to be encrypted. The system encrypts the data based on a key, and determines a key label for the key. The system transmits the encrypted data and the key label to a remote device, thereby facilitating secure encryption and decryption of data on the remote device.

[0007] In some embodiments, the system updates the key based on a dynamic key refreshment protocol by performing several operations. The system obtains a first new key from a first key pool. The system transmits a first synchronization request to a key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a second new key which is the same as the first new key. The system obtains a first new key label for the first new key, wherein the first new key label is the same as a second new key label obtained for the second new key, wherein the first new key is the key used to encrypt the data, and wherein the first new key label is the determined key label.

[0008] In some embodiments, the dynamic key refreshment protocol is based on one or more of: determining that a predetermined time interval has passed; and receiving the first request.

[0009] In some embodiments, the first new key and the second new key are obtained based on a same method, and the first new key label and the second new key label are obtained based on a same algorithm.

[0010] In some embodiments, the system computes a first hash value of the first new key. The system includes the first hash value in the first synchronization request, which allows the key-managing device to verify the second new key by confirming that a second hash value of the second new key is the same as the first hash value.

[0011] In some embodiments, a quantum engine of the computer system generates one or more key sequences based on a quantum communication with a quantum engine of a key-managing device. The computer system stores the generated key sequences in a first key pool of the computer system. The key-managing device stores the generated key sequences in a second key pool of the key-managing device. The computer system generates a first key label for a respective key sequence in the first key pool. The key-managing device generates a second key label for a respective key sequence in the second key pool, wherein the first key label and the second key label are generated based on a same algorithm.

[0012] In some embodiments, the system receives a second request to decrypt the encrypted data, wherein the second request indicates the encrypted data and the key label. The system transmits a third request for the updated key, wherein the third request includes the key label. The system receives the updated key, and decrypts the encrypted data based on the received updated key.

[0013] In some embodiments, the third request is transmitted to a key-managing device, and the updated key is received from the key-managing device. Prior to transmitting the third request to the key-managing device, the system obtains obtaining a third key from the first key pool, and transmits a second synchronization request to the key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a fourth key which is the same as the third key. The system encrypts, based on the third key, the key label included in the third request, wherein the received updated key is encrypted based on the fourth key. The system decrypts, based on the third key, the encrypted received updated key.

[0014] Another embodiment of the present invention provides a system for facilitating storage encryption and decryption. During operation, the system receives, based on a dynamic key refreshment protocol, a first synchronization request which indicates a first key obtained from a first key pool. The system obtains a second key from a second key pool, wherein the second key is the same as the first key. The system obtains a second key label for the second key, wherein the second key label is the same as a first key label obtained for the first key. The system stores the second key and the second key label, thereby facilitating secure encryption and decryption of data on a remote device.

[0015] In some embodiments, the first synchronization request includes a first hash value of the first key. The system computes a second hash value of the second key. The system verifies the second key by confirming that the second hash value is the same as the first hash value.

[0016] In some embodiments, the second key and the second key label are stored in a database associated with the computing system. The system receives a request for the second key, wherein the request includes the second key label. The system retrieves the second key from the database based on the second key label included in the request for the second key. The system returns the retrieved second key.

[0017] In some embodiments, the request is received from a security device, and the retrieved second key is returned to the security device. Prior to receiving the request from the security device: the system receives, from the security device, a second synchronization request associated with a third key from the first key pool, wherein the second key label included in the request for the second key is encrypted based on the third key; the system obtains, from the second key pool, a fourth key which is the same as the third key; and the system decrypts the encrypted second key label based on the fourth key. Prior to returning the retrieved second key, the system encrypts the retrieved second key based on the fourth key.

BRIEF DESCRIPTION OF THE FIGURES

[0018] FIG. 1 illustrates an exemplary environment that facilitates storage encryption and decryption, in accordance with an embodiment of the present application.

[0019] FIG. 2 illustrates an exemplary environment that facilitates storage encryption and decryption, including exemplary communications, in accordance with an embodiment of the present application.

[0020] FIG. 3A presents a flowchart illustrating a method by a security device for encrypting data to be stored on a storage device, in accordance with an embodiment of the present application.

[0021] FIG. 3B presents a flowchart illustrating a method by a security device for updating a key, in accordance with an embodiment of the present application.

[0022] FIG. 3C presents a flowchart illustrating a method by a security device for decrypting data stored on a storage device, in accordance with an embodiment of the present application.

[0023] FIG. 4 presents a flowchart illustrating a method for generating keys based on a quantum key distribution protocol, in accordance with an embodiment of the present application.

[0024] FIG. 5A presents a flowchart illustrating a method by a key-managing device for facilitating storage encryption and decryption, in accordance with an embodiment of the present application.

[0025] FIG. 5B presents a flowchart illustrating a method by a key-managing device for retrieving a key based on a corresponding key label, in accordance with an embodiment of the present application.

[0026] FIG. 6 illustrates an exemplary computer system that facilitates storage encryption and decryption, in accordance with an embodiment of the present application.

[0027] FIG. 7 illustrates an exemplary apparatus that facilitates storage encryption and decryption, in accordance with an embodiment of the present application.

[0028] In the figures, like reference numerals refer to the same figure elements.

DETAILED DESCRIPTION

[0029] The following description is presented to enable any person skilled in the art to make and use the embodiments, 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 disclosure. 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.

Overview

[0030] Embodiments of the present invention solve the problem of security-related deficiencies in a data storage encryption system by applying a quantum key distribution protocol to generate keys, and by storing a corresponding key label for a generated key. In a typical data storage encryption system, an encryption/decryption unit (a "security device") and a key management center (a "key-managing device") may share a fixed ("original") key with which to encrypt data to be stored and to decrypt the stored data ("original data"). Replacing or updating such a fixed key may require the system to retain access to both the original key and the original encrypted data. This may result in several deficiencies. First, because of the fixed key, the stored encrypted data may be vulnerable to a brute force attack. If the fixed key is discovered, all of the stored data encrypted based on that fixed key may face the risk of exposure. Second, the key distribution process (e.g., selecting or updating a key) may pose a security risk. For example, adopting an asymmetric encryption algorithm for a key distribution process requires transmissions between two entities. These transmissions may be susceptible to eavesdropping and algorithm hacking. Furthermore, a manual update may face the risk of malicious disclosure. Third, in order for the system to retain access to the original key and the original encrypted data upon updating a key, the system may require administrative users (e.g., management and maintenance personnel) to perform additional and potentially computationally complex work to handle the original key and the original encrypted data. These problems may decrease the efficiency and security of the data storage encryption system

[0031] Embodiments of the present invention address these issues by applying a quantum key distribution technology to generate and distribute keys, selecting a new key periodically or continuously (i.e., updating the key), and storing a corresponding key label for an updated key. Specifically, upon receiving a request to encrypt data to be stored on a storage device, the system updates a key based on a dynamic key refreshment protocol, such as periodically (based on a certain time interval) or continuously (based on a "one storage per key" method). The system determines a key label for the selected key, encrypts the data using the selected key, and returns both the encrypted data and the corresponding key label for storage in the storage device.

[0032] A key-managing device and a security device can generate the same pools of keys (i.e., the same key sequences in each respective key pool) by communicating via a quantum channel or a quantum key distribution protocol. Such a protocol is based on the principle of quantum mechanics, which guarantees the security of the key distribution process, as described herein. Communication via such a protocol further allows the key-managing device and the security device to coordinate and synchronize the selection of the same key and the corresponding key label. Furthermore, by updating the key based on a dynamic key refreshment protocol (i.e., periodically or continuously) instead of using a fixed key, the system may flexibly encrypt data using different keys, which increases the security of the data storage. In addition, the key-managing device and the security device can determine a same corresponding key label for an updated key, which label can be subsequently used to retrieve the updated key and correctly decrypt the stored data. By using a corresponding key label to access each updated key, different keys may be used to encrypt the data to be stored. This allows an increased diversity in the granularity of the key usage, which can both increase the security of the data storage and simplify the process of replacing or updating the key.

[0033] Thus, the present system provides improvements to data storage encryption technology, where the improvements are fundamentally technological. Embodiments of the present invention provide a technological solution (e.g., applying a quantum key distribution technology, updating a key based on a dynamic key refreshment protocol, and storing a corresponding key label for the updated key for subsequent decrypting of the data) to the technological problem of secure data storage encryption (as in the above-described risks in a typical data storage encryption system).

[0034] The terms “quantum key distribution technology,” “quantum key distribution protocol,” and “quantum key distribution operation” refer to operations performed by a quantum engine or a quantum device of a first entity (e.g., a key-managing device) and a second entity (e.g., a security device). The first and second entities each generate the same random key sequence after performing standard processing operations, such as original key negotiation, key screening, error correction, and privacy amplification. Upon generating the same random key sequence, the first and second entities may each store the key sequences in a respective key pool. Exemplary quantum key distribution protocols include the BB84, B91, and B92 protocols, as well as other protocols which have been proposed to improve the code rate, such as continuous variable QKD, DSP-QKP, and SARG. Assume two entities which communicate during a quantum key distribution protocol. The first entity (“Alice”) can randomly generate a set of binary data strings, and can select a basic vector based on the data strings to prepare a corresponding encoded quantum state to be sent to the second entity (“Bob”) via a quantum channel. Subsequently, Bob can disclose his own basic vector via a classical channel (i.e., an electrical or wired channel which is not a quantum channel), which allows both Alice and Bob to select a key using the same basic vector. As a result, both Alice and Bob can obtain the unconditional secure key sequence by estimating the error rate, error correction, and the privacy amplification, and by performing other processing operations.

