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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
1 September 2016 (01.09.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/137464 A1

(51) International Patent Classification:
G06F 11/10 (2006.01) *G06F 12/06* (2006.01)

(21) International Application Number:
PCT/US2015/017676

(22) International Filing Date:
26 February 2015 (26.02.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventor: **BALLARD, Curtis C.**; 3404 E Harmony Rd., Ft. Collins, Colorado 80528-9544 (US).

(74) Agents: **SU, Benjamin** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 80528 (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, 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).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

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

(54) Title: CHECKSUM REQUESTS OF FILES

(57) Abstract: An example data storage device includes a communication interface to receive a checksum request from a computing device. The checksum request includes file segment information of a file and an expected checksum of the file. The file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment. The data storage device also includes a checksum calculation module to, in response to a reception of the checksum request, generate a computed checksum of the file using the file segment information. The data storage device further includes a checksum comparison module to compare the computed checksum to the expected checksum to generate a comparison result of the file. The checksum comparison module is also to transmit the comparison result to the computing device via the communication interface.

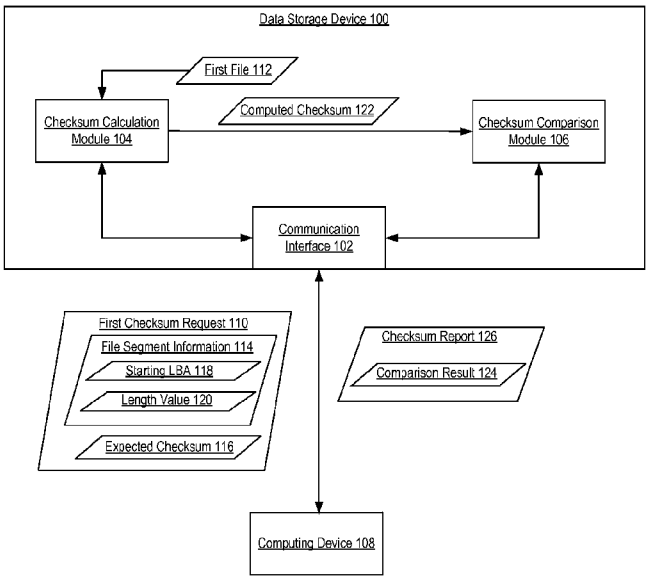

FIG. 1

WO 2016/137464 A1

CHECKSUM REQUESTS OF FILES

BACKGROUND

[0001] Data stored in a data storage device, such as a hard disk drive, may be susceptible to loss due to a variety of factors. For example, the data may be loss or corrupted due to a media failure of the storage device.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Some examples of the present application are described with respect to the following figures:

[0003] FIG. 1 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to an example;

[0004] FIG. 2 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to another example;

[0005] FIG. 3 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to another example;

[0006] FIG. 4 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to another example;

[0007] FIG. 5 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to another example;

[0008] FIG. 6 is a block diagram of a data storage device to generate a checksum of a file based on a checksum request, according to another example;

[0009] FIG. 7 is a block diagram of a computing device to generate a checksum request, according to an example;

[0010] FIG. 8 is a flowchart illustrating a method of operation at a data storage device to generate a checksum of a file based on a checksum request, according to an example;

[0011] FIG. 9 is a flowchart illustrating a method of operation at a data storage device to generate a checksum of a file based on a checksum request, according to another example; and

[0012] FIG. 10 is a flowchart illustrating a method of operation at a computing device to generate a checksum request, according to an example.

DETAILED DESCRIPTION

[0013] To verify the integrity of data stored in a data storage device, a computing device may read the data from the data storage device and generate a checksum of the data. The computing device may compare the checksum to a known good checksum of the data to verify the integrity of the data. However, when the data storage device is connected to the computing device via a network, such as the Internet, reading the data from the data storage device consumes network bandwidth. Thus, available network bandwidth may be reduced.

[0014] Examples described herein provide a data storage device that may include a communication interface to receive a checksum request from a computing device. The checksum request may include file segment information of a file and an expected checksum of the file. The file segment information may include a starting logical block address (LBA) of a segment of the file and a length value of the segment. The data storage device may also include a checksum calculation module to, in response to a reception of the checksum request generate a checksum of the segment using the file segment information. In some examples, the data storage device may further include a checksum comparison module to compare the checksum to the expected checksum to generate a comparison result. The checksum comparison module may also transmit the comparison result to the computing device via the communication interface. In some examples, the data storage device may transmit the checksum to the computing device via the communication interface. In this manner, examples described herein may reduce consumption of network bandwidth.

