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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 August 2015 (13.08.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/119610 A1

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

(21) International Application Number:
PCT/US2014/015058

(22) International Filing Date:
6 February 2014 (06.02.2014)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **EMPIRE TECHNOLOGY DEVELOPMENT, LLC** [US/US]; 2711 Centerville Road, Suite 400, Wilmington, DE 19808 (US).

(72) Inventor: **KRUGLICK, Ezekiel**; 13842 Deergrass Ct., Poway, CA 92064-2276 (US).

(74) Agent: **TRUK, Carl, K.**; Turk IP Law, LLC, 2885 Sanford Ave. S.W. #23998, Grandville, MI 49418 (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, 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, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, 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, 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).

**Published**:

— *with international search report (Art. 21(3))*

(54) Title: SERVER-CLIENT SECRET GENERATION WITH CACHED DATA

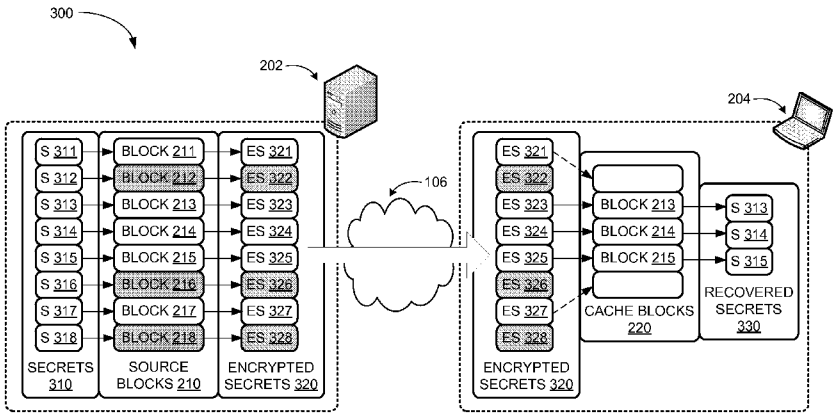


FIG. 3

(57) Abstract: Technologies are provided for shared secret generation between a server and a client using cached data. In some examples, a server may send a number of encrypted secrets to a client that caches a number of data blocks previously provided by the server. Each of the encrypted secrets may be encrypted using a data block that may or may not be cached at the client. The client may then identify the encrypted secrets that correspond to data blocks in its cache and use those data blocks to recover those secrets. The client may then encrypt a message for the server using the recovered secrets. Upon reception of the message, the server may then recover the message using its knowledge of the data blocks cached at the client.

WO 2015/119610 A1

# SERVER-CLIENT SECRET GENERATION WITH CACHED DATA

## BACKGROUND

**[0001]** Unless otherwise indicated herein, the materials described in this section are not prior art to the claims in this application and are not admitted to be prior art by inclusion in this section.

**[0002]** Web-enabled applications continue to grow in popularity. Some web applications may take advantage of application caches that store generated data, frequently accessed data, and/or reusable elements relatively close to end users, thereby reducing latency and accelerate application performance. Such caching may be implemented at a number of different levels. For example, content delivery networks (CDNs) are geographically distributed networks that may store static content close to end users. In some situations, application accelerators may improve web application or network performance by caching web application data at end user devices, for example on mobile devices such as laptops or smartphones.

## SUMMARY

**[0003]** The present disclosure generally describes techniques to generate shared secrets using cached data.

**[0004]** According to some examples, a method is provided to generate shared secrets between a server and a client. The method may include transmitting, by the server, multiple encrypted secrets corresponding to multiple data blocks to the client, the multiple encrypted secrets generated by encrypting each secret with a respective data block in the multiple data blocks. The method may further include recovering, by the client, a first subset of the multiple secrets from the encrypted secrets, where the first subset of secrets corresponds to a first subset of data blocks stored at the client and the first subset of data blocks is a subset of the multiple data blocks. The method may further include encrypting a message at the client using the first subset of secrets and transmitting, by the client, the message to the server. The method may further include recovering, by the server, the message using a second subset of the multiple secrets, where the second subset of secrets corresponds to a second subset of data blocks known

by the server to be previously stored at the client and the second subset of data blocks is a subset of the multiple data blocks.

**[0005]** According to other examples, a method for a server is provided to generate shared secrets with a client. The method may include generating multiple encrypted secrets corresponding to multiple data blocks by encrypting each secret in a set of multiple secrets with a respective data block in the multiple data blocks. The method may further include transmitting the multiple encrypted secrets to the client and receiving, from the client, a message encrypted using at least one secret from the multiple secrets. The method may further include recovering the message using a first subset of the multiple secrets, where the first subset of the secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the multiple data blocks.

**[0006]** According to further examples, a server is provided to generate shared secrets with a client. The server may include a memory and a processing module. The memory may be configured to store instructions and multiple data blocks. The processing module may be configured to generate multiple encrypted secrets corresponding to multiple data blocks by encrypting each secret in a set of multiple secrets using a respective data block in the multiple data blocks. The processing module may be further configured to transmit the multiple encrypted secrets to the client and receive from the client a message encrypted using at least one secret from the multiple secrets. The processing module may be further configured to recover the message using a first subset of the multiple secrets, where the first subset of secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the multiple data blocks.

**[0007]** According to yet further examples, a method for a client is provided to use shared secrets generated with a server. The method may include receiving from the server multiple encrypted secrets corresponding to multiple data blocks and recovering a first subset of secrets from the encrypted secrets, where the first subset of secrets corresponds to the first subset of data blocks. The method may further include encrypting a message using the first subset of secrets and transmitting the message to the server.

**[0008]** According to some examples, a client is provided to generate shared secrets with a server. The client may include a memory and a processing module. The memory may be configured to store instructions and a first subset of data blocks, where the first subset of data

blocks is a subset of multiple data blocks. The processing module may be configured to receive from the server multiple encrypted secrets corresponding to multiple data blocks and recover a first subset of secrets from the encrypted secrets, where the first subset of secrets corresponds to the first subset of data blocks. The processing module may be further configured to encrypt a message using the first subset of secrets and transmit the message to the server.

**[0009]** According to other examples, a computer readable medium may store instructions which when executed on one or more computing devices may execute a method to generate shared secrets between a server and a client. The methods may be similar to the methods described above.

**[0010]** The foregoing summary is illustrative only and is not intended to be in any way limiting. In addition to the illustrative aspects, embodiments, and features described above, further aspects, embodiments, and features will become apparent by reference to the drawings and the following detailed description.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0011]** The foregoing and other features of this disclosure will become more fully apparent from the following description and appended claims, taken in conjunction with the accompanying drawings. Understanding that these drawings depict only several embodiments in accordance with the disclosure and are, therefore, not to be considered limiting of its scope, the disclosure will be described with additional specificity and detail through use of the accompanying drawings, in which:

FIG. 1 illustrates an example datacenter-based system where server-client shared secrets generation using cached data may be implemented;

FIG. 2 illustrates an example system where a server may cache data at a client;

FIG. 3 illustrates an example process where a server may use data cached at a client to generate server-client shared secrets;

FIG. 4 illustrates an example system where cached data may be used to generate server-client shared secrets;

FIG. 5 illustrates a general purpose computing device, which may be used to generate server-client shared secrets using cached data;

FIG. 6 is a flow diagram illustrating an example method to generate server-client shared secrets using cached data that may be performed by a computing device such as the computing device in FIG. 5; and

FIG. 7 illustrates a block diagram of an example computer program product,

all arranged in accordance with at least some embodiments described herein.

## DETAILED DESCRIPTION

**[0012]** In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. In the drawings, similar symbols typically identify similar components, unless context dictates otherwise. The illustrative embodiments described in the detailed description, drawings, and claims are not meant to be limiting. Other embodiments may be utilized, and other changes may be made, without departing from the spirit or scope of the subject matter presented herein. The aspects of the present disclosure, as generally described herein, and illustrated in the Figures, can be arranged, substituted, combined, separated, and designed in a wide variety of different configurations, all of which are explicitly contemplated herein.

**[0013]** This disclosure is generally drawn, *inter alia*, to methods, apparatus, systems, devices, and/or computer program products related to shared secret generation between a server and a client using cached data.

**[0014]** Briefly stated, technologies are generally described for shared secret generation between a server and a client using cached data. In some examples, a server may send a number of encrypted secrets to a client that caches a number of data blocks previously provided by the server. Each of the encrypted secrets may be encrypted using a data block that may or may not be cached at the client. The client may then identify the encrypted secrets that correspond to data blocks in its cache and use those data blocks to recover those secrets. The client may then encrypt a message for the server using the recovered secrets. Upon reception of the message, the server may then recover the message using its knowledge of the data blocks cached at the client.

**[0015]** A datacenter as used herein refers to an entity that hosts services and applications for customers through one or more physical server installations and one or more virtual machines executed in those server installations. Customers of the datacenter, also referred to as tenants, may be organizations that provide access to their services for multiple users. One example

configuration may include an online retail service that provides retail sale services to consumers (users). The retail service may employ multiple applications (e.g., presentation of retail goods, purchase management, shipping management, inventory management, etc.), which may be hosted by one or more datacenters. Thus, a consumer may communicate with those applications of the retail service through a client application such as a browser over one or more networks and receive the provided service without realizing where the individual applications are actually executed.

**[0016]** In some situations, it may be desirable to generate shared server-client secrets for use in securing communications between a server and a client. Various embodiments described herein are directed to generation of shared server-client secrets based on cached data. A server configured to store some data may cache a subset of the data at a client to accelerate web application performance at the client. Because the server may know the data subset it cached at the client and the client itself may know what data it has cached, secrets shared between the server and the client may be generated based on the cached data. Additional security may be added to protect against man-in-the-middle attacks by obfuscating the exact data cached at the client, as described in more detail below.

**[0017]** FIG. 1 illustrates an example datacenter-based system where server-client shared secrets generation using cached data may be implemented, arranged in accordance with at least some embodiments described herein.

**[0018]** As shown in a diagram 100, a physical datacenter 102 may include one or more physical servers 110, 111, and 113, each of which may be configured to provide one or more virtual machines 104. For example, the physical servers 111 and 113 may be configured to provide four virtual machines and two virtual machines, respectively. In some embodiments, one or more virtual machines may be combined into one or more virtual datacenters. For example, the four virtual machines provided by the server 111 may be combined into a virtual datacenter 112. The virtual machines 104 and/or the virtual datacenter 112 may be configured to provide cloud-related data/computing services such as various applications, data storage, data processing, or comparable ones to a group of customers 108, such as individual users or enterprise customers, via a network 106.

**[0019]** As described above, an application accelerator may create an application data cache at an end user device. The presence of the cache may accelerate web application performance at

the device, because data available in the local cache can be fetched by the application faster than data from the web. An application data cache may be reactive (i.e., storing frequently-accessed content) and/or predictive (storing data the system predicts a user may access, such as mail attachments or updated documents). An accelerator hardware appliance or software installation at a remote server may manage the data cache at the device, for example by scheduling data transfer to avoid interrupting other traffic while tracking the data cached on the device.