[0035] The term “key sequence” refers to a sequence consisting of a number of key bits. The term “key pool” refers to a memory area which stores key sequences generated upon performing a quantum key distribution operation.

[0036] The term “key label” refers to an identifier or identifying information used to distinguish between different keys. The key label may be in the form of, e.g., numerical values or character strings.

[0037] The term “synchronization request” refers to a request sent by a security device to a key-managing device, which request triggers the key-managing device to obtain a key from its key pool which is the same as a key from the key pool of the security device.

<u>Exemplary System</u>

[0038] FIG. 1 illustrates an exemplary environment 100 that facilitates storage encryption and decryption, in accordance with an embodiment of the present application. Environment 100 can include a computing device 110 which is associated with a user 112. Computing device 110 can include, for example, a tablet, a mobile phone, an electronic reader, a laptop computer, a desktop computer, or any other computing device. Environment 100 can also include a server 120, a server 130, and a storage device 140, which may communicate with each other (and computing device 110) via a network 102. Server 120 may be, e.g., a security device 120 with an encryption/decryption system 121, which includes: a data decryption module 122 which decrypts data based on a key label; a data encryption module 126 which encrypts data and returns the encrypted data and a corresponding key label; a key pool 127 which may be associated with data encryption module 126; and a quantum engine 128 which generates key sequences based on a quantum key distribution protocol or operation. Server 130 may be, e.g., a key-managing device 130 with a key management system 131, which includes: a key provision module 132 which processes requests for keys by querying a key database 134; key database 134 which stores keys and their corresponding key labels; a key storage module 136 which processes requests for synchronized and updated keys from a key pool generated during a quantum key distribution protocol or operation; and a quantum engine 138 which generates key sequences based on a quantum key distribution protocol or operation.

[0039] Data decryption module 122 can communicate with key provision module 132 via a communication 104, and data encryption module 126 can communicate with key storage module 136 via a communication 106. Quantum engine 128 can communicate with quantum engine 138 via a communication 108, which can be via a quantum channel as part of a quantum key distribution protocol or operation.

[0040] During operation, user 112 may send, via computing device 110, a request 152 to encrypt data which is to be stored on storage device 140. Security device 120 can receive request 152, update a key based on a dynamic key refreshment protocol (e.g., periodically at a certain time interval or continuously for each new request), obtain a key label for the updated key, encrypt the data using the updated key, and send a packet 154 to storage device 140, where packet 154 can indicate the encrypted data and the corresponding key label. These operations are described in detail below in relation to FIG. 2.

[0041] Subsequently, an application or function which can interface with storage device 140 may send a request (not shown) to decrypt the encrypted data stored on storage device 140. This may trigger storage device 140 to send a packet 156 to security device 120, where packet 156 can indicate the encrypted data and the corresponding key label (similar to packet 154). Security device 120 can receive a request to decrypt data based on received packet 156, obtain a key based on the key label included in packet 156, decrypt the data using the obtained key, and send a packet 158 back to the interfacing application or function, e.g., computing device 110 associated with user 112, wherein packet 158 can indicate the decrypted data. These operations are described in detail below in relation to FIG. 2.

[0042] FIG. 2 illustrates an exemplary environment 200 that facilitates storage encryption and decryption, including exemplary communications, in accordance with an embodiment of the present application. Environment 200 can include security device 120, key-managing device 130, and storage device 140, as in environment 100 of FIG. 1. While environment 200 depicts various modules as residing inside each of security device 120 and key-managing device 130, in some embodiments, one or more of these modules may reside within a single system or entity. During operation, quantum engine 128 and quantum engine 138 can perform a quantum key distribution protocol 240, which generates the same key sequences and stores the generated key sequences in, respectively, key pool 127 and key pool 137. Subsequently, data encryption module 126 can receive a request 201 to encrypt data 202 which is to be stored on storage device 140. Request 201 can indicate data of a packet 202. Prior to indicating data 202 in request 201, the system can perform fractionation, compression, and other data processing on data 202. A functional unit may trigger data encryption module 126 to encrypt data 202 via an application programming interface (API) or by transmitting a message (such as request 201).

[0043] Data encryption module 126 can update a key based on a dynamic key refreshment protocol. For example, when the dynamic key refreshment protocol is based on the passage of a certain time period or interval, or based on simply receiving a new request (e.g., request 201), data encryption module may proceed with updating the key by obtaining a new key via a synchronization request, as described below. When the dynamic key refreshment protocol determines that the key is not to be updated (e.g., a certain time interval has not yet passed), the system may use a key which has been most recently used for an encryption operation. In some embodiments, request 201 may include an instruction to update the key, and data encryption module 126 or any other module may process such an included instructions.

[0044] Data encryption module 126 can obtain a key from key pool 127 (function 204), and send a synchronization request 206 to key storage module 136. Synchronization request 206 can indicate that key storage module 136 is to obtain a key from key pool 137 which is the same as the key obtained by data encryption module 126. That is, upon receiving synchronization request 206, key storage module 136 can obtain a key from key pool 137 (function 208), where the obtained key is the same as the key obtained by data encryption module 126 from key pool 127 (as in function 204). Functions 204 and 208 may be based on applying the same predetermined method to obtain the same key from the respective key pool. For example, the key may be obtained by a method which is based on information such as a key selection position and a key length, which may be indicated or carried in synchronization request 206. Any method may be used to obtain the key, as long as the same key is obtained in function 208 as in function 204.

[0045] Key storage module 136 can generate a key label (function 210) for the obtained key, and send both the obtained key and the generated key label to key database 134 (function 212). Data encryption module 126 can also obtain or generate the key label for the obtained key (function 214). Functions 210 and 214 may be based on applying the same predetermined algorithm to generate the key label, or based on a communication between security device 120 and key-managing device 130 regarding the same predetermined algorithm. For example, in applying the same algorithm, a counter may be used to generate the key label for the key. The initial value of the counter may be set to a value of zero, and the counter value may be increased by one for each key replacement. After a key replacement and corresponding increase to the counter value, the counter value may be indicated in the key label. Both security device 120 and key-managing device 130 may use this same algorithm to ensure the generation of the same key label. As another example, the security device may generate the key label, and subsequently send the generated key label to the key-managing device so that the key-managing device has the same generated key label for the key. Alternatively, the key-managing device may generate the key label, and subsequently send the generated key label to the security device so that the security device has the same generated key label for the key.

[0046] Data encryption module 126 may encrypt the data (function 216) based on the obtained key, and send both the encrypted data and the corresponding key label (packet 218) to storage device 140. The system can use a symmetric encryption algorithm, e.g., RC2, RC4, Data Encryption Standard (DES), 3DES, or Advanced Encryption Standard (AES).

[0047] Subsequently, data decryption module 122 can receive a request 221 to decrypt data of a packet 222, which indicates the encrypted data and the corresponding key label, as previously stored on storage device 140. Request 221 may be received from, e.g., a functional unit with an access interface to storage device 140, as described above in relation to request 201. Data decryption module 122 can send a request 224 to key provision module 132. Request 224 can include the key label, and can further indicate a request for the key corresponding to the included key label. Key provision module 132 can receive request 224 and retrieve the requested key from key database 134 based on the corresponding key label (function 226). Recall that the key and the corresponding key label are previously both stored in key database 134. Key provision module 132 can send the retrieved key 228 to data decryption module 122, which can decrypt the data based on the retrieved key (function 230) and transmit the decrypted data (packet 232) back to, e.g., a requesting application.

[0048] Thus, environment 200 depicts a system which facilitates secure encryption and decryption of data on a remote device (e.g., a storage device) by updating a key based on a dynamic key refreshment protocol, storing a key label for the updated key, and applying a quantum key distribution protocol to ensure the synchronization and security of key distribution. The system allows a new key to be selected (e.g., updating a key) on a periodic basis (e.g., based on a predetermined time interval) or on a continuous basis (e.g., based on receiving a request to encrypt data, that is, a "one key one storage" method).

Method for Facilitating Data Storage Encryption by a Security Device

[0049] FIG. 3A presents a flowchart 300 illustrating a method by a security device for encrypting data to be stored on a storage device, in accordance with an embodiment of the present application. During operation, the system receives, by a security device, a first request to encrypt data which is to be stored on a remote device, wherein the first request indicates the data (operation 302). The system updates the key based on a dynamic key refreshment protocol (operation 304). The system determines a key label for the updated key (operation 306). The system encrypts the data based on the updated key (operation 308). The system transmits the encrypted data and the key label to the remote device, thereby facilitating secure encryption and decryption of data on the remote device (operation 310).

[0050] FIG. 3B presents a flowchart 320 illustrating a method by a security device for updating a key, in accordance with an embodiment of the present application. During operation, the system determines whether to update the key based on a dynamic key refreshment protocol (decision 322). If the system determines not to update the key (decision 322), the system selects a key most recently used for an encryption operation (operation 324), and the operation returns. If the system determines to update the key based on a dynamic key refreshment protocol (decision 322), the system obtains a first new key from a first key pool (operation 332). The first key pool may be associated with the security device. The system transmits a first synchronization request to a key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a second new key which is the same as the first new key, wherein the first new key and the second new key are obtained based on a same method (operation 334). The system obtains a first new key label for the first new key, wherein the first new key label is the same as a second new key label obtained for the second new key, wherein the first new key label and the second new key label are obtained based on a same algorithm (operation 336). Upon updating the key, the system may also delete both the key most recently used for an encryption operation, and the corresponding key label. This reduces the amount of storage space required, and also decreases the risk of key exposure by allowing the key-managing device to maintain centralized management.