[0015] Referring now to the figures, FIG. 1 is a block diagram of a data storage device 100 to generate a checksum of a file based on a checksum

request, according to an example. Data storage device 100, for example, may be a hard disk drive, a tape drive, or any device that stores data via a storage medium. Data storage device 100 may include communication interface 102, a checksum calculation module 104, and a checksum comparison module 106.

[0016] Communication interface 102 may be a transceiver. In some examples, communication interface 102 may be implemented using a transceiver that is compliant with an Ethernet protocol, such as one from the Institute of Electrical and Electronics Engineers (IEEE) 802.11 family of protocols, or one from the IEEE 802.3 family of protocols. In some examples, communication interface 102 may be implemented using a transceiver that is compliant with an InterNational Committee for Information Technology Standards (INCITS) T10 protocol, such as the Fibre Channel Protocol-4 (FCP-4) for fibre channel storage devices. Communication interface 102 may be implemented using hardware, processor executable instructions, or a combination thereof.

[0017] Checksum calculation module 104 may be any device that calculates a checksum of data, such as data stored as a file. As used herein, a file may be a collection of data that is identified by a distinct file name. As used herein, a checksum may be a numerical value generated using the data as input to a checksum function. For example, a checksum function may be a hash function, such as the secure hash algorithm-1 (SHA-1). Checksum calculation module 104 may be implemented using hardware, processor executable instructions, or a combination thereof. In some examples, checksum calculation module 104 may be implemented as a circuit. In some examples, checksum calculation module 104 may be implemented using a controller, such as a field-programmable gate array (FPGA) and instructions executable by the controller.

[0018] Checksum comparison module 106 may be any device that compares the difference between multiple checksums. Checksum comparison module 106 may be implemented using hardware, processor executable instructions, or a combination thereof. In some examples, checksum comparison module 106 may be implemented as a circuit. In some examples, checksum comparison module

106 may be implemented using a controller, such as a FPGA and instructions executable by the controller.

[0019] During operation, a data storage device 100 may be connected to a computing device 108 to store data for computing device 108. In some example, data storage device 100 may be connected to computing device 108 via a network, such as the Internet or a local area network (LAN). Computing device 108 may transmit a first checksum request 110 associated with a first file 112 to data storage device 100 to initiate an integrity verification operation of first file 112. First checksum request 110 may be generated by a data storage management application (not shown in FIG. 1) executing on computing device 108. The data storage management application may be implemented using processor executable instructions.

[0020] First checksum request 110 may include file segment information 114 and an expected checksum 116. File segment information 114 may identify a range of logical block addresses (LBAs) where a segment of first file 112 is stored in a storage medium (not shown in FIG. 1) of data storage device 100. In some examples, first file 112 may include a single segment. Thus, file segment information 114 may identify a range of LBAs where first file 112 is stored. Checksum calculation of a file having a plurality of segments is described in more detail in FIG. 2.

[0021] File segment information 114 may include a starting LBA 118 and a length value 120. Starting LBA 118 may identify the beginning of the range of LBAs where first file 112 is stored. Length value 120 may indicate the number of LBAs in the range. For example, starting LBA 118 may indicate a LBA of 0001 and length value 120 may be a value of 2. Thus, first file 112 may be stored in a range of LBAs from 0001 to 0002. Expected checksum 116 may be a checksum of first file 112 that is generated from a known good copy of first file 112. Thus, expected checksum 116 may serve as a reference checksum for first file 112 to verify the integrity of first file 112.

[0022] Data storage device 100 may receive first checksum request 110 via communication interface 102. Communication interface 102 may transmit first checksum request 110 to checksum calculation module 104. In response to a reception of first checksum request 110, checksum calculation module 104 may generate/compute a checksum of first file 112 using file segment information 114. For example, checksum calculation module 104 may read first file 112 by reading data stored in the range of LBAs determined using starting LBA 118 and length value 120. Checksum calculation module 104 may generate a computed checksum 122 of first file 112 via a checksum function after reading first file 112.

[0023] Checksum calculation module 104 may transmit computed checksum 122 to checksum comparison module 106. Checksum comparison module 106 may compare computed checksum 122 to expected checksum 116 to generate a comparison result 124. Comparison result 124 may indicate whether computed checksum 122 matches expected checksum 116. In some examples, checksum comparison module 106 may receive expected checksum 116 from checksum calculation module 104. In some examples, checksum comparison module 106 may receive expected checksum 116 via communication interface 102 when communication interface 102 receives file segment information 114.