**[0020]** FIG. 2 illustrates an example system where a server may cache data at a client, arranged in accordance with at least some embodiments described herein.

**[0021]** According to a diagram 200, a server 202 may communicate with a client 204. The server 202 may be or implement an application accelerator server or appliance, as described above, and may provide application data blocks to cache at end user devices to accelerate applications executing on the devices. For example, the server 202 may store source blocks 210 including blocks 211-218 to cache at end user devices. In turn, the client 204, which may be a physical end user device or an application executing on an end user device, may store cache blocks 220 provided by the server 202 in system memory (e.g., random access memory) or storage memory (e.g., removable or non-removable storage such as hard drives, solid-state drives, and the like). The cache blocks 220 at the client 204 may not include all of the source blocks 210 stored at the server 202. For example, the cache blocks 220 may include the block 211, the blocks 213-215, and the block 217 instead of all of the blocks 211-218. Web applications executing on the end user device may then be accelerated using the cache blocks 220.

**[0022]** In some embodiments, after a web application session has been concluded or paused, the client 204 may indicate that the memory storing some or all of the cache blocks 220 as available, despite still containing the cache blocks 220. This may be done to keep the cache blocks 220 from consuming visible amounts of memory or disk space and impacting other systems or applications. This may mean that the contents of the cache blocks 220 may decrease over time as other applications and data overwrite portions of the cache blocks 220. Moreover, if no active connection is maintained between the server 202 and the client 204, the server 202 may not know which of the blocks in the cache blocks 220 are still present and which have been overwritten. This situation is depicted in a diagram 250, also illustrated in FIG. 2, which shows the contents of the cache blocks 220 after web application processing has concluded or paused

for some time. As shown in the diagram 250, the client 204 may have deleted the block 211 and the block 217 from the cache blocks 220. However, this deletion may not be known to the server 202, which may still record that the previously-provided blocks 211, 213-215, and 217 still reside in the cache blocks 220 at the client 204.

**[0023]** FIG. 3 illustrates an example process where a server may use data cached at a client to generate server-client shared secrets, arranged in accordance with at least some embodiments described herein.

**[0024]** As described above, data cached by a server (e.g., the server 202) at a client (e.g., the client 204) may be overwritten over time, without the knowledge of the server. As a result, at some point in time the data recorded by the server as being cached at the client may not match the actual data cached at the client.

**[0025]** According to a diagram 300, the server 202 may generate or otherwise obtain a set of secrets 310. In some embodiments, the secrets 310 may include random numbers generated by the server 202 or received from some other source. For example, a hardware random number generator may be used. The secrets may also be non-invertible hashes of particular data or source table. The number of secrets in the set of secrets 310 may match the number of data blocks in the source blocks 210, and the server 202 may encrypt each secret in the set of secrets 310 using a data block in the source blocks 210 to generate a set of encrypted secrets 320, for example by generating a hash of each source block to serve as a key for encrypting each secret using an established encryption algorithm. For example, the set of secrets 310 may include secrets 311-318. Each of the secrets 311-318 may be encrypted using a corresponding data block in the source blocks 210 to form encrypted secrets 321-328. The secret 311 may be encrypted using the data block 211 to form the encrypted secret 321, the secret 312 may be encrypted using the data block 212 to form the encrypted secret 322, the secret 313 may be encrypted using the data block 213 to form the encrypted secret 323, and so on. The server 202 may specifically use data blocks that have not been previously provided to the client 202, such as the blocks 212, 216, and 218, to encrypt secrets. These encrypted secrets may serve as decoys for attackers, as described below. Alternate decoys may also be encrypted using random or meaningless keys.

**[0026]** In some embodiments, the encryption may be performed using symmetric cryptography, where the data blocks in the source blocks 210 are used as the encryption keys for the encryption operation, either directly or by deriving a key from them as by hashing. The

server 202 may then send the set of encrypted secrets 320 to the client 204 over the network 106. In some embodiments, the server 202 may also send a set of identifiers identifying the data block corresponding to each of the encrypted secrets in the set of encrypted secrets 320 to the client 204. For example, the set of identifiers may indicate that encrypted secret 321 corresponds to block 211, encrypted secret 322 corresponds to block 212, and encrypted secret 323 corresponds to block 213, and so on.

**[0027]** In response to receiving the set of encrypted secrets 320, the client 204 may recover one or more secrets in the set of secrets 310 using retained data blocks in the cache blocks 220. As depicted in the diagram 300, the client 204 may retain the blocks 213-215 in the cache blocks 220. The client 204 may then identify the encrypted secrets in the set of encrypted secrets 320 that correspond to the blocks 213-215, for example using the set of identifiers provided by the server 202. After identifying the appropriate encrypted secrets, which in this instance are the encrypted secrets 323-325, the client 204 may then use the blocks 213-215 to decrypt the encrypted secrets 323-325, thereby recovering a set of secrets 330 that includes the secrets 313-315.

**[0028]** Subsequently, the client 204 may then use the recovered secrets 330 to encrypt a message to the server 202. In some embodiments, the client 204 may use every secret in the recovered secrets 330 (i.e., secrets 313, 314, and 315) to encrypt the message, although in other embodiments the client 204 may use less than every secret. The client 204 may encrypt the message in a number of ways. For example, the client 204 may encrypt the message serially, by first encrypting the message with one secret to produce a first result, then encrypting the first result with another secret to produce a second result, then encrypting the second result with another secret, and so on. As another example, the client 204 may combine the secrets in the recovered secrets 330 to form a longer secret, then use the longer secret to encrypt the message. In some embodiments, the message itself may contain a secondary secret determined by the client 204 for use by the server 202 to further secure server-client communications. For example, the message may include a handshake, login information for the server 202, one of the secrets in the recovered secrets 330, or any other suitable authentication information to secure the server-client connection once a secure shared secret exists. After encryption, the client 204 may then send the encrypted message to the server 202. In some embodiments, the client 204 may also send an indication of the number of secrets (but not the identity of the secrets) used to encrypt

the message. For example, the client 204 may indicate that three secrets were used to encrypt the message if secrets 313, 314, and 315 were used to encrypt the message.

**[0029]** In response to receiving the encrypted message, the server 202 may recover the message using its record of data blocks previously provided to the client 204. For example, the server 202 may record that it previously provided the blocks 211, 213-215, and 217 to the client 204. Based on that knowledge, the server 202 may then know that the client 204 could at most recover five secrets from the encrypted secrets 320. Specifically, the server 202 may know that the client could at most recover the secrets 311, 313-315, and 317, since those secrets were encrypted using the blocks 211, 213-215, and 217. By iteratively using various combinations and permutations of two, three, four, and five of the secrets 311, 313-315, and 317 to attempt decryption of the encrypted message, the server 202 may be able to eventually recover the message. In some embodiments, the client 204 may provide an indication of the number and/or order of secrets used to encrypt the message, and the server 202 may use the number of secrets to reduce the number of combinations and permutations to be tested in order to recover the message.

**[0030]** While the server 202 may iterate through a number of combinations and permutations of secrets, the iteration may be relatively quick, especially when the number of data blocks previously provided to the client 204 by the client 202 and used for secret generation is relatively small. Meanwhile, an attacker somewhere in the network 106 may intercept the set of encrypted secrets 320 sent from the server 202 and the encrypted message sent from the client 204. The attacker may even know the set of source blocks 210, and, therefore, be able to determine the secrets 310. However, the attacker will not know the identity of the cache blocks 220, or even the identity of the data blocks previously provided by the server 202 to the client 204 to cache. As a result, the attacker may have to perform the iterative combination and permutation decryption process using all of the secrets 310. Moreover, the attacker may have to successfully decrypt the encrypted message, modify the message, and re-encrypt the message before the server 202 and the client 204 realize that the message has been compromised (e.g., by detecting an excessive time delay). If the number of secrets in the secrets 310 is substantially larger than the number of blocks previously provided to the client 204 to cache, the computational cost to the attacker to decrypt the encrypted message before the server 202 and the client 204 detect the excessive time delay may be intractably large. If an attacker instead opts to

use randomly selected data blocks from the source blocks 210 to encrypt a counterfeit message, it may be likely that at least one of the data blocks is a decoy (i.e., not corresponding to a data block in the cache blocks 220), resulting in an invalid message.

**[0031]** FIG. 4 illustrates an example system where cached data may be used to generate server-client shared secrets, arranged in accordance with at least some embodiments described herein.

**[0032]** According to a diagram 400, the server 202 may implement an application accelerator server module 412 that may accelerate web application execution on the client 204 by sending application data to an application accelerator client module 432 to cache at a cache 430. The application accelerator server module 412 may store and retrieve application data blocks (e.g., the source blocks 210) at a file system 410. Upon sending data blocks to the client 204 to cache, the application accelerator server module 412 may use a user cache block tracking module 414 to record the data blocks sent to the client 204 in user records 416.

**[0033]** A security interface 422 may then generate encrypted secrets (e.g., the encrypted secrets 320) intended for the client 204. For example, the security interface 422 may use a secret generator 418 (e.g., a random number generator) to generate a number of secrets corresponding to some or all of the number of cached data blocks at the client 204. The security interface 422 may then encrypt each secret with a corresponding cached data block. The security interface 422 may also use a decoy generator 420 to generate a number of decoy secrets. Some members of encrypted secrets may be generated for source blocks that are known not to be in the client cache blocks. These secrets may be referred to as decoy secrets or encrypted decoys and be generated, in some examples, by the decoy generator 420. In some embodiments, the secret generator 418 may instead be used to generate the decoy secrets. The security interface 422 may then encrypt each of the decoy secrets with a corresponding data block that the application accelerator server module 412 has not previously provided to the client 204. Subsequently, the security interface 422 may associate each encrypted secret with an identifier identifying the particular data block used in the encryption of that secret. The security interface 422 may then transmit the encrypted secrets and the identifiers to the client 204.

**[0034]** The client 204 may itself include a security interface 434, which may receive the encrypted secrets and the identifiers from the server 202. In response to receiving the encrypted secrets and the identifiers, the security interface 434 may identify the data blocks cached at the

cache 430 and use the identity of the cached data blocks and the provided identifiers to determine which encrypted secrets correspond to the cached data blocks. A cache block-based decryption module 436 may then decrypt the encrypted secrets corresponding to the cached data blocks using the corresponding cached data blocks. A shared secret handshake module 438 may then use the recovered secrets to encrypt a message containing authentication information (e.g., a handshake, login information, a recovered secret, etc.) for the server 202. The shared secret handshake module 438 may then provide the encrypted message to the security interface 434 to transmit to the security interface 422 of the server 202.