[0051] In some embodiments, the system may compute a hash of the first new key, and include the computed hash value in the first synchronization request, which allows the key-managing device to verify the second new key by confirming that a second hash value of the second new key is the same as the first hash value. If the first and second hash values do not match, the security device and the key-managing device may re-select the key via another exchange or negotiation.

[0052] FIG. 3C presents a flowchart 340 illustrating a method by a security device for decrypting data stored on a storage device, in accordance with an embodiment of the present application. During operation, the system receives a second request to decrypt encrypted data, wherein the second request indicates the encrypted data and the key label (operation 342). The system transmits a third request for the updated key, wherein the third request includes the key label (operation 344). The system receives the updated key (operation 346), and decrypts the encrypted data based on the received updated key (operation 348).

[0053] To protect the security of the key label included in the third request, the system can encrypt the key label using another key ("third key") and based on a symmetric encryption algorithm (as described above) and included the encrypted key label in the third request. Specifically, the system can obtain this third key from the first key pool (of the system or security device), and transmit a second synchronization request to the key-managing device, which causes the key-managing device to obtain from the second key pool (of the key-managing device) a fourth key which is the same as the third key. The system can then encrypt, based on the third key, the key label included in the third request (as part of operation 344). The key-managing device can decrypt the encrypted key label based on the fourth key, and also encrypt the received updated key based on the fourth key (as part of operation 346). Finally, the system can decrypt the encrypted received updated key based on the third key.

Generating Keys Based on a Quantum Key Distribution Protocol

[0054] FIG. 4 presents a flowchart 400 illustrating a method for generating keys based on a quantum key distribution protocol, in accordance with an embodiment of the present application. During operation, a quantum engine of a computer system (e.g., a security device) generates one or more key sequences based on a quantum communication with a quantum engine of a key-managing device (operation 402). The two quantum engines may communicate as part of a quantum key distribution protocol, via a quantum channel. The advantages of such a protocol are described above. The computer system stores the generated key sequences in a first key pool of the computer system (operation 404). The key-managing device stores the generated key sequences in a second key pool of the key-managing device (operation 406). The computer system generates a first key label for a respective key sequence in the first key pool based a predetermined algorithm (operation 408). The key-managing device generates a second key label for a respective key sequence in the second key pool based on the (same) predetermined algorithm (operation 410).

Method for Facilitating Data Storage Encryption by a Key-Managing Device

[0055] FIG. 5A presents a flowchart 500 illustrating a method by a key-managing device for facilitating storage encryption and decryption, in accordance with an embodiment of the present application. During operation, the system receives, based on a dynamic key refreshment protocol, a first synchronization request which indicates a first key obtained from a first key pool (operation 502). The system obtains a second key from a second key pool, wherein the second key is the same as the first key, wherein the first key and the second key are obtained based on a same method (operation 504). The system obtains a second key label for the second key, wherein the second key label is the same as a first key label obtained for the first key, wherein the first key label and the second key label are obtained based on a same algorithm (operation 506). The system stores the second key and the second key label, thereby facilitating secure encryption and decryption of data on a remote device (operation 508).

[0056] FIG. 5B presents a flowchart 520 illustrating a method by a key-managing device for retrieving a key based on a corresponding key label, in accordance with an embodiment of the present application. During operation, the system receives a request for the second key, wherein the request includes the second key label (operation 522). The system retrieves the second key from a database associated with the system based on the second key label included in the request for the second key (operation 524). The system returns the retrieved second key (operation 526).

Exemplary Computer System and Device

[0057] FIG. 6 illustrates an exemplary computer system 600 that facilitates storage encryption and decryption, in accordance with an embodiment of the present application. Computer system 600 includes a processor 602, a memory 604, a quantum engine 606, and a storage device 608. Memory 604 can include a volatile memory (e.g., RAM) that serves as a managed memory, and can be used to store one or more memory pools. Furthermore, computer system 600 can be coupled to a display device 610, a keyboard 612, and a pointing device 614. Quantum engine 606 can generate quantum keys and communicate via a quantum or photonic communication with another system by distributing and maintaining a same pool of keys.

Storage device 608 can store an operating system 616, a content-processing system 618, and data 632.

[0058] Content-processing system 618 can include instructions, which when executed by computer system 600, can cause computer system 600 to perform methods and/or processes described in this disclosure. Specifically, content-processing system 618 can include: a communication module 620 for transmitting and receiving keys or requests which indicate data, keys, or key labels, including via a quantum or photonic communication; a key-selecting module 622 for updating a key based on a dynamic key refreshment protocol and for selected a most recently used key for an encryption operation; a key label-managing module 624 for generating key labels and for determining a key label for a key; an encrypting/decrypting module 626 for encrypting or decrypting data based on a key; a key-generating module 628 for generating key sequences or keys to be stored in a key pool; and a verification module 630 for verifying a key whose hash value is included in a request.

[0059] Data 632 can include any data that is required as input or that is generated as output by the methods and/or processes described in this disclosure. Specifically, data 632 can store at least: data to be encrypted and stored; a request to encrypt data to be stored on a storage device; a key; an updated key; a dynamic key refreshment protocol; a key label for a key; a key pool; a key sequence; a synchronization request; a predetermined time interval; an indicator of receiving a request; a method used to obtain a key from a key pool; an algorithm used to obtain or determine a key label for a key; a hash value; a hash value of a key; a request to decrypt encrypted data; a request for a key, where the request include a key label for the requested key; a database; encrypted data; decrypted data; and an encrypted or decrypted key or key label.

[0060] FIG. 7 illustrates an exemplary apparatus 700 that facilitates storage encryption and decryption, in accordance with an embodiment of the present application. Apparatus 700 can comprise a plurality of units or apparatuses which may communicate with one another via a wired, wireless, quantum light, or electrical communication channel. Device 700 may be realized using one or more integrated circuits, and may include fewer or more units or apparatuses than those shown in FIG. 7. Further, device 700 may be integrated in a computer system, or realized as a separate device which is capable of communicating with other computer systems and/or devices. Specifically, device 700 can comprise units 702-712 which perform functions or operations similar to modules 620-630 of computer system 600 of FIG. 6, including: a communication unit 702; a key-selecting unit 704; a key label-managing unit 706; an encrypting/decrypting unit 708; a key-generating unit 710; and a verification unit 712.

[0061] 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. The computer-readable storage medium includes, but is not limited to, volatile memory, non-volatile memory, magnetic and optical storage devices such as disk drives, magnetic tape, CDs (compact discs), DVDs (digital versatile discs or digital video discs), or other media capable of storing computer-readable media now known or later developed.

[0062] The methods and processes described in the detailed description section can be embodied as code and/or data, which can be stored in a computer-readable storage medium as described above. When a computer system reads and executes the code and/or data stored on the computer-readable storage medium, the computer system performs the methods and processes embodied as data structures and code and stored within the computer-readable storage medium.

[0063] Furthermore, the methods and processes described above can be included in hardware modules. For example, the hardware modules can include, but are not limited to, application-specific integrated circuit (ASIC) chips, field-programmable gate arrays (FPGAs), and other programmable-logic devices now known or later developed. When the hardware modules are activated, the hardware modules perform the methods and processes included within the hardware modules.

[0064] The foregoing descriptions of embodiments of the present invention have been presented for purposes of illustration and description only. 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 computer system for facilitating storage encryption and decryption, the system comprising:

a processor; and

a memory coupled to the processor and storing instructions, which when executed by the processor cause the processor to perform a method, the method comprising:

receiving a first request to encrypt data, wherein the first request indicates the data to be encrypted;

encrypting the data based on a key;

determining a key label for the key; and

transmitting the encrypted data and the key label to a remote device,

thereby facilitating secure encryption and decryption of data on the remote device.

2. The computer system of claim 1, wherein the method further comprises updating the key based on the dynamic key refreshment protocol, which comprises:

obtaining a first new key from a first key pool;

transmitting a first synchronization request to a key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a second new key which is the same as the first new key; and

obtaining a first new key label for the first new key, wherein the first new key label is the same as a second new key label obtained for the second new key,

wherein the first new key is the key used to encrypt the data, and wherein the first new key label is the determined key label.

3. The computer system of claim 2, wherein the dynamic key refreshment protocol is based on one or more of:

determining that a predetermined time interval has passed; and

receiving the first request.

4. The computer system of claim 2, wherein the first new key and the second new key are obtained based on a same method, and wherein the first new key label and the second new key label are obtained based on a same algorithm.

5. The computer system of claim 2, wherein the method further comprises:

computing a first hash value of the first new key;

including the first hash value in the first synchronization request, which allows the key-managing device to verify the second new key by confirming that a second hash value of the second new key is the same as the first hash value.

6. The computer system of claim 1, wherein the method further comprises:

generating, by a quantum engine of the computer system, one or more key sequences based on a quantum communication with a quantum engine of a key-managing device;

storing, by the computer system, the generated key sequences in a first key pool of the computer system;

storing, by the key-managing device, the generated key sequences in a second key pool of the key-managing device;

generating, by the computer system, a first key label for a respective key sequence in the first key pool; and

generating, by the key-managing device, a second key label for a respective key sequence in the second key pool,

wherein the first key label and the second key label are generated based on a same algorithm.