[0024] Checksum comparison module 106 may transmit comparison result 124 to computing device 108 via communication interface 102. In some examples, checksum comparison module 106 may transmit comparison result 124 in a checksum report 126. Computing device 108 may receive comparison result 124 and/or checksum report 126 from data storage device 100. Computing device 108 may determine an integrity of first file 112 using comparison result 124. For example, when comparison result 124 indicates that computed checksum 122 does not match expected checksum 116, computing device 108 may determine that the integrity of first file 112 has been compromised or corrupted. When comparison result 124 indicates that computed checksum 122 matches expected checksum 116, computing device 108 may determine that the integrity of first file 112 is not compromised or corrupted. Thus, by computing computed checksum 122 at data storage device

100 and transmitting comparison result 124 to computing device 108 instead of transmitting first file 112 to computing device 108, network bandwidth consumption associated with an integrity verification operation may be reduced.

[0025] FIG. 2 is a block diagram of data storage device 200 to generate a checksum of a file based on a checksum request, according to another example. Data storage device 200 may be similar to data storage device 100 of FIG. 1. Data storage device 200 may include communication interface 102, checksum calculation module 104, checksum comparison module 106, a storage medium 202, and a read mechanism. Storage medium 202 may be any electronic, magnetic, optical, or other physical storage device that contains or stores data, such as files. Thus, storage medium 202 may be, for example, a hard disk drive platter, a magnetic tape cartridge, etc. Read mechanism 204 may be any device that reads data from storage medium 202. For example, when storage medium 202 is a magnetic tape cartridge, read mechanism 204 may be a tape head. In some examples, a plurality of files may be stored in storage medium 202, such as first file 112 and a second file 206. In some examples, second file 206 may include a plurality of segments, such as a first segment and a second segment. Each of the plurality of segments of second file 206 may be stored at a distinct range of LBAs.

[0026] During operation, computing device 108 may transmit a second checksum request 208 associated with files 112 and 206 to data storage device 200 to initiate an integrity verification operation of files 112 and 206. Second checksum request 208 may include a file identifier 210 of first file 112, a file identifier 212 of second file 206, file segment information 114 of first file 112, expected checksum 116 of first file 112, file segment information 214 of second file 206, and an expected checksum 216 of second file 206. File segment information 214 may include a starting LBA 218 of the first segment of second file 206, a length value 220 of the first segment of second file 206, a starting LBA 222 of the second segment of second file 206, and a length value 224 of the second segment of second file 206.

[0027] File identifier 210 may include information to enable data storage device 100 to identify first file 112. For example, file identifier 210 may indicate that file segment information 114 and expected checksum 116 are associated with first file 112. In some examples, file identifier 210 may also include a file name of first file 112. File identifier 212 may include information to enable data storage device 100 to identify second file 206. For example, file identifier 212 may indicate that file segment information 214 and expected checksum 216 are associated with second file 206. File identifier 212 may indicate that starting LBA 218 and length value 220 are associated with the first segment of second file 206. File identifier 212 may also indicate that starting LBA 222 and length value 224 are associated with the second segment of second file 206. In some examples, file identifier 212 may also include a file name of second file 206.

[0028] Data storage device 200 may receive second checksum request 208 via communication interface 102. Communication interface 102 may transmit second checksum request 208 to checksum calculation module 104. In response to a reception of second checksum request 208, checksum calculation module 104 may generate distinct computed checksums for files 112 and 206 using second checksum request 208.

[0029] Checksum calculation module 104 may read first file 112 via read mechanism 204 using file identifier 210 and file segment information 114. Checksum calculation module 104 may read second file 206 via read mechanism 204 using file identifier 212 and file segment information 214. Checksum calculation module 104 may generate distinct computed checksums for files 112 and 206 after reading files 112 and 206. For example, checksum calculation module 104 may generate computed checksum 122 for file 112 as described in FIG. 1. Checksum calculation module 104 may generate a computed checksum 226 of second file 206. To generate/compute computed checksum 226, checksum calculation module 104 may add a first segment checksum of the first segment to a second segment checksum of the second segment.

[0030] Checksum calculation module 104 may generate the first segment checksum of the first segment after reading the first segment via read mechanism 204. For example, checksum calculation module 104 may determine a LBA range (i.e., starting LBA 218 and length value 220) of the first segment via file identifier 212. Checksum calculation module 104 may read the first segment by reading the LBA range via read mechanism 204. Checksum calculation module 104 may apply a checksum function to the first segment to generate the first segment checksum.

[0031] Checksum calculation module 104 may generate the second segment checksum of the second segment after reading the second segment via read mechanism 204. For example, checksum calculation module 104 may determine a LBA range (i.e., starting LBA 222 and length value 224) of the second segment via file identifier 212. Checksum calculation module 104 may read the second segment by reading the LBA range via read mechanism 204. Checksum calculation module 104 may apply a checksum function to the second segment to generate the second segment checksum.