**[0035]** In response to receiving the encrypted message, the security interface 422 may forward the encrypted message to a permutation module 424. The permutation module 424, in turn, may attempt to decrypt the encrypted message using various permutations of data blocks previously provided to the client 204 and recorded in the user records 416. Once decryption is successful, the permutation module 424 may deliver the message and client cache status (represented by the actual data block permutation used to successfully decrypt the message) to an intrusion detection module 426. In some embodiments, the intrusion detection module 426 may be used to evaluate the security status of the client 204 and/or the connection between the server 202 and the client 204. For example, if client 204 was recently connected to server 202, yet had a relatively large change in the contents of cache 430, the security status of the client 204 and/or the server-client connection may be suspect. Similarly, the security status of the client 204 and/or the server-client connection may be suspect when client 204 has been disconnected from server 202 for a long period of time, yet the contents of cache 430 has not changed significantly. The security status of the client 204 and/or the server-client connection may also be suspect if the security response took longer than expected or if the security response contains decryptions of decoys, for example.

**[0036]** FIG. 5 illustrates a general purpose computing device, which may be used to generate server-client shared secrets using cached data, arranged in accordance with at least some embodiments described herein.

**[0037]** For example, the computing device 500 may be used to generate server-client shared secrets using cached data as described herein. In an example basic configuration 502, the computing device 500 may include one or more processors 504 and a system memory 506. A memory bus 508 may be used to communicate between the processor 504 and the system

memory 506. The basic configuration 502 is illustrated in FIG. 5 by those components within the inner dashed line.

**[0038]** Depending on the desired configuration, the processor 504 may be of any type, including but not limited to a microprocessor (μP), a microcontroller (μC), a digital signal processor (DSP), or any combination thereof. The processor 504 may include one more levels of caching, such as a level cache memory 512, a processor core 514, and registers 516. The example processor core 514 may include an arithmetic logic unit (ALU), a floating point unit (FPU), a digital signal processing core (DSP Core), or any combination thereof. An example memory controller 518 may also be used with the processor 504, or in some implementations the memory controller 518 may be an internal part of the processor 504.

**[0039]** Depending on the desired configuration, the system memory 506 may be of any type including but not limited to volatile memory (such as RAM), non-volatile memory (such as ROM, flash memory, etc.) or any combination thereof. The system memory 506 may include an operating system 520, a cryptographic module 522, and program data 524. The cryptographic module 522 may include a permutation decryption module 526 to implement using data block permutation to recover encrypted messages as described herein. The program data 524 may include, among other data, client cache data 528, cache data blocks 530, or the like, as described herein.

**[0040]** The computing device 500 may have additional features or functionality, and additional interfaces to facilitate communications between the basic configuration 502 and any desired devices and interfaces. For example, a bus/interface controller 530 may be used to facilitate communications between the basic configuration 502 and one or more data storage devices 532 via a storage interface bus 534. The data storage devices 532 may be one or more removable storage devices 536, one or more non-removable storage devices 538, or a combination thereof. Examples of the removable storage and the non-removable storage devices include magnetic disk devices such as flexible disk drives and hard-disk drives (HDD), optical disk drives such as compact disk (CD) drives or digital versatile disk (DVD) drives, solid state drives (SSD), and tape drives to name a few. Example computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data.

**[0041]** The system memory 506, the removable storage devices 536 and the non-removable storage devices 538 are examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD), solid state drives, or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which may be used to store the desired information and which may be accessed by the computing device 500. Any such computer storage media may be part of the computing device 500.

**[0042]** The computing device 500 may also include an interface bus 540 to facilitate communication from various interface devices (e.g., one or more output devices 542, one or more peripheral interfaces 544, and one or more communication devices 566) to the basic configuration 502 via the bus/interface controller 530. Some of the example output devices 542 include a graphics processing unit 548 and an audio processing unit 550, which may be configured to communicate to various external devices such as a display or speakers via one or more A/V ports 552. One or more example peripheral interfaces 544 may include a serial interface controller 554 or a parallel interface controller 556, which may be configured to communicate with external devices such as input devices (e.g., keyboard, mouse, pen, voice input device, touch input device, etc.) or other peripheral devices (e.g., printer, scanner, etc.) via one or more I/O ports 558. An example communication device 566 includes a network controller 560, which may be arranged to facilitate communications with one or more other computing devices 562 over a network communication link via one or more communication ports 564. The one or more other computing devices 562 may include servers at a datacenter, customer equipment, and comparable devices.

**[0043]** The network communication link may be one example of a communication media. Communication media may be embodied by computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave or other transport mechanism, and may include any information delivery media. A "modulated data signal" may be a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media may include wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, radio frequency (RF), microwave, infrared (IR)

and other wireless media. The term computer readable media as used herein may include both storage media and communication media.

**[0044]** The computing device 500 may be implemented as a part of a general purpose or specialized server, mainframe, or similar computer that includes any of the above functions. The computing device 500 may also be implemented as a personal computer including both laptop computer and non-laptop computer configurations.

**[0045]** FIG. 6 is a flow diagram illustrating an example method to generate server-client shared secrets using cached data that may be performed by a computing device such as the computing device in FIG. 5, arranged in accordance with at least some embodiments described herein.

**[0046]** Example methods may include one or more operations, functions or actions as illustrated by one or more of blocks 622, 624, 626, 628, and/or 630, and may in some embodiments be performed by a computing device such as the computing device 600 in FIG. 6. The operations described in the blocks 622-630 may also be stored as computer-executable instructions in a computer-readable medium such as a computer-readable medium 620 of a computing device 610.

**[0047]** An example process to generate server-client shared secrets may begin with block 622, "TRANSMIT MULTIPLE ENCRYPTED SECRETS CORRESPONDING TO MULTIPLE DATA BLOCKS FROM THE SERVER TO THE CLIENT, EACH ENCRYPTED SECRET ENCRYPTED USING A RESPECTIVE DATA BLOCK", where a set of encrypted secrets (e.g., the encrypted secrets 320), each encrypted with a particular data block as described above, may be sent from a server (e.g., the server 202) to a client (e.g., the client 204).

**[0048]** Block 622 may be followed by block 624, "RECOVER, AT THE CLIENT, A FIRST SUBSET OF SECRETS FROM THE ENCRYPTED SECRETS CORRESPONDING TO A FIRST SUBSET OF DATA BLOCKS STORED AT THE CLIENT", where the client uses cached data blocks in a set of cache blocks (e.g., the cache blocks 220) stored at the client to recover secrets from the particular encrypted secrets in the set of encrypted secrets that correspond to those cache data blocks, as described above.

**[0049]** Block 624 may be followed by block 626, "ENCRYPT A MESSAGE AT THE CLIENT USING THE FIRST SUBSET OF SECRETS", where the client may use the recovered secrets to encrypt a message for the server as described above. In some embodiments, the

message may contain a handshake or other information suitable to establish a secure server-client connection.

**[0050]** Block 626 may be followed by block 628, "TRANSMIT THE MESSAGE FROM THE CLIENT TO THE SERVER", where the client transmits the encrypted message to the server as described above.

**[0051]** Block 628 may be followed by block 630, "RECOVER, AT THE SERVER, THE MESSAGE USING A SECOND SUBSET OF SECRETS CORRESPONDING TO A SECOND SUBSET OF DATA BLOCKS KNOWN BY THE SERVER TO BE PREVIOUSLY STORED AT THE CLIENT", where the server recovers the message based on a record of data blocks previously provided to the client, as described above. In some embodiments, the server tests various permutations of the data blocks previously provided to the client to decrypt the message until a successful permutation is found.

**[0052]** FIG. 7 illustrates a block diagram of an example computer program product, arranged in accordance with at least some embodiments described herein.

**[0053]** In some examples, as shown in FIG. 7, a computer program product 700 may include a signal bearing medium 702 that may also include one or more machine readable instructions 704 that, when executed by, for example, a processor may provide the functionality described herein. Thus, for example, referring to the processor 504 in FIG. 5, the cryptographic module 522 may undertake one or more of the tasks shown in FIG. 7 in response to the instructions 704 conveyed to the processor 504 by the medium 702 to perform actions associated with generating server-client shared secrets as described herein. Some of those instructions may include, for example, transmitting multiple encrypted secrets corresponding to multiple data blocks from the server to the client, each encrypted secret encrypted using a respective data block, recovering, at the client, a first subset of secrets from the encrypted secrets corresponding to a first subset of data blocks stored at the client, encrypting a message at the client using the first subset of secrets, transmitting the message from the client to the server, and/or recovering, at the server, the message using a second subset of secrets corresponding to a second subset of data blocks known by the server to be previously stored at the client, according to some embodiments described herein.

**[0054]** In some implementations, the signal bearing media 702 depicted in FIG. 7 may encompass computer-readable media 706, such as, but not limited to, a hard disk drive, a solid

state drive, a Compact Disc (CD), a Digital Versatile Disk (DVD), a digital tape, memory, etc. In some implementations, the signal bearing media 702 may encompass recordable media 707, such as, but not limited to, memory, read/write (R/W) CDs, R/W DVDs, etc. In some implementations, the signal bearing media 702 may encompass communications media 710, such as, but not limited to, a digital and/or an analog communication medium (e.g., a fiber optic cable, a waveguide, a wired communications link, a wireless communication link, etc.). Thus, for example, the program product 700 may be conveyed to one or more modules of the processor 504 by an RF signal bearing medium, where the signal bearing media 702 is conveyed by the wireless communications media 710 (e.g., a wireless communications medium conforming with the IEEE 802.11 standard).

**[0055]** According to some examples, a method is provided to generate shared secrets between a server and a client. The method may include transmitting, by the server, multiple encrypted secrets corresponding to multiple data blocks to the client, the multiple encrypted secrets generated by encrypting each secret with a respective data block in the multiple data blocks. The method may further include recovering, by the client, a first subset of the multiple secrets from the encrypted secrets, where the first subset of secrets corresponds to a first subset of data blocks stored at the client and the first subset of data blocks is a subset of the multiple data blocks. The method may further include encrypting a message at the client using the first subset of secrets and transmitting, by the client, the message to the server. The method may further include recovering, by the server, the message using a second subset of the multiple secrets, where the second subset of secrets corresponds to a second subset of data blocks known by the server to be previously stored at the client and the second subset of data blocks is a subset of the multiple data blocks.

**[0056]** According to some embodiments, the method may further include transmitting, by the server, multiple identifiers to the client, each identifier associated with a respective encrypted secret in the multiple encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret. The method may further include recovering, by the client, the first subset of secrets by using the multiple identifiers to identify the encrypted secrets corresponding to the first subset of data blocks. The multiple secrets may include a random number and/or a third subset of secrets corresponding to a third subset of data blocks known by

the server to not be stored at the client, and the third subset of data blocks may be a subset of the multiple data blocks.

**[0057]** According to other embodiments, the method may further include encrypting, by the client, the message using every secret in the first subset of secrets, and transmitting, by the client, an indicator of a number of secrets in the first subset of secrets to the server. The method may further include recovering, by the server, the message using a permutation of data blocks in the second subset of data blocks based on the indicator. The message may include a handshake, login information, and/or at least one secret from the first subset of secrets. The method may further include establishing a secondary secret shared between the server and the client based on the message. The first subset of data blocks may include data cached at the client for application acceleration.