7. A computer system for facilitating data access and data decryption, the system comprising:

a processor; and

a memory coupled to the processor and storing instructions, which when executed by the processor cause the processor to perform a method, the method comprising:

receiving a first request to decrypt encrypted data, wherein the first request indicates the encrypted data and a key label for a first key used to encrypt the data;

transmitting a second request for the first key, wherein the second request includes the key label;

receiving the first key; and

decrypting the encrypted data based on the received first key.

8. The computer system of claim 7, wherein the second request is transmitted to a key-managing device, wherein the first key is received from the key-managing device, wherein the method further comprises:

prior to transmitting the second request to the key-managing device:

obtaining a second key from a first key pool; and

transmitting a synchronization request to the key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a third key which is the same as the second key;

encrypting, based on the second key, the key label included in the second request, wherein the received first key is encrypted based on the third key; and

decrypting, based on the second key, the encrypted received first key.

9. A computer system for facilitating storage encryption and decryption, the system comprising:

a processor; and

a memory coupled to the processor and storing instructions, which when executed by the processor cause the processor to perform a method, the method comprising:

receiving, based on a dynamic key refreshment protocol, a first synchronization request which indicates a first key obtained from a first key pool;

obtaining a second key from a second key pool, wherein the second key is the same as the first key;

obtaining a second key label for the second key, wherein the second key label is the same as a first key label obtained for the first key; and

storing the second key and the second key label,

thereby facilitating secure encryption and decryption of data on a remote device.

10. The computer system of claim 9,

wherein the first key and the second key are obtained based on a same method, and wherein the first key label and the second key label are obtained based on a same algorithm.

11. The computer system of claim 9, wherein the first synchronization request includes a first hash value of the first key, wherein the method further comprises:

computing a second hash value of the second key; and

verifying the second key by confirming that the second hash value is the same as the first hash value.

12. The computer system of claim 9, wherein the second key and the second key label are stored in a database associated with the computing system, wherein the method further

comprises:

receiving a request for the second key, wherein the request includes the second key label;

retrieving the second key from the database based on the second key label included in the request for the second key; and

returning the retrieved second key.

13. The computer system of claim 12, wherein the request is received from a security device, wherein the retrieved second key is returned to the security device, wherein the method further comprises:

prior to receiving the request from the security device:

receiving, from the security device, a second synchronization request associated with a third key from the first key pool, wherein the second key label included in the request for the second key is encrypted based on the third key; and

obtaining, from the second key pool, a fourth key which is the same as the third key;

decrypting the encrypted second key label based on the fourth key; and

prior to returning the retrieved second key, encrypting the retrieved second key based on the fourth key.

14. A computer-implemented method for facilitating storage encryption and decryption, the method comprising:

receiving a first request to encrypt data, wherein the first request indicates the data to be encrypted;

encrypting the data based on a key;

determining a key label for the key; and

transmitting the encrypted data and the key label to a remote device,

thereby facilitating secure encryption and decryption of data on the remote device.

15. The method of claim 14, further comprising updating the key based on the dynamic key refreshment protocol, which comprises:

obtaining a first new key from a first key pool;

transmitting a first synchronization request to a key-managing device, which causes the key-managing device to obtain from a second key pool of the key-managing device a second new key which is the same as the first new key; and

obtaining a first new key label for the first new key, wherein the first new key label is the same as a second new key label obtained for the second new key,

wherein the first new key is the key used to encrypt the data, and wherein the first new key label is the determined key label.

16. The method of claim 15, wherein the dynamic key refreshment protocol is based on one or more of:

determining that a predetermined time interval has passed; and

receiving the first request.

17. The method of claim 15, wherein the first new key and the second new key are obtained based on a same method, and wherein the first new key label and the second new key label are obtained based on a same algorithm.

18. The method of claim 15, further comprising:

computing a first hash value of the first new key;

including the first hash value in the first synchronization request, which allows the key-managing device to verify the second new key by confirming that a second hash value of the second new key is the same as the first hash value.

19. The method of claim 14, further comprising:

generating, by a quantum engine of the computer system, one or more key sequences based on a quantum communication with a quantum engine of a key-managing device;

storing, by the computer system, the generated key sequences in a first key pool of the computer system;

storing, by the key-managing device, the generated key sequences in a second key pool of the key-managing device;

generating, by the computer system, a first key label for a respective key sequence in the first key pool; and

generating, by the key-managing device, a second key label for a respective key sequence in the second key pool,

wherein the first key label and the second key label are generated based on a same algorithm.

20. A computer-implemented method for facilitating data access and data decryption, the method comprising:

receiving a first request to decrypt encrypted data, wherein the first request indicates the encrypted data and a key label for a first key used to encrypt the data;

transmitting a second request for the first key, wherein the second request includes the key label;

receiving the first key; and

decrypting the encrypted data based on the received first key.

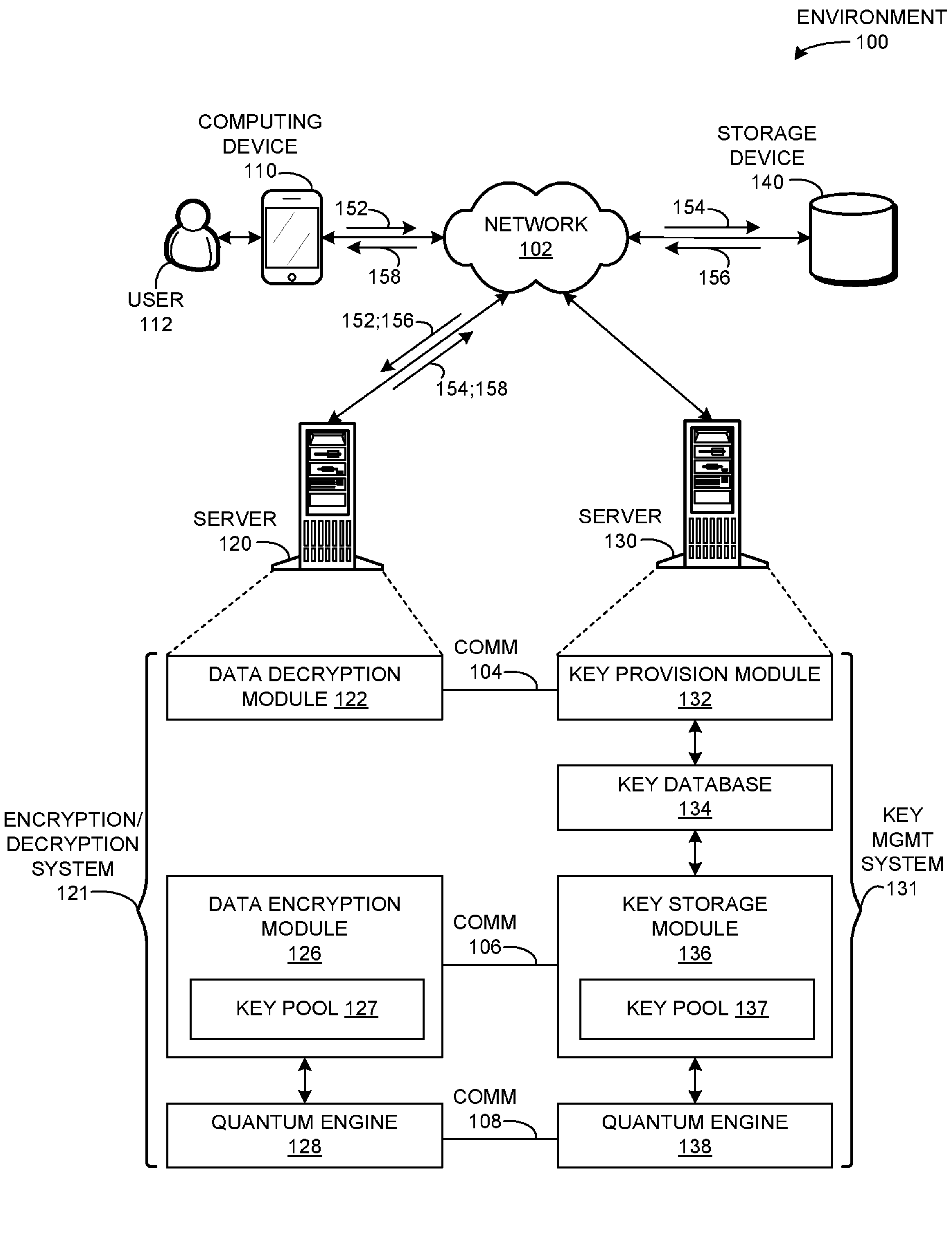
ENVIRONMENT
100
COMPUTING DEVICE
110
USER
112
152
158
NETWORK
102
STORAGE DEVICE
140
154
156
152;156
154;158
SERVER
120
SERVER
130
DATA DECRYPTION MODULE 122
COMM
104
KEY PROVISION MODULE
132
KEY DATABASE
134
ENCRYPTION/ DECRYPTION SYSTEM
121
KEY MGMT SYSTEM
131
DATA ENCRYPTION MODULE
126
KEY POOL 127
COMM
106
KEY STORAGE MODULE
136
KEY POOL 137
QUANTUM ENGINE
128
COMM
108
QUANTUM ENGINE
138