[0032] After generating computed checksums 122 and 226, checksum calculation module 104 may transmit computed checksums 122 and 226 to checksum comparison module 106. Checksum comparison module 106 may generate comparison result 124 of first file 112 as described in FIG. 1. Checksum comparison module 106 may also generate a comparison result 228 of second file 206 by comparing computed checksum 226 to expected checksum 216. Checksum comparison module 106 may transmit a checksum report 230 that includes comparison results 124 and 228 to computing device 108 via communication interface 102. In some examples, checksum report 230 may indicate that comparison result 124 is for first file 112 and comparison result 228 is for second file 206. In some examples, checksum comparison module 106 may transmit comparison results 124 and 228 to computing device 108 separately.

[0033] Computing device 108 may determine integrities of files 112 and 206 via comparison results 124 and 228, respectively. In some example, in response to a determination that a file is corrupted, computing device 108 may transmit a replacement copy of the corrupted file to data storage device 200 so that the corrupted file can be replaced. For example, computing device 108 may determine that first file 112 is corrupted via comparison result 124. Computing device 108 may transmit a replacement copy 232 of first file 112 to data storage device 200 so that first file 112 may be replaced. For example, first file 112 may be erased from storage medium 202 and rewritten using replacement copy 232 via a write mechanism (not shown in FIG. 2). In some examples, computing device 108 may transmit a write instruction (not shown in FIG. 2) to data storage device 200 so data storage device 200 may overwrite first file 112 with a new copy.

[0034] FIG. 3 is a block diagram of a data storage device 300 to generate a checksum of a file based on a checksum request, according to another example. Data storage device 300 may be similar to data storage device 100 of FIG. 1. Data storage device 300 may include communication interface 102 and a checksum calculation module 302. Checksum calculation module 302 may be similar to checksum calculation module 104 of FIG. 1. Data storage device 300 may operate similar to data storage device 100 to calculate a checksum for a file. However, unlike data storage device 100 in which a comparison result is transmitted back to computing device 108, data storage device 300 may transmit a checksum of a file to computing device 108.

[0035] For example, data storage device 300 may receive a first checksum request 304 associated with first file 112 from computing device 108. First checksum request 304 may include file segment information 114. Checksum calculation module 302 may generate computed checksum 122 of first file 112 based on first checksum request 304. Checksum calculation module 302 may transmit computed checksum 122 to computing device 108 via communication interface 102. In some examples, checksum calculation module 302 may transmit computed checksum 122 in a checksum report 306 to computing device

108 via communication interface 102. Computing device 108 may determine the integrity of first file 112 using computed checksum 122.

[0036] FIG. 4 is a block diagram of a data storage device 400 to generate a checksum of a file based on a checksum request, according to another example. Data storage device 400 may be similar to data storage device 200 of FIG. 2. Data storage device 400 may include communication interface 102, checksum calculation module 302, storage medium 202, and read mechanism 204. Data storage device 400 may operate similar to data storage device 200 to calculate a checksum for a file. However, unlike data storage device 200 in which a comparison result is transmitted back to computing device 108, data storage device 400 may transmit a checksum of a file to computing device 108.

[0037] For example, data storage device 400 may receive a second checksum request 402 associated with files 112 and 206 from computing device 108. Second checksum request 402 may include file segment information 114, file identifiers 210 and 212, and file segment information 214. Based on second checksum request 402, checksum calculation module 302 may read files 112 and 206 from storage medium 202 via read mechanism 204. Checksum calculation module 302 may generate computed checksum 122 of first file 112 and computed checksum 226 of second file 206 based on second checksum request 402. Checksum calculation module 302 may transmit a checksum report 404 that includes computed checksums 122 and 226 to computing device 108 via communication interface 102. Computing device 108 may determine the integrities of files 112 and 206 using checksum report 404.

[0038] FIG. 5 is a block diagram of a data storage device 500 to generate a checksum of a file based on a checksum request, according to another example. Data storage device 500 may implement data storage device 100 of FIG. 1 and/or data storage device 200 of FIG. 2. Data storage device 500 may include a processor 502 and a computer-readable storage medium 504.

[0039] Processor 502 may be a central processing unit (CPU), a semiconductor-based microprocessor, and/or other hardware devices suitable for

retrieval and execution of instructions stored in computer-readable storage medium 504. Processor 502 may fetch, decode, and execute instructions 506, 508, 510, and and 512 to control a process of generating a checksum of a file. As an alternative or in addition to retrieving and executing instructions, processor 502 may include at least one electronic circuit that includes electronic components for performing the functionality of instructions 506, 508, 510, 512 or a combination thereof.