**[0058]** According to other examples, a method for a server is provided to generate shared secrets with a client. The method may include generating multiple encrypted secrets corresponding to multiple data blocks by encrypting each secret in a set of multiple secrets with a respective data block in the multiple data blocks. The method may further include transmitting the multiple encrypted secrets to the client and receiving, from the client, a message encrypted using at least one secret from the multiple secrets. The method may further include recovering the message using a first subset of the multiple secrets, where the first subset of the secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the multiple data blocks.

**[0059]** According to some embodiments, the method may further include transmitting multiple identifiers to the client, each identifier associated with a respective encrypted secret in the multiple encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret. The method may further include receiving, from the client, an indicator of a number of secrets used to encrypt the message, and recovering the message using a permutation of data blocks in the first subset of data blocks based on the indicator.

**[0060]** According to further examples, a server is provided to generate shared secrets with a client. The server may include a memory and a processing module. The memory may be configured to store instructions and multiple data blocks. The processing module may be configured to generate multiple encrypted secrets corresponding to multiple data blocks by encrypting each secret in a set of multiple secrets using a respective data block in the multiple

data blocks. The processing module may be further configured to transmit the multiple encrypted secrets to the client and receive from the client a message encrypted using at least one secret from the multiple secrets. The processing module may be further configured to recover the message using a first subset of the multiple secrets, where the first subset of secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the multiple data blocks.

**[0061]** According to some embodiments, the processing block may be further configured to transmit multiple identifiers to the client, each identifier associated with a respective encrypted secret in the multiple encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret. The multiple secrets may include a random number and/or a second subset of secrets corresponding to a second subset of data blocks known by the server to not be stored at the client, and the second subset of data blocks may be a subset of the multiple data blocks.

**[0062]** According to other embodiments, the processing block may be further configured to receive, from the client, an indicator of a number of secrets used to encrypt the message and recover the message using a permutation of data blocks in the first subset of data blocks based on the indicator. The message may include a handshake, login information, and/or at least one secret from the first subset of secrets. The processing block may be further configured to establish a secondary secret shared between the server and the client based on the message. The first subset of data blocks may include data cached at the client for application acceleration.

**[0063]** According to yet further examples, a method for a client is provided to use shared secrets generated with a server. The method may include receiving from the server multiple encrypted secrets corresponding to multiple data blocks and recovering a first subset of secrets from the encrypted secrets, where the first subset of secrets corresponds to the first subset of data blocks. The method may further include encrypting a message using the first subset of secrets and transmitting the message to the server.

**[0064]** According to some embodiments, the method may further include receiving, from the server, multiple identifiers, each identifier associated with a respective encrypted secret in the multiple encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret. The method may further include recovering the first subset of secrets by using the multiple identifiers to identify the encrypted secrets corresponding to the first subset

of data blocks. The method may further include encrypting the message using every secret in the first subset of secrets and transmitting an indicator of a number of secrets in the first subset of secrets to the server.

**[0065]** According to some examples, a client is provided to generate shared secrets with a server. The client may include a memory and a processing module. The memory may be configured to store instructions and a first subset of data blocks, where the first subset of data blocks is a subset of multiple data blocks. The processing module may be configured to receive from the server multiple encrypted secrets corresponding to multiple data blocks and recover a first subset of secrets from the encrypted secrets, where the first subset of secrets corresponds to the first subset of data blocks. The processing module may be further configured to encrypt a message using the first subset of secrets and transmit the message to the server.

**[0066]** According to some embodiments, the processing block may be further configured to receive multiple identifiers from the server, each identifier associated with a respective encrypted secret in the multiple encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret. The processing block may be further configured to recover the first subset of secrets by using the multiple identifiers to identify the encrypted secrets corresponding to the first subset of data blocks. The multiple secrets may include a random number. The processing block may be further configured to encrypt the message using every secret in the first subset of secrets and transmit an indicator of a number of secrets in the first subset of secrets to the server.

**[0067]** According to other embodiments, the message may include a handshake, login information, and/or at least one secret from the first subset of secrets. The processing block may be further configured to establish a secondary secret shared between the server and the client based on the message. The first subset of data blocks may include data cached at the client for application acceleration.

**[0068]** According to other examples, a computer readable medium may store instructions which when executed on one or more computing devices may execute a method to generate shared secrets between a server and a client. The methods may be similar to the methods described above.

**[0069]** There is little distinction left between hardware and software implementations of aspects of systems; the use of hardware or software is generally (but not always, in that in certain

contexts the choice between hardware and software may become significant) a design choice representing cost vs. efficiency tradeoffs. There are various vehicles by which processes and/or systems and/or other technologies described herein may be effected (e.g., hardware, software, and/or firmware), and that the preferred vehicle will vary with the context in which the processes and/or systems and/or other technologies are deployed. For example, if an implementer determines that speed and accuracy are paramount, the implementer may opt for a mainly hardware and/or firmware vehicle; if flexibility is paramount, the implementer may opt for a mainly software implementation; or, yet again alternatively, the implementer may opt for some combination of hardware, software, and/or firmware.

**[0070]** The foregoing detailed description has set forth various embodiments of the devices and/or processes via the use of block diagrams, flowcharts, and/or examples. Insofar as such block diagrams, flowcharts, and/or examples contain one or more functions and/or operations, it will be understood by those within the art that each function and/or operation within such block diagrams, flowcharts, or examples may be implemented, individually and/or collectively, by a wide range of hardware, software, firmware, or virtually any combination thereof. In one embodiment, several portions of the subject matter described herein may be implemented via Application Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), digital signal processors (DSPs), or other integrated formats. However, those skilled in the art will recognize that some aspects of the embodiments disclosed herein, in whole or in part, may be equivalently implemented in integrated circuits, as one or more computer programs executing on one or more computers (e.g., as one or more programs executing on one or more computer systems), as one or more programs executing on one or more processors (e.g., as one or more programs executing on one or more microprocessors), as firmware, or as virtually any combination thereof, and that designing the circuitry and/or writing the code for the software and or firmware would be well within the skill of one of skill in the art in light of this disclosure.

**[0071]** The present disclosure is not to be limited in terms of the particular embodiments described in this application, which are intended as illustrations of various aspects. Many modifications and variations can be made without departing from its spirit and scope, as will be apparent to those skilled in the art. Functionally equivalent methods and apparatuses within the scope of the disclosure, in addition to those enumerated herein, will be apparent to those skilled in the art from the foregoing descriptions. Such modifications and variations are intended to fall

within the scope of the appended claims. The present disclosure is to be limited only by the terms of the appended claims, along with the full scope of equivalents to which such claims are entitled. It is also to be understood that the terminology used herein is for the purpose of describing particular embodiments only, and is not intended to be limiting.

**[0072]** In addition, those skilled in the art will appreciate that the mechanisms of the subject matter described herein are capable of being distributed as a program product in a variety of forms, and that an illustrative embodiment of the subject matter described herein applies regardless of the particular type of signal bearing medium used to actually carry out the distribution. Examples of a signal bearing medium include, but are not limited to, the following: a recordable type medium such as a floppy disk, a hard disk drive, a Compact Disc (CD), a Digital Versatile Disk (DVD), a digital tape, a computer memory, a solid state drive, etc.; and a transmission type medium such as a digital and/or an analog communication medium (e.g., a fiber optic cable, a waveguide, a wired communications link, a wireless communication link, etc.).

**[0073]** Those skilled in the art will recognize that it is common within the art to describe devices and/or processes in the fashion set forth herein, and thereafter use engineering practices to integrate such described devices and/or processes into data processing systems. That is, at least a portion of the devices and/or processes described herein may be integrated into a data processing system via a reasonable amount of experimentation. Those having skill in the art will recognize that a data processing system may include one or more of a system unit housing, a video display device, a memory such as volatile and non-volatile memory, processors such as microprocessors and digital signal processors, computational entities such as operating systems, drivers, graphical user interfaces, and applications programs, one or more interaction devices, such as a touch pad or screen, and/or control systems including feedback loops and control motors (e.g., feedback for sensing position and/or velocity of gantry systems; control motors to move and/or adjust components and/or quantities).

**[0074]** A data processing system may be implemented utilizing any suitable commercially available components, such as those found in data computing/communication and/or network computing/communication systems. The herein described subject matter sometimes illustrates different components contained within, or connected with, different other components. It is to be understood that such depicted architectures are merely exemplary, and that in fact many other

architectures may be implemented which achieve the same functionality. In a conceptual sense, any arrangement of components to achieve the same functionality is effectively "associated" such that the desired functionality is achieved. Hence, any two components herein combined to achieve a particular functionality may be seen as "associated with" each other such that the desired functionality is achieved, irrespective of architectures or intermediate components. Likewise, any two components so associated may also be viewed as being "operably connected", or "operably coupled", to each other to achieve the desired functionality, and any two components capable of being so associated may also be viewed as being "operably couplable", to each other to achieve the desired functionality. Specific examples of operably couplable include but are not limited to physically connectable and/or physically interacting components and/or wirelessly interactable and/or wirelessly interacting components and/or logically interacting and/or logically interactable components.

**[0075]** With respect to the use of substantially any plural and/or singular terms herein, those having skill in the art can translate from the plural to the singular and/or from the singular to the plural as is appropriate to the context and/or application. The various singular/plural permutations may be expressly set forth herein for sake of clarity.

**[0076]** It will be understood by those within the art that, in general, terms used herein, and especially in the appended claims (*e.g.*, bodies of the appended claims) are generally intended as "open" terms (*e.g.*, the term "including" should be interpreted as "including but not limited to," the term "having" should be interpreted as "having at least," the term "includes" should be interpreted as "includes but is not limited to," etc.). It will be further understood by those within the art that if a specific number of an introduced claim recitation is intended, such an intent will be explicitly recited in the claim, and in the absence of such recitation no such intent is present. For example, as an aid to understanding, the following appended claims may contain usage of the introductory phrases "at least one" and "one or more" to introduce claim recitations. However, the use of such phrases should not be construed to imply that the introduction of a claim recitation by the indefinite articles "a" or "an" limits any particular claim containing such introduced claim recitation to embodiments containing only one such recitation, even when the same claim includes the introductory phrases "one or more" or "at least one" and indefinite articles such as "a" or "an" (*e.g.*, "a" and/or "an" should be interpreted to mean "at least one" or "one or more"); the same holds true for the use of definite articles used to introduce claim

recitations. In addition, even if a specific number of an introduced claim recitation is explicitly recited, those skilled in the art will recognize that such recitation should be interpreted to mean at least the recited number (*e.g.*, the bare recitation of "two recitations," without other modifiers, means at least two recitations, or two or more recitations).