FIG. 1

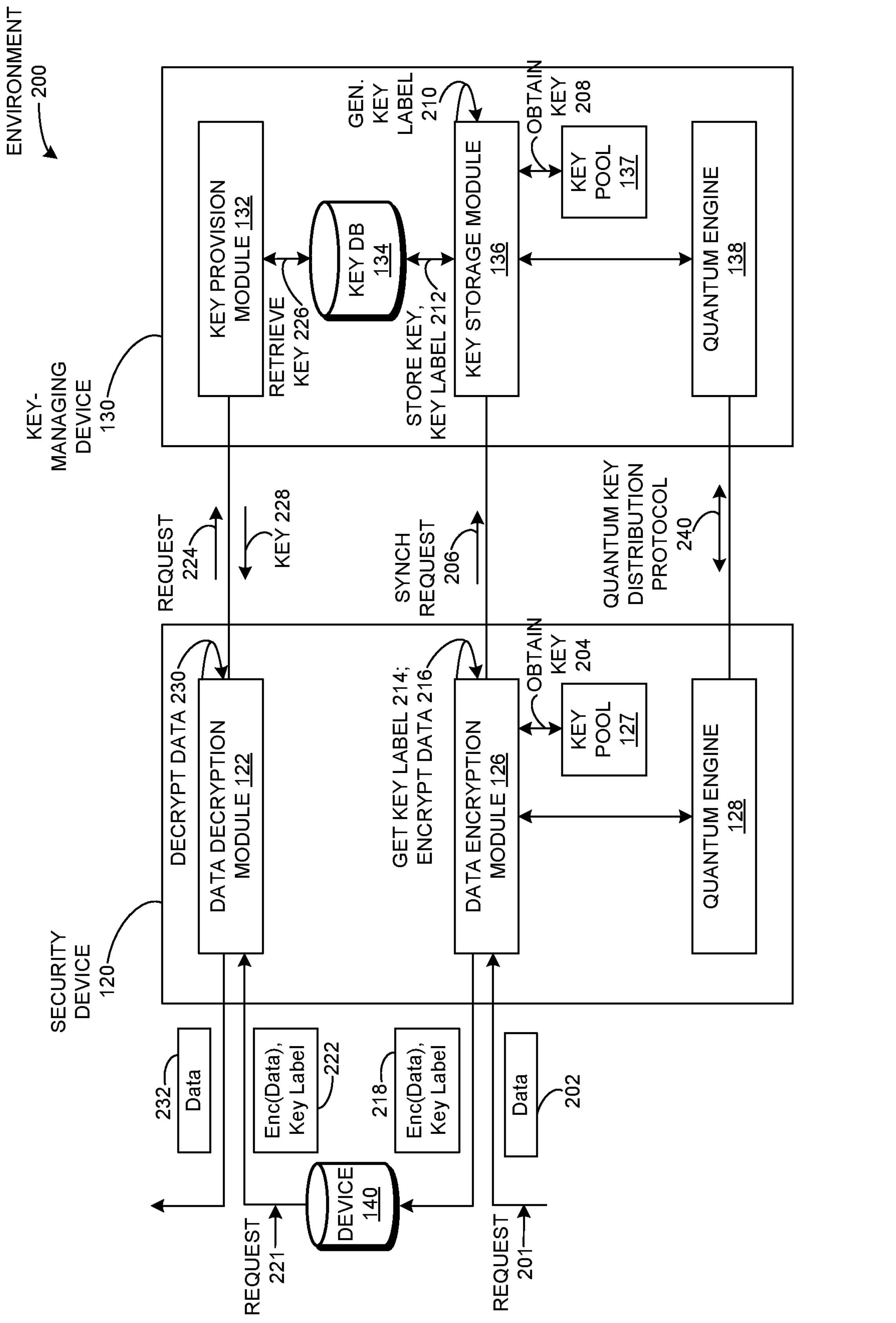
ENVIRONMENT 200
KEY-MANAGING DEVICE 130
KEY PROVISION MODULE 132
RETRIEVE KEY 226
KEY DB 134
STORE KEY, KEY LABEL 212
GEN. KEY LABEL 210
KEY STORAGE MODULE 136
OBTAIN KEY 208
KEY POOL 137
QUANTUM ENGINE 138
REQUEST 224
KEY 228
SYNCH REQUEST 206
QUANTUM KEY DISTRIBUTION PROTOCOL 240
SECURITY DEVICE 120
DECRYPT DATA 230
DATA DECRYPTION MODULE 122
GET KEY LABEL 214; ENCRYPT DATA 216
DATA ENCRYPTION MODULE 126
OBTAIN KEY 204
KEY POOL 127
QUANTUM ENGINE 128
Data 232
Enc(Data), Key Label 222
Enc(Data), Key Label 218
Data 202
DEVICE 140
REQUEST 221
REQUEST 201

FIG. 2

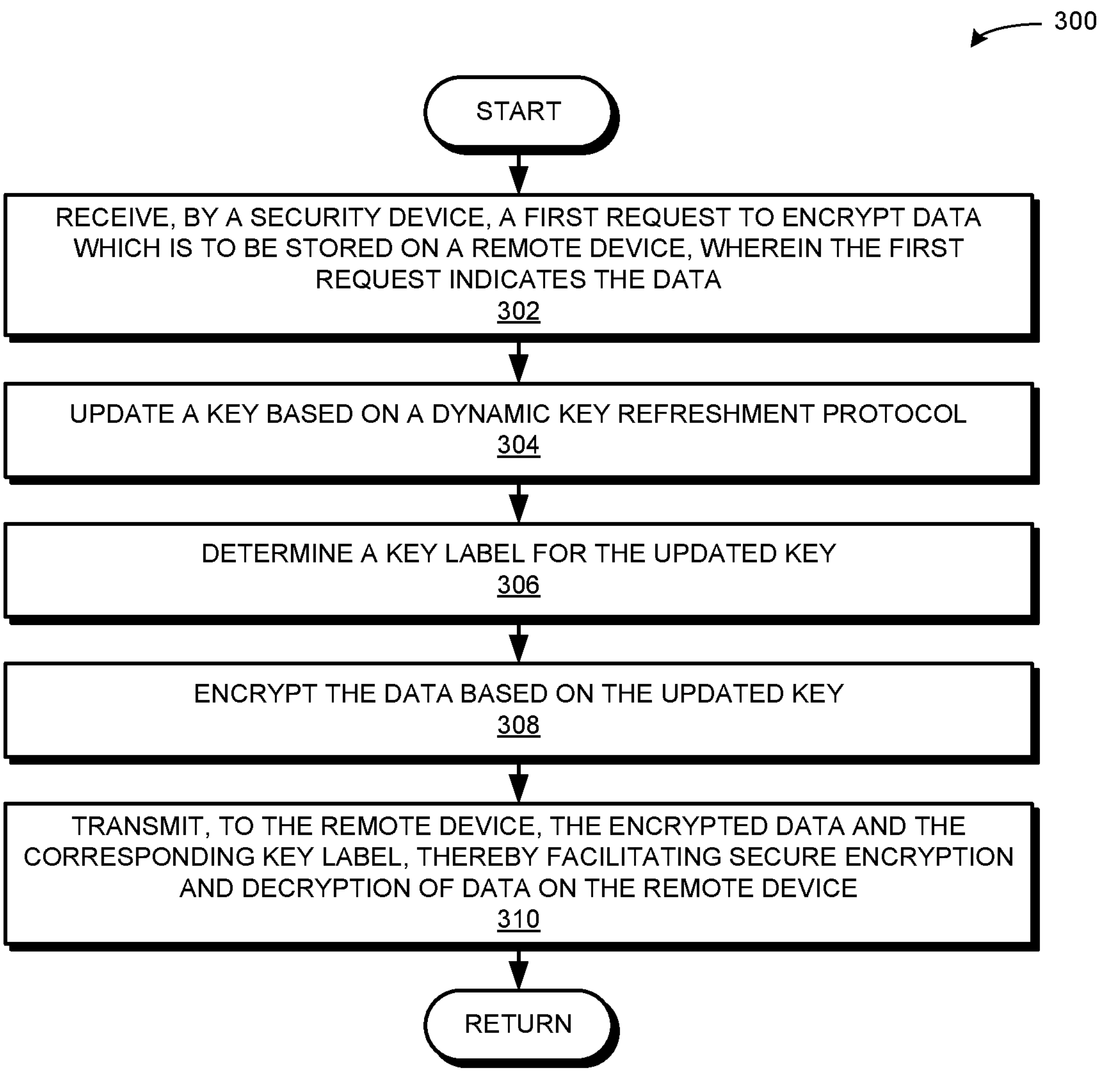
300
START
RECEIVE, BY A SECURITY DEVICE, A FIRST REQUEST TO ENCRYPT DATA WHICH IS TO BE STORED ON A REMOTE DEVICE, WHEREIN THE FIRST REQUEST INDICATES THE DATA
302
UPDATE A KEY BASED ON A DYNAMIC KEY REFRESHMENT PROTOCOL
304
DETERMINE A KEY LABEL FOR THE UPDATED KEY
306
ENCRYPT THE DATA BASED ON THE UPDATED KEY
308
TRANSMIT, TO THE REMOTE DEVICE, THE ENCRYPTED DATA AND THE CORRESPONDING KEY LABEL, THEREBY FACILITATING SECURE ENCRYPTION AND DECRYPTION OF DATA ON THE REMOTE DEVICE
310
RETURN