[0040] Computer-readable storage medium 504 may be any electronic, magnetic, optical, or other physical storage device that contains or stores executable instructions. Thus, computer-readable storage medium 504 may be, for example, Random Access Memory (RAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage device, an optical disc, etc. In some examples, computer-readable storage medium 504 may be a non-transitory storage medium, where the term "non-transitory" does not encompass transitory propagating signals. As described in detail below, computer-readable storage medium 504 may be encoded with a series of processor executable instructions 506, 508, 510, and 512 for receiving a checksum request, generating a computed checksum of a file, comparing the computed checksum to an expected checksum to generate a comparison result, and transmitting the comparison result to a computing device.

[0041] Checksum request reception instructions 506 may receive a checksum request from a computing device. For example, referring to FIG. 1, data storage device 100 may receive first checksum request 110 via communication interface 102. Checksum generation instructions 508 may generate a computed checksum of a file associated with a checksum request using the checksum request. For example, referring to FIG. 1, in response to a reception of first checksum request 110, checksum calculation module 104 may generate/compute a checksum of first file 112 using file segment information 114.

[0042] Checksum comparison instructions 510 may compare a computed checksum of a file to an expected checksum of the file. For example, referring to FIG. 1, checksum comparison module 106 may compare computed checksum

122 to expected checksum 116 to generate a comparison result 124. Comparison result transmission instructions 512 may transmit a comparison result to a computing device. For example, referring to FIG. 1, checksum comparison module 106 may transmit comparison result 124 to computing device 108 via communication interface 102.

[0043] FIG. 6 is a block diagram of a data storage device 600 to generate a checksum of a file based on a checksum request, according to another example. Data storage device 600 may implement data storage device 300 of FIG. 3 and/or data storage device 400 of FIG. 4. Data storage device 600 may include a processor 602 and a computer-readable storage medium 604. Processor 602 may be similar to processor 502 of FIG. 5. Computer-readable storage medium 604 may be similar to computer-readable storage medium 504 of FIG. 5. Computer-readable storage medium 604 may be encoded with instructions 506, 508, and 606. Checksum transmission instructions 606 may transmit a computed checksum to a computing device. For example, referring to FIG. 3, checksum calculation module 302 may transmit computed checksum 122 to computing device 108 via communication interface 102.

[0044] FIG. 7 is a block diagram of a computing device 700 to generate a checksum request, according to an example. Computing device 700 may implement computing device 108 of FIGs. 1-4. Computing device 700 may include a processor 702 and a computer-readable storage medium 704. Processor 702 may be similar to processor 502 of FIG. 5. Computer-readable storage medium 704 may be similar to computer-readable storage medium 504.

[0045] Computer-readable storage medium 704 may be encoded with a series of instructions 706, 708, and 710 to control a process of transmitting a checksum request associated with a file to a data storage device, receiving a checksum report associated with the file, and determining an integrity of the file using the checksum report.

[0046] Checksum request transmission instructions 706 may transmit a checksum request to a data storage device. For example, referring to FIG. 1,

computing device 108 may transmit a first checksum request 110 associated with a first file 112 to data storage device 100 to initiate an integrity verification operation of first file 112. Checksum report reception instructions 708 may receive a checksum report from the data storage device. For example, referring to FIG. 1, computing device 108 may receive comparison result 124 and/or checksum report 126 from data storage device 100. File integrity determination instructions 710 may determine an integrity of a file using the checksum report. For example, referring to FIG. 1, computing device 108 may determine an integrity of first file 112 using comparison result 124.

[0047] FIG. 8 is a flowchart illustrating a method 800 of operation at a data storage device to generate a checksum of a file based on a checksum request, according to an example. Method 800 may be implemented using data storage device 100 of FIG. 1, data storage device 200 of FIG. 2, and/or data storage device 600 of FIG. 6.

[0048] Method 800 may include receiving, at a data storage device, a checksum request from a computing device, where the checksum request includes file segment information of a file and an expected checksum of the file, and where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment, at 802. For example, referring to FIG. 1, data storage device 100 may receive first checksum request 110 via communication interface 102.

[0049] Method 800 may also include, in response to a reception of the checksum request, generating, via a checksum calculation module of the data storage device, a computed checksum of the file using the file segment information, at 804. For example, referring to FIG. 1, in response to a reception of first checksum request 110, checksum calculation module 104 may generate/compute a checksum of first file 112 using file segment information 114.

[0050] Method 800 may further include comparing