**[0077]** Furthermore, in those instances where a convention analogous to "at least one of A, B, and C, etc." is used, in general such a construction is intended in the sense one having skill in the art would understand the convention (*e.g.*, " a system having at least one of A, B, and C" would include but not be limited to systems that have A alone, B alone, C alone, A and B together, A and C together, B and C together, and/or A, B, and C together, etc.). It will be further understood by those within the art that virtually any disjunctive word and/or phrase presenting two or more alternative terms, whether in the description, claims, or drawings, should be understood to contemplate the possibilities of including one of the terms, either of the terms, or both terms. For example, the phrase "A or B" will be understood to include the possibilities of "A" or "B" or "A and B."

**[0078]** As will be understood by one skilled in the art, for any and all purposes, such as in terms of providing a written description, all ranges disclosed herein also encompass any and all possible subranges and combinations of subranges thereof. Any listed range can be easily recognized as sufficiently describing and enabling the same range being broken down into at least equal halves, thirds, quarters, fifths, tenths, etc. As a non-limiting example, each range discussed herein can be readily broken down into a lower third, middle third and upper third, etc. As will also be understood by one skilled in the art all language such as "up to," "at least," "greater than," "less than," and the like include the number recited and refer to ranges which can be subsequently broken down into subranges as discussed above. Finally, as will be understood by one skilled in the art, a range includes each individual member. Thus, for example, a group having 1-3 cells refers to groups having 1, 2, or 3 cells. Similarly, a group having 1-5 cells refers to groups having 1, 2, 3, 4, or 5 cells, and so forth.

**[0079]** While various aspects and embodiments have been disclosed herein, other aspects and embodiments will be apparent to those skilled in the art. The various aspects and embodiments disclosed herein are for purposes of illustration and are not intended to be limiting, with the true scope and spirit being indicated by the following claims.

# CLAIMS

WHAT IS CLAIMED IS:

1. A method to generate shared secrets between a server and a client, the method comprising:

transmitting, by the server, a plurality of encrypted secrets corresponding to a plurality of data blocks to the client, the plurality of encrypted secrets generated by encrypting each secret with a respective data block in the plurality of data blocks;

recovering, by the client, a first subset of the plurality of secrets from the encrypted secrets, wherein the first subset of secrets corresponds to a first subset of data blocks stored at the client and the first subset of data blocks is a subset of the plurality of data blocks;

encrypting a message at the client using the first subset of secrets;

transmitting, by the client, the message to the server; and

recovering, by the server, the message using a second subset of the plurality of secrets, wherein the second subset of secrets corresponds to a second subset of data blocks known by the server to be previously stored at the client and the second subset of data blocks is a subset of the plurality of data blocks.

2. The method of claim 1, further comprising:

transmitting, by the server, a plurality of identifiers to the client, each identifier associated with a respective encrypted secret in the plurality of encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret; and

recovering, by the client, the first subset of secrets by using the plurality of identifiers to identify the encrypted secrets corresponding to the first subset of data blocks.

3. The method of claim 1, wherein the plurality of secrets includes a random number.

4. The method of claim 1, wherein:

the plurality of secrets includes a third subset of secrets corresponding to a third subset of data blocks known by the server to not be stored at the client; and

the third subset of data blocks is a subset of the plurality of data blocks.

5. The method of claim 1, further comprising:

encrypting, by the client, the message using every secret in the first subset of secrets;

transmitting, by the client, an indicator of a number of secrets in the first subset of secrets to the server; and

recovering, by the server, the message using a permutation of data blocks in the second subset of data blocks based on the indicator.

6. The method of claim 1, wherein the message includes at least one of a handshake, login information, and at least one secret from the first subset of secrets.

7. The method of claim 1, further comprising establishing a secondary secret shared between the server and the client based on the message.

8. The method of claim 1, wherein the first subset of data blocks includes data cached at the client for application acceleration.

9. A method for a server to generate shared secrets with a client, the method comprising:

generating a plurality of encrypted secrets corresponding to a plurality of data blocks by encrypting each secret in a plurality of secrets with a respective data block in the plurality of data blocks;

transmitting the plurality of encrypted secrets to the client;

receiving, from the client, a message encrypted using at least one secret from the plurality of secrets; and

recovering the message using a first subset of the plurality of secrets, wherein the first subset of secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the plurality of data blocks.

10. The method of claim 9, further comprising:

transmitting a plurality of identifiers to the client, each identifier associated with a respective encrypted secret in the plurality of encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret.

11. The method of claim 9, further comprising:

receiving, from the client, an indicator of a number of secrets used to encrypt the message; and

recovering the message using a permutation of data blocks in the first subset of data blocks based on the indicator.

12. A server configured to generate shared secrets with a client, the server comprising:

a memory configured to store instructions and a plurality of data blocks; and

a processing module configured to:

generate a plurality of encrypted secrets corresponding to the plurality of data blocks by encrypting each secret in a plurality of secrets using a respective data block in the plurality of data blocks;

transmit the plurality of encrypted secrets to the client;

receive, from the client, a message encrypted using at least one secret from the plurality of secrets; and

recover the message using a first subset of the plurality of secrets, wherein the first subset of secrets corresponds to a first subset of data blocks known by the server to be previously stored at the client and the first subset of data blocks is a subset of the plurality of data blocks.

13. The server of claim 12, wherein the processing module is further configured to:

transmit a plurality of identifiers to the client, each identifier associated with a respective encrypted secret in the plurality of encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret.

14. The server of claim 12, wherein the plurality of secrets includes a random number.

15. The server of claim 12, wherein:

the plurality of secrets includes a second subset of secrets corresponding to a second subset of data blocks known by the server to not be stored at the client; and

the second subset of data blocks is a subset of the plurality of data blocks.

16. The server of claim 12, wherein the processing module is further configured to:

receive, from the client, an indicator of a number of secrets used to encrypt the message; and

recover the message using a permutation of data blocks in the first subset of data blocks based on the indicator.

17. The server of claim 12, wherein the message includes at least one of a handshake, login information, and at least one secret from the plurality of secrets.

18. The server of claim 12, wherein the processing module is further configured to establish a secondary secret shared between the server and the client based on the message.

19. The server of claim 12, wherein the first subset of data blocks includes data cached at the client for application acceleration.

20. A method for a client to use shared secrets generated with a server, the method comprising:

receiving, from the server, a plurality of encrypted secrets corresponding to the plurality of data blocks;

recovering a first subset of secrets from the encrypted secrets, wherein the first subset of secrets corresponds to the first subset of data blocks;

encrypting a message using the first subset of secrets; and

transmitting the message to the server.

21. The method of claim 20, further comprising:

receiving, from the server, a plurality of identifiers, each identifier associated with a respective encrypted secret in the plurality of encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret; and

recovering the first subset of secrets by using the plurality of identifiers to identify the encrypted secrets corresponding to the first subset of data blocks.

22. The method of claim 20, further comprising:

encrypting the message using every secret in the first subset of secrets; and

transmitting an indicator of a number of secrets in the first subset of secrets to the server.

23. A client configured to generate shared secrets with a server, the client comprising:

a memory configured to store instructions and a first subset of data blocks, wherein the first subset of data blocks is a subset of a plurality of data blocks; and

a processing module configured to:

receive, from the server, a plurality of encrypted secrets corresponding to the plurality of data blocks;

recover a first subset of secrets from the encrypted secrets, wherein the first subset of secrets corresponds to the first subset of data blocks;

encrypt a message using the first subset of secrets; and

transmit the message to the server.

24. The client of claim 23, wherein the processing module is further configured to:

receive, from the server, a plurality of identifiers, each identifier associated with a respective encrypted secret in the plurality of encrypted secrets and identifying the respective data block used to encrypt the respective encrypted secret; and

recover the first subset of secrets by using the plurality of identifiers to identify the encrypted secrets corresponding to the first subset of data blocks.

25. The client of claim 23, wherein the first subset of secrets includes a random number.

26. The client of claim 23, wherein the processing module is further configured to:

encrypt the message using every secret in the first subset of secrets; and

transmit an indicator of a number of secrets in the first subset of secrets to the server.

27. The client of claim 23, wherein the message includes at least one of a handshake, login information, and at least one secret from the first subset of secrets.

28. The client of claim 23, wherein the processing module is further configured to establish a secondary secret shared between the server and the client based on the message.

29. The client of claim 23, wherein the first subset of data blocks includes data cached at the client for application acceleration.

30. A computer readable storage medium with instructions stored thereon, which when executed on one or more computing devices execute a method to generate shared secrets between a server and a client, wherein the method includes action of claims 1 through 8, 9 through 11, or 20 through 22.

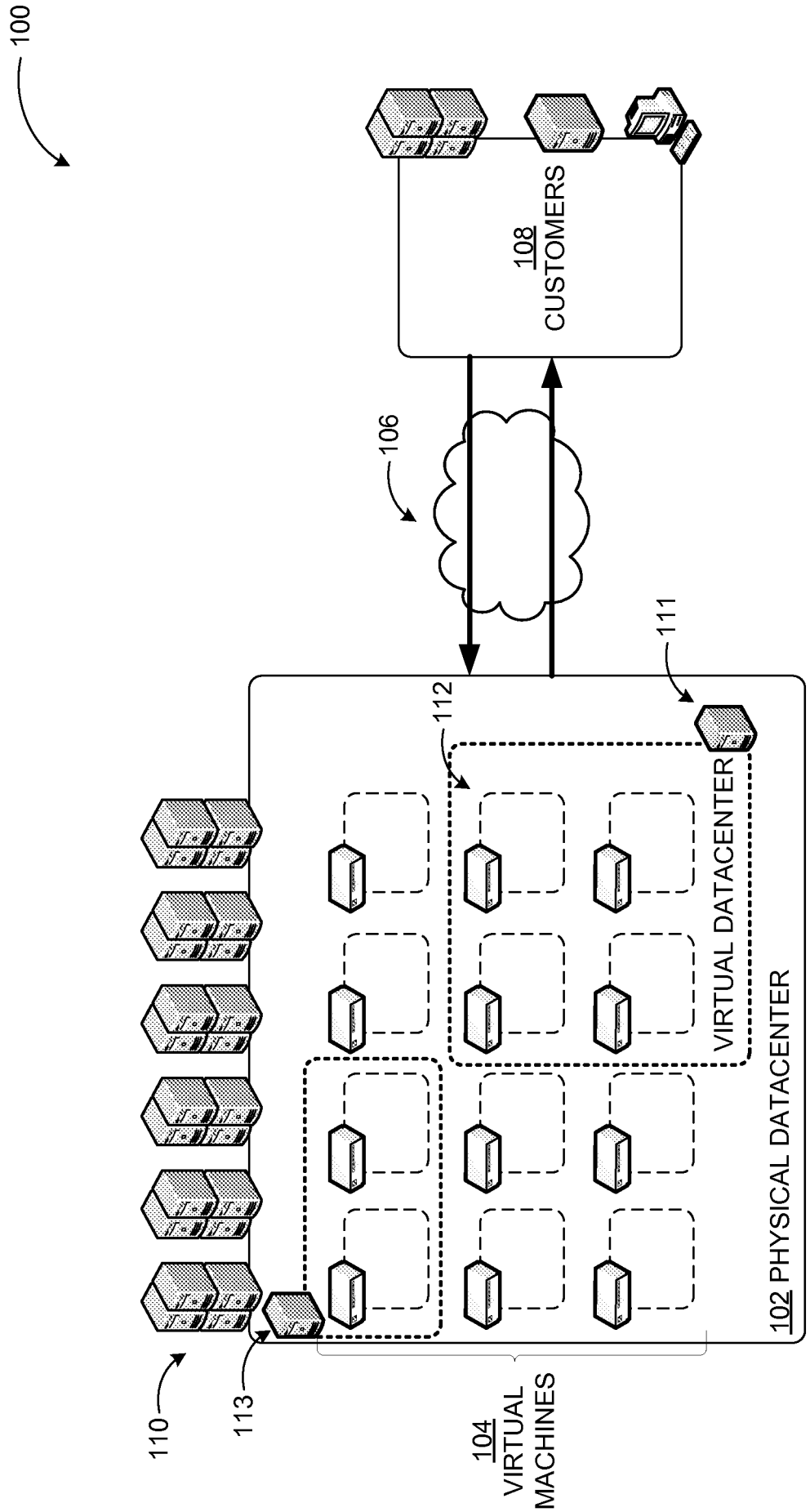
100
108 CUSTOMERS
106
111
112
VIRTUAL DATACENTER
102 PHYSICAL DATACENTER
110
113
104 VIRTUAL MACHINES