FIG. 3A

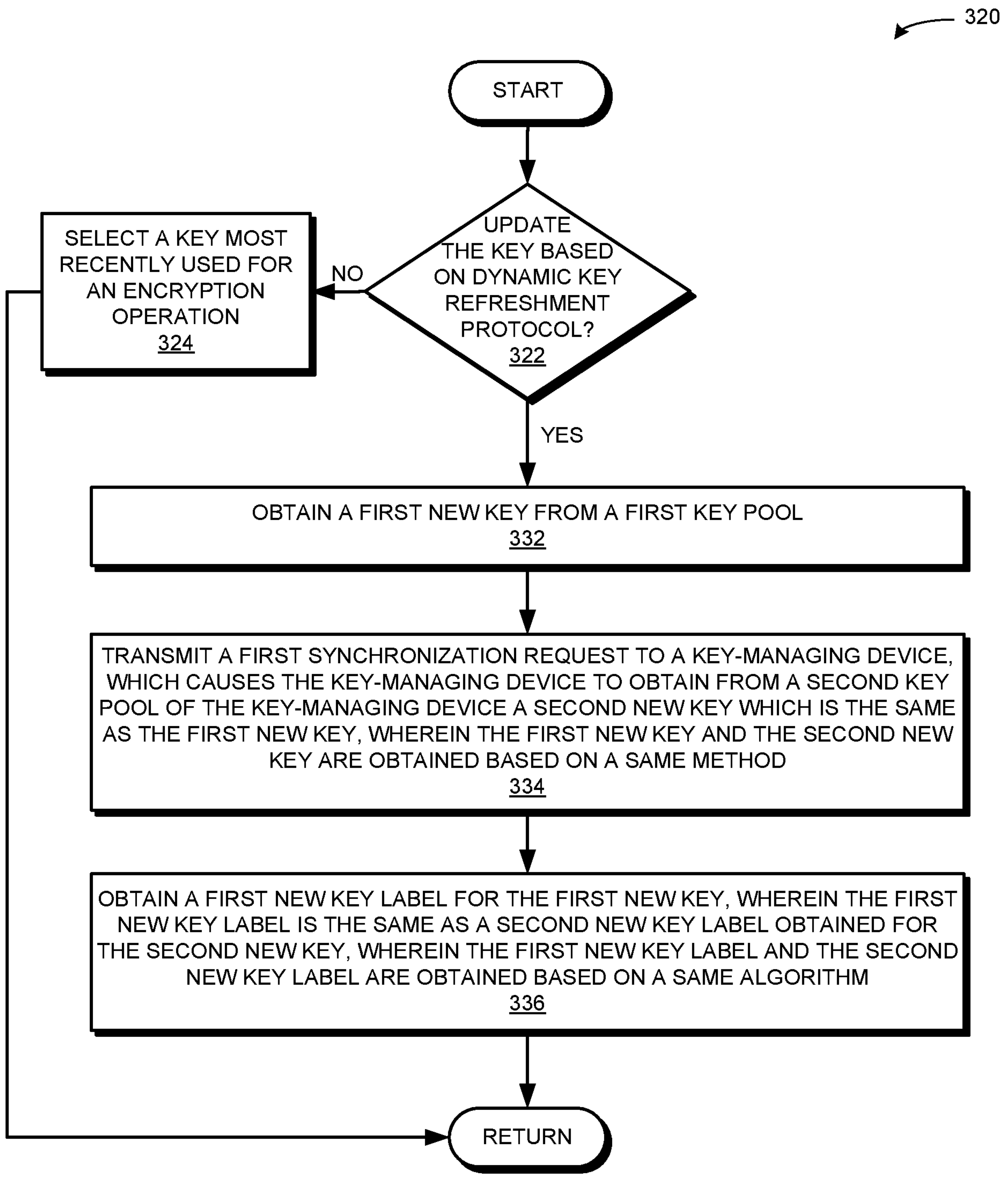
320
START
UPDATE THE KEY BASED ON DYNAMIC KEY REFRESHMENT PROTOCOL? 322
NO
SELECT A KEY MOST RECENTLY USED FOR AN ENCRYPTION OPERATION 324
YES
OBTAIN A FIRST NEW KEY FROM A FIRST KEY POOL 332
TRANSMIT A FIRST SYNCHRONIZATION REQUEST TO A KEY-MANAGING DEVICE, WHICH CAUSES THE KEY-MANAGING DEVICE TO OBTAIN FROM A SECOND KEY POOL OF THE KEY-MANAGING DEVICE A SECOND NEW KEY WHICH IS THE SAME AS THE FIRST NEW KEY, WHEREIN THE FIRST NEW KEY AND THE SECOND NEW KEY ARE OBTAINED BASED ON A SAME METHOD 334
OBTAIN A FIRST NEW KEY LABEL FOR THE FIRST NEW KEY, WHEREIN THE FIRST NEW KEY LABEL IS THE SAME AS A SECOND NEW KEY LABEL OBTAINED FOR THE SECOND NEW KEY, WHEREIN THE FIRST NEW KEY LABEL AND THE SECOND NEW KEY LABEL ARE OBTAINED BASED ON A SAME ALGORITHM 336
RETURN

FIG. 3B

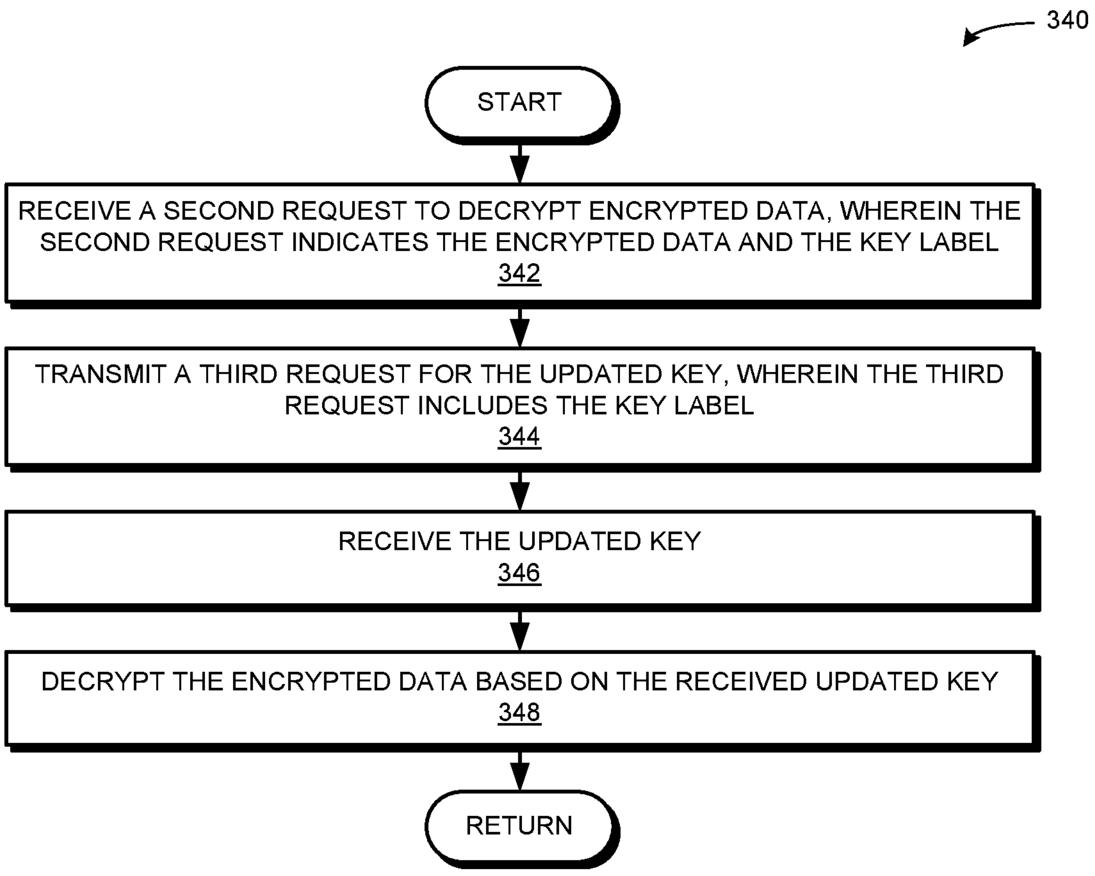
340
START
RECEIVE A SECOND REQUEST TO DECRYPT ENCRYPTED DATA, WHEREIN THE SECOND REQUEST INDICATES THE ENCRYPTED DATA AND THE KEY LABEL 342
TRANSMIT A THIRD REQUEST FOR THE UPDATED KEY, WHEREIN THE THIRD REQUEST INCLUDES THE KEY LABEL 344
RECEIVE THE UPDATED KEY 346
DECRYPT THE ENCRYPTED DATA BASED ON THE RECEIVED UPDATED KEY 348
RETURN

FIG. 3C

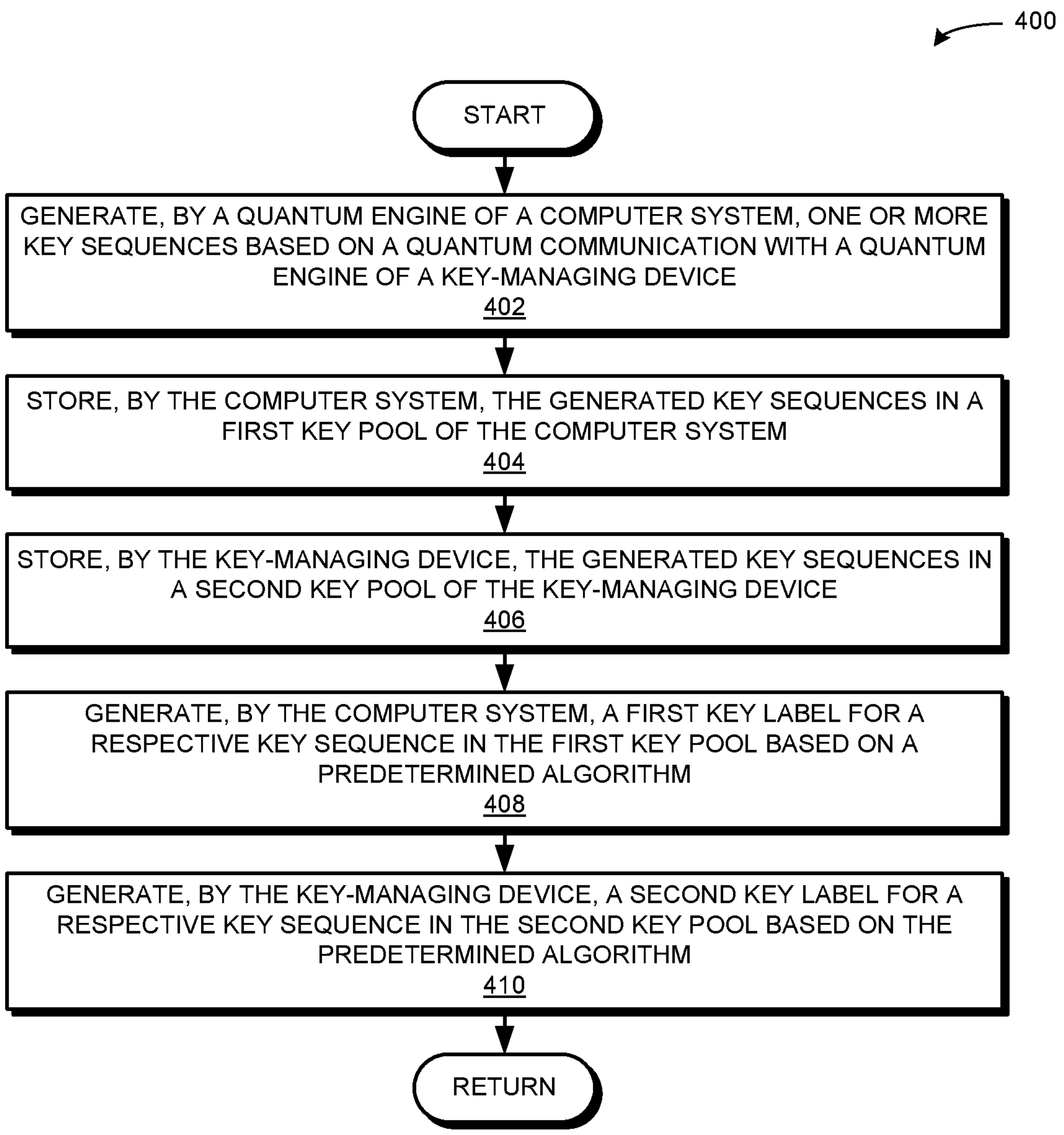
400
START
GENERATE, BY A QUANTUM ENGINE OF A COMPUTER SYSTEM, ONE OR MORE KEY SEQUENCES BASED ON A QUANTUM COMMUNICATION WITH A QUANTUM ENGINE OF A KEY-MANAGING DEVICE
402
STORE, BY THE COMPUTER SYSTEM, THE GENERATED KEY SEQUENCES IN A FIRST KEY POOL OF THE COMPUTER SYSTEM
404
STORE, BY THE KEY-MANAGING DEVICE, THE GENERATED KEY SEQUENCES IN A SECOND KEY POOL OF THE KEY-MANAGING DEVICE
406
GENERATE, BY THE COMPUTER SYSTEM, A FIRST KEY LABEL FOR A RESPECTIVE KEY SEQUENCE IN THE FIRST KEY POOL BASED ON A PREDETERMINED ALGORITHM
408
GENERATE, BY THE KEY-MANAGING DEVICE, A SECOND KEY LABEL FOR A RESPECTIVE KEY SEQUENCE IN THE SECOND KEY POOL BASED ON THE PREDETERMINED ALGORITHM
410
RETURN

FIG. 4

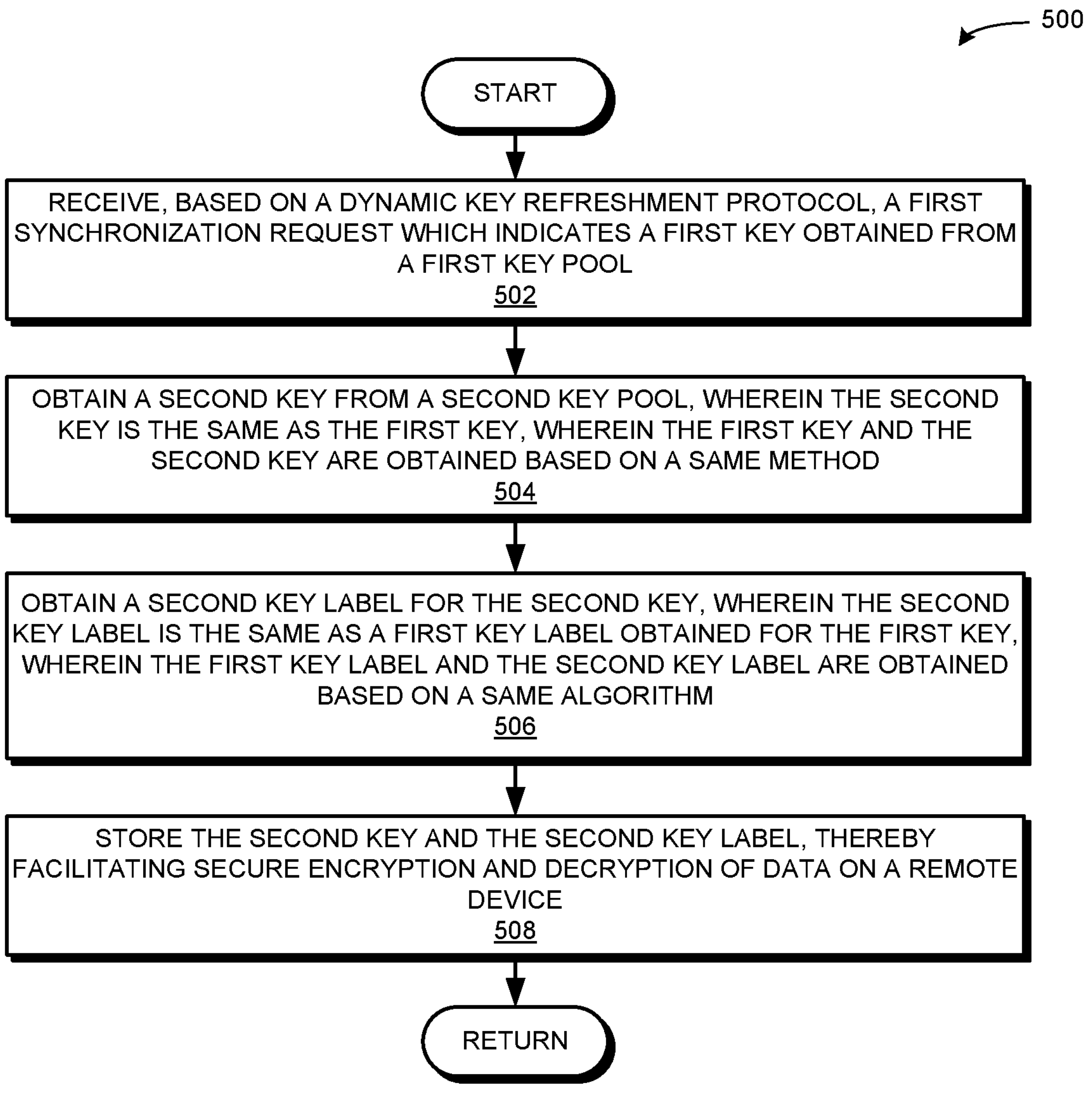
500
START
RECEIVE, BASED ON A DYNAMIC KEY REFRESHMENT PROTOCOL, A FIRST SYNCHRONIZATION REQUEST WHICH INDICATES A FIRST KEY OBTAINED FROM A FIRST KEY POOL
502
OBTAIN A SECOND KEY FROM A SECOND KEY POOL, WHEREIN THE SECOND KEY IS THE SAME AS THE FIRST KEY, WHEREIN THE FIRST KEY AND THE SECOND KEY ARE OBTAINED BASED ON A SAME METHOD
504
OBTAIN A SECOND KEY LABEL FOR THE SECOND KEY, WHEREIN THE SECOND KEY LABEL IS THE SAME AS A FIRST KEY LABEL OBTAINED FOR THE FIRST KEY, WHEREIN THE FIRST KEY LABEL AND THE SECOND KEY LABEL ARE OBTAINED BASED ON A SAME ALGORITHM
506
STORE THE SECOND KEY AND THE SECOND KEY LABEL, THEREBY FACILITATING SECURE ENCRYPTION AND DECRYPTION OF DATA ON A REMOTE DEVICE
508
RETURN

FIG. 5A

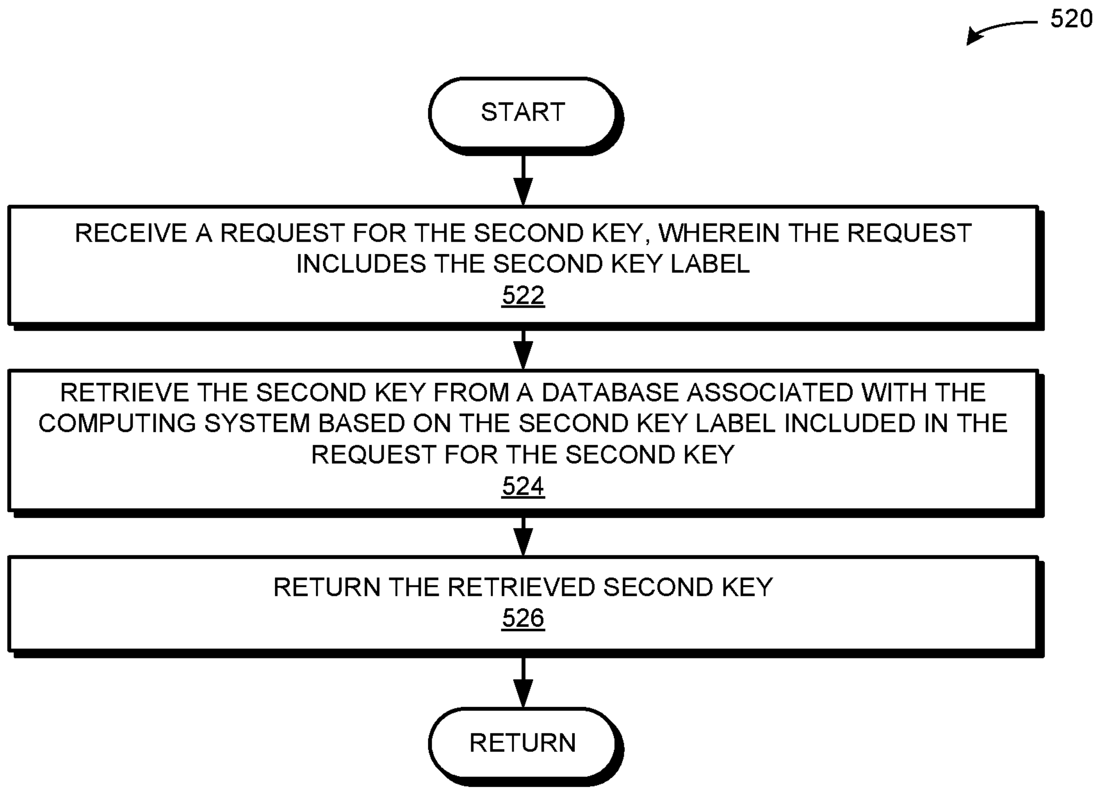
520
START
RECEIVE A REQUEST FOR THE SECOND KEY, WHEREIN THE REQUEST INCLUDES THE SECOND KEY LABEL
522
RETRIEVE THE SECOND KEY FROM A DATABASE ASSOCIATED WITH THE COMPUTING SYSTEM BASED ON THE SECOND KEY LABEL INCLUDED IN THE REQUEST FOR THE SECOND KEY
524
RETURN THE RETRIEVED SECOND KEY
526
RETURN

FIG. 5B

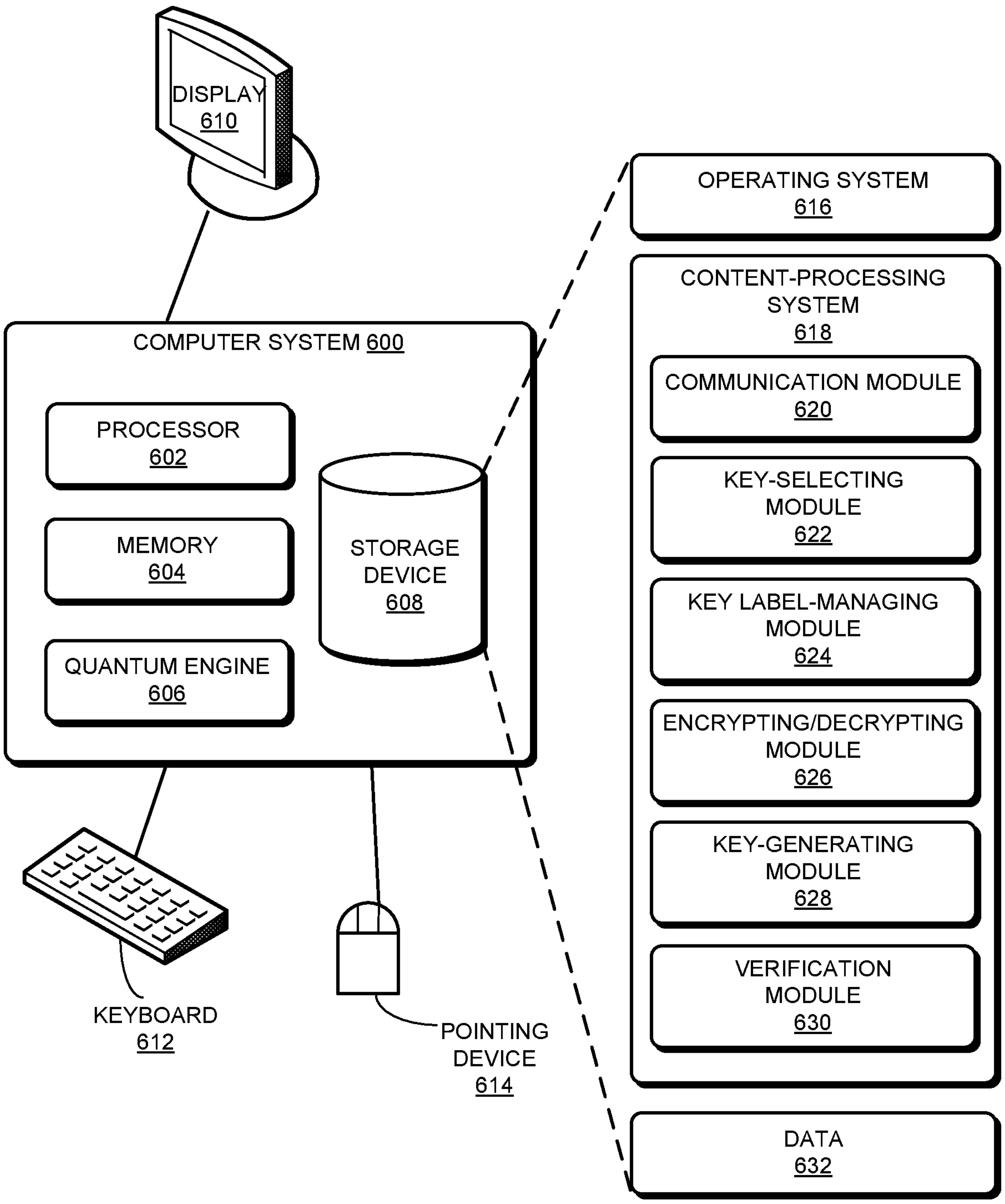
DISPLAY 610
COMPUTER SYSTEM 600
PROCESSOR 602
MEMORY 604
QUANTUM ENGINE 606
STORAGE DEVICE 608
KEYBOARD 612
POINTING DEVICE 614
OPERATING SYSTEM 616
CONTENT-PROCESSING SYSTEM 618
COMMUNICATION MODULE 620
KEY-SELECTING MODULE 622
KEY LABEL-MANAGING MODULE 624
ENCRYPTING/DECRYPTING MODULE 626
KEY-GENERATING MODULE 628
VERIFICATION MODULE 630
DATA 632

FIG. 6

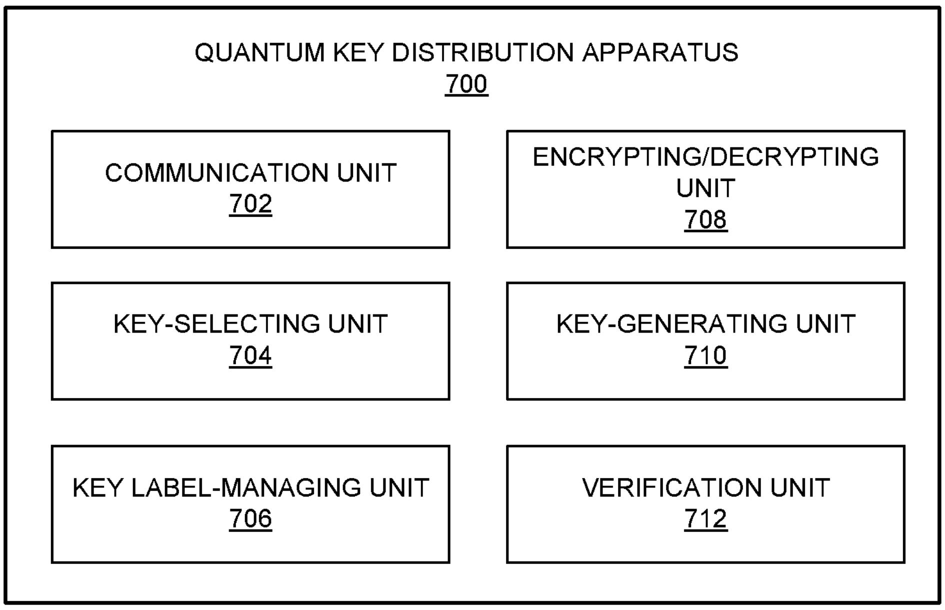
QUANTUM KEY DISTRIBUTION APPARATUS
700
COMMUNICATION UNIT
702
ENCRYPTING/DECRYPTING UNIT
708
KEY-SELECTING UNIT
704
KEY-GENERATING UNIT
710
KEY LABEL-MANAGING UNIT
706
VERIFICATION UNIT
712

FIG. 7