FIG. 1

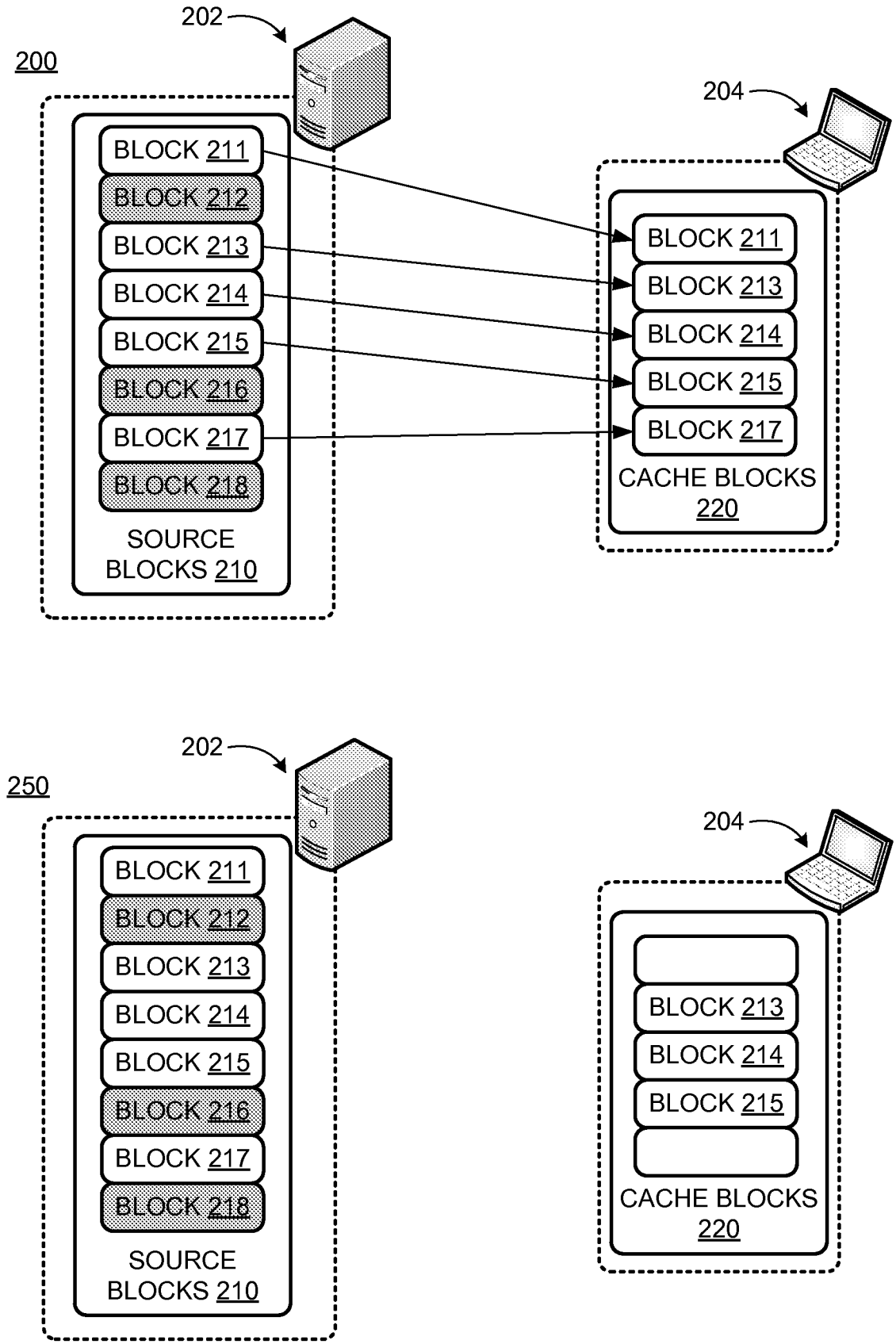
202
200
204
BLOCK 211
BLOCK 212
BLOCK 213
BLOCK 214
BLOCK 215
BLOCK 216
BLOCK 217
BLOCK 218
SOURCE BLOCKS 210
BLOCK 211
BLOCK 213
BLOCK 214
BLOCK 215
BLOCK 217
CACHE BLOCKS 220
202
250
204
BLOCK 211
BLOCK 212
BLOCK 213
BLOCK 214
BLOCK 215
BLOCK 216
BLOCK 217
BLOCK 218
SOURCE BLOCKS 210
BLOCK 213
BLOCK 214
BLOCK 215
CACHE BLOCKS 220

FIG. 2

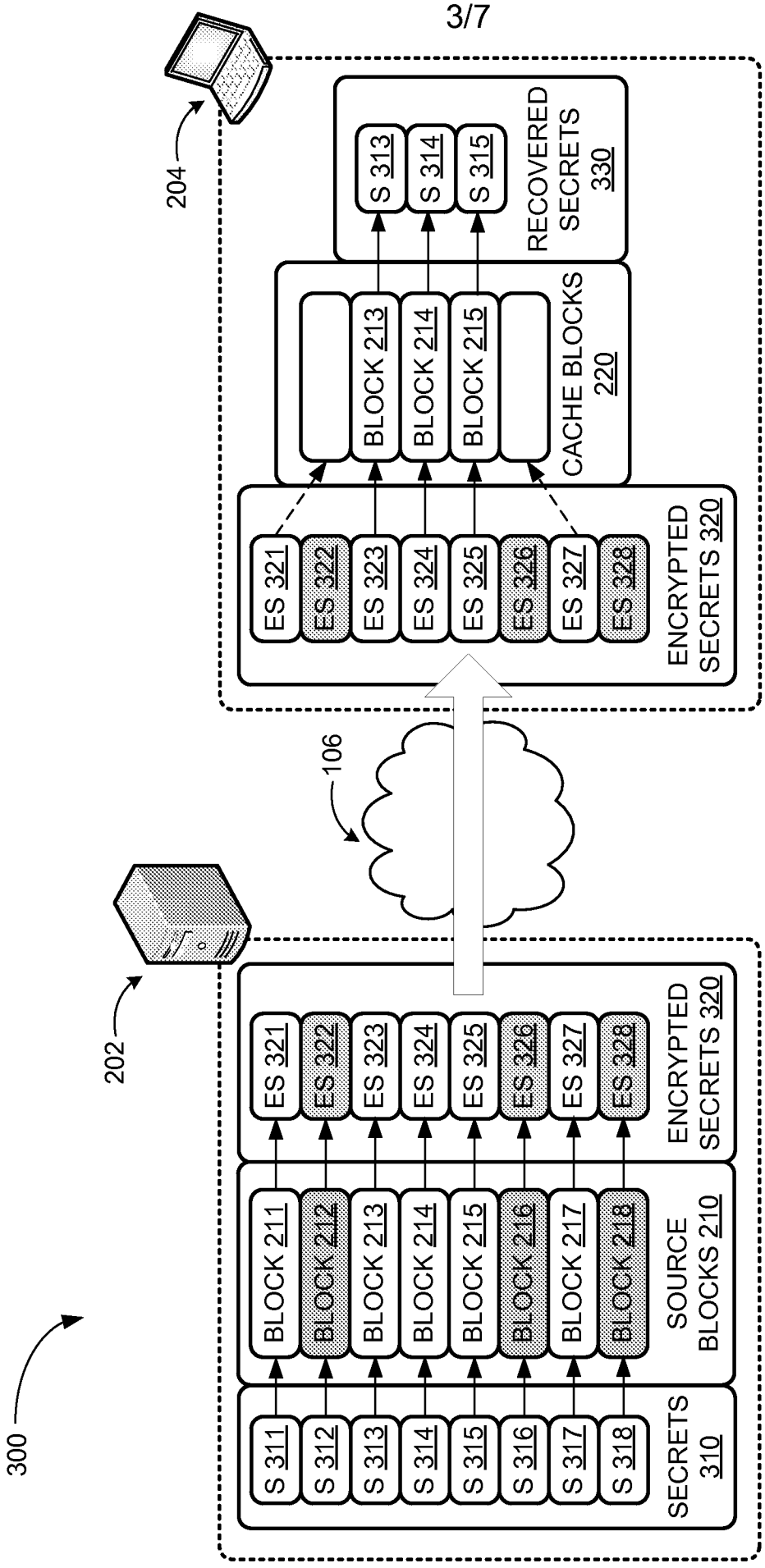
300
202
204
106
SECRETS 310
S 311
S 312
S 313
S 314
S 315
S 316
S 317
S 318
SOURCE BLOCKS 210
BLOCK 211
BLOCK 212
BLOCK 213
BLOCK 214
BLOCK 215
BLOCK 216
BLOCK 217
BLOCK 218
ENCRYPTED SECRETS 320
ES 321
ES 322
ES 323
ES 324
ES 325
ES 326
ES 327
ES 328
CACHE BLOCKS 220
BLOCK 213
BLOCK 214
BLOCK 215
RECOVERED SECRETS 330
S 313
S 314
S 315

FIG. 3

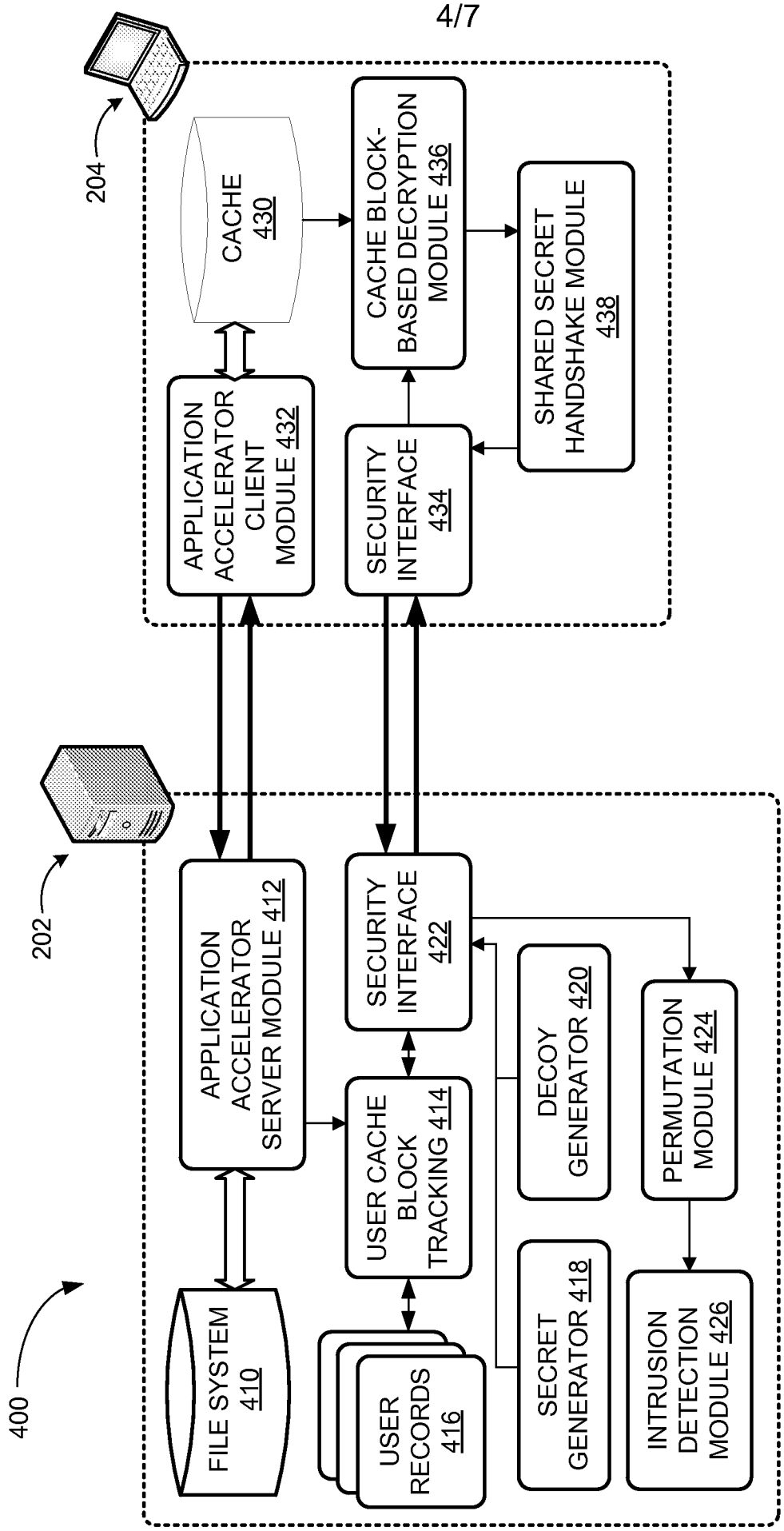
400
202
204
FILE SYSTEM 410
APPLICATION ACCELERATOR SERVER MODULE 412
USER CACHE BLOCK TRACKING 414
USER RECORDS 416
SECRET GENERATOR 418
DECOY GENERATOR 420
SECURITY INTERFACE 422
PERMUTATION MODULE 424
INTRUSION DETECTION MODULE 426
APPLICATION ACCELERATOR CLIENT MODULE 432
CACHE 430
CACHE BLOCK-BASED DECRYPTION MODULE 436
SECURITY INTERFACE 434
SHARED SECRET HANDSHAKE MODULE 438

FIG. 4

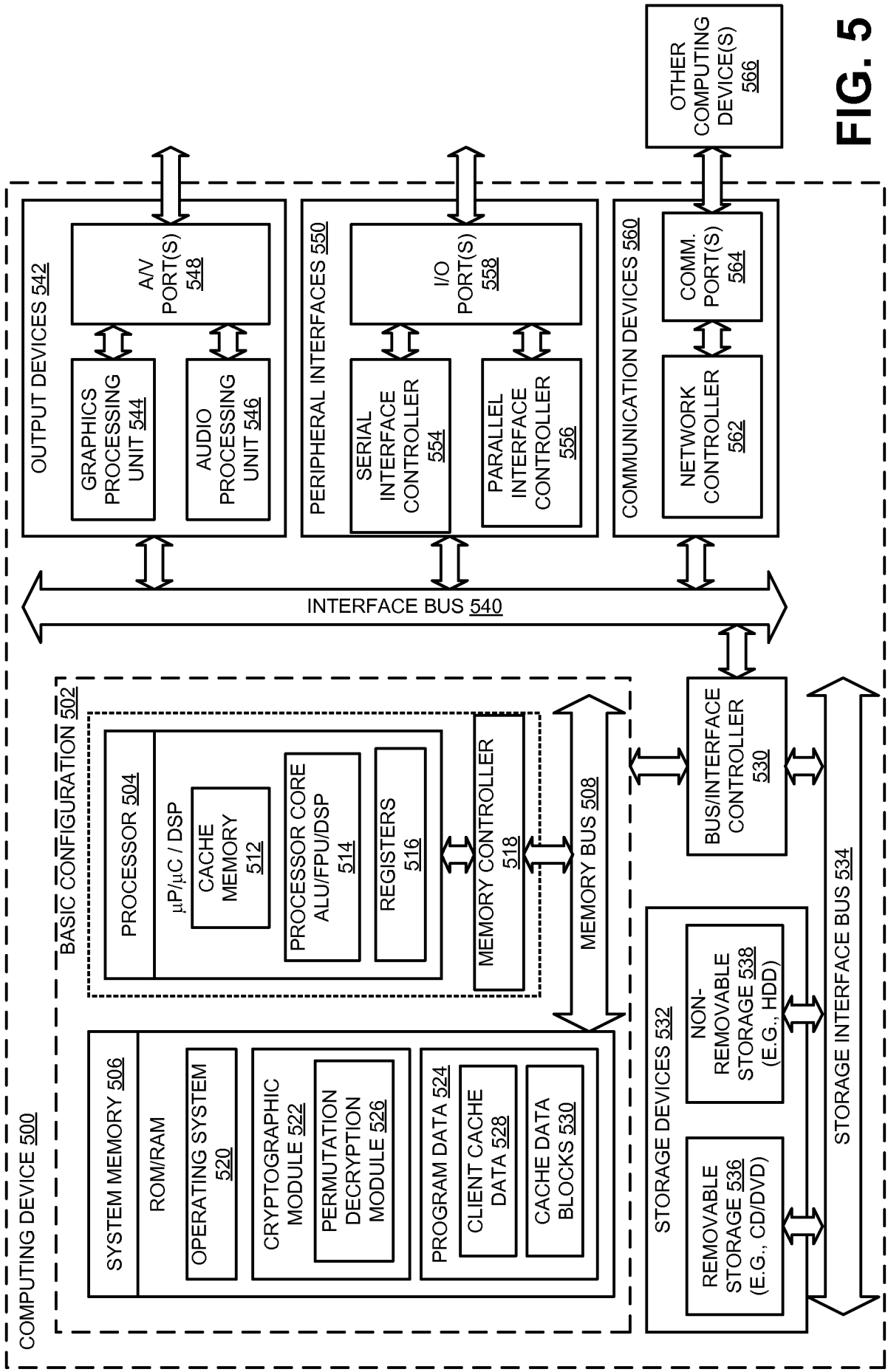
COMPUTING DEVICE 500
OUTPUT DEVICES 542
A/V PORT(S) 548
GRAPHICS PROCESSING UNIT 544
AUDIO PROCESSING UNIT 546
PERIPHERAL INTERFACES 550
I/O PORT(S) 558
SERIAL INTERFACE CONTROLLER 554
PARALLEL INTERFACE CONTROLLER 556
COMMUNICATION DEVICES 560
COMM. PORT(S) 564
NETWORK CONTROLLER 562
OTHER COMPUTING DEVICE(S) 566
INTERFACE BUS 540
BASIC CONFIGURATION 502
PROCESSOR 504
μP/μC / DSP
CACHE MEMORY 512
PROCESSOR CORE ALU/FPU/DSP 514
REGISTERS 516
MEMORY CONTROLLER 518
MEMORY BUS 508
BUS/INTERFACE CONTROLLER 530
SYSTEM MEMORY 506
ROM/RAM
OPERATING SYSTEM 520
CRYPTOGRAPHIC MODULE 522
PERMUTATION DECRYPTION MODULE 526
PROGRAM DATA 524
CLIENT CACHE DATA 528
CACHE DATA BLOCKS 530
STORAGE DEVICES 532
NON-REMOVABLE STORAGE 538 (E.G., HDD)
REMOVABLE STORAGE 536 (E.G., CD/DVD)
STORAGE INTERFACE BUS 534

FIG. 5

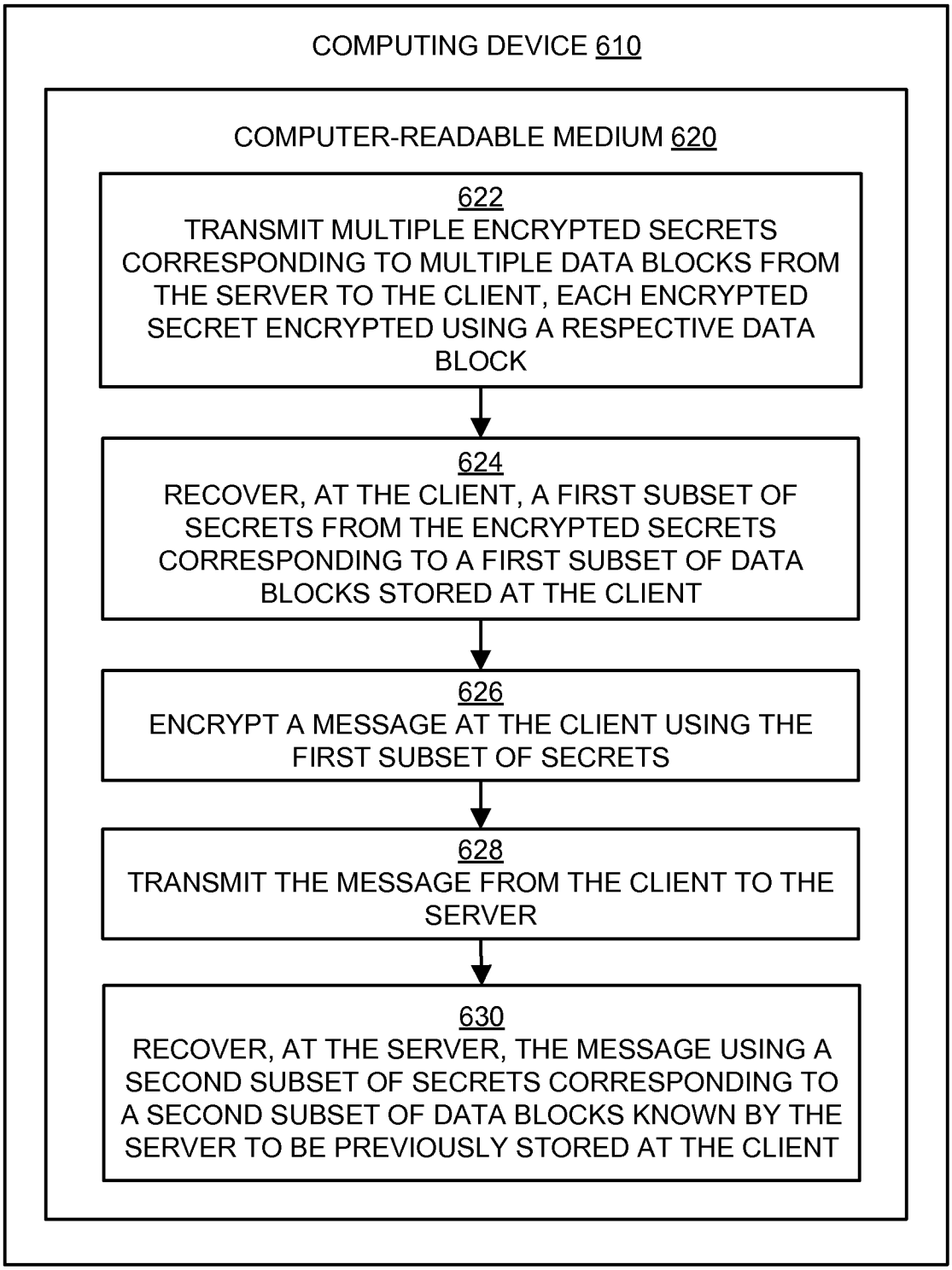
COMPUTING DEVICE 610
COMPUTER-READABLE MEDIUM 620
622
TRANSMIT MULTIPLE ENCRYPTED SECRETS CORRESPONDING TO MULTIPLE DATA BLOCKS FROM THE SERVER TO THE CLIENT, EACH ENCRYPTED SECRET ENCRYPTED USING A RESPECTIVE DATA BLOCK
624
RECOVER, AT THE CLIENT, A FIRST SUBSET OF SECRETS FROM THE ENCRYPTED SECRETS CORRESPONDING TO A FIRST SUBSET OF DATA BLOCKS STORED AT THE CLIENT
626
ENCRYPT A MESSAGE AT THE CLIENT USING THE FIRST SUBSET OF SECRETS
628
TRANSMIT THE MESSAGE FROM THE CLIENT TO THE SERVER
630
RECOVER, AT THE SERVER, THE MESSAGE USING A SECOND SUBSET OF SECRETS CORRESPONDING TO A SECOND SUBSET OF DATA BLOCKS KNOWN BY THE SERVER TO BE PREVIOUSLY STORED AT THE CLIENT

FIG. 6

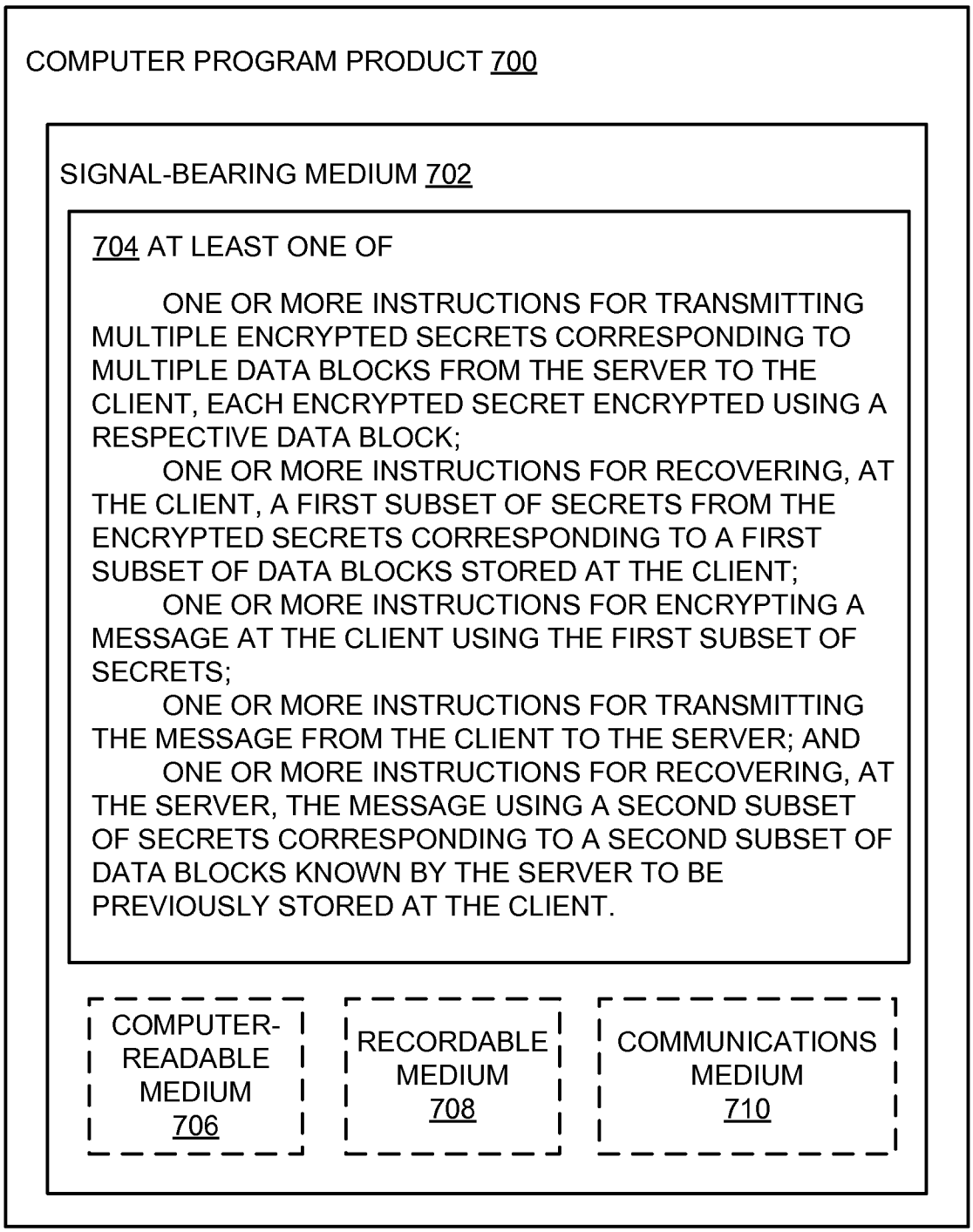
COMPUTER PROGRAM PRODUCT 700
SIGNAL-BEARING MEDIUM 702
704 AT LEAST ONE OF
ONE OR MORE INSTRUCTIONS FOR TRANSMITTING MULTIPLE ENCRYPTED SECRETS CORRESPONDING TO MULTIPLE DATA BLOCKS FROM THE SERVER TO THE CLIENT, EACH ENCRYPTED SECRET ENCRYPTED USING A RESPECTIVE DATA BLOCK;
ONE OR MORE INSTRUCTIONS FOR RECOVERING, AT THE CLIENT, A FIRST SUBSET OF SECRETS FROM THE ENCRYPTED SECRETS CORRESPONDING TO A FIRST SUBSET OF DATA BLOCKS STORED AT THE CLIENT;
ONE OR MORE INSTRUCTIONS FOR ENCRYPTING A MESSAGE AT THE CLIENT USING THE FIRST SUBSET OF SECRETS;
ONE OR MORE INSTRUCTIONS FOR TRANSMITTING THE MESSAGE FROM THE CLIENT TO THE SERVER; AND
ONE OR MORE INSTRUCTIONS FOR RECOVERING, AT THE SERVER, THE MESSAGE USING A SECOND SUBSET OF SECRETS CORRESPONDING TO A SECOND SUBSET OF DATA BLOCKS KNOWN BY THE SERVER TO BE PREVIOUSLY STORED AT THE CLIENT.
COMPUTER-READABLE MEDIUM 706
RECORDABLE MEDIUM 708
COMMUNICATIONS MEDIUM 710

FIG. 7

WO 2015/119610 PCT/US2014/015058

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 14/15058

**Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)**

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☒ Claims Nos.: 30
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

**Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)**

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

**Remark on Protest**

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 14/15058

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04L 29/06 (2014.01)
USPC - 713/171
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)
IPC (8) - H04L 29/06 (2014.01)
USPC - 713/171

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC - 380/285, 380/44, 380/282, 380/30, 380/259, 713/171, 713/151, 713/156, 713/176, 713/169, 713/175, 726/3, 713/170, 726/2, 713/168 (See keywords Below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
Thomsoninnovation.com; Patbase; Google Scholar; Google Patents; Gogole.com; Freepatentsonline; ProQuest Dialog
Search Terms: Secret, key, session, share, exchange, mutual, agreed, plurality, many, data block, known, cache, transmit, encrypted, hash, recover, resolve, regenerate, message, server, client, random number, subset, associate, assign, ma

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | US 2010/0306525 A1 (FERGUSON), 02 December 2010 (02.12.2010), entire document, especially Abstract; para [0004], [0024], [0025], [0031], [0045]-[0047] | 1-29 |
| Y | US 2013/0227292 A1 (SUFFLING), 29 August 2013 (29.08.2013), entire document, especially Abstract; para [0027]-[0029], [0035], [0037], [0039]-[0041], [0073]-[0075], [0097]-[0099] | 1-29 |
| Y | US 2009/0144546 A1 (JANCULA et al.), 04 June 2009 (04.06.2009), entire document, especially Abstract; para [0010], [0025]-[0027] | 8, 19 and 29 |
| A | US 2011/0231650 A1 (COULIER), 22 September 2011 (22.09.2011), entire document | 1-29 |
| A | US 6,182,220 B1 (Chen et al.), 30 January 2001 (30.01.2001), entire document | 1-29 |

☐ Further documents are listed in the continuation of Box C. ☐

* 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 application or patent 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 |
|---|---|
| 02 June 2014 (02.06.2014) | 08 JUL 2014 |
| Name and mailing address of the ISA/US<br>Mail Stop PCT, Attn: ISA/US, Commissioner for Patents<br>P.O. Box 1450, Alexandria, Virginia 22313-1450<br>Facsimile No. 571-273-3201 | Authorized officer:<br>Lee W. Young<br>PCT Helpdesk: 571-272-4300<br>PCT OSP: 571-272-7774 |

Form PCT/ISA/210 (second sheet) (July 2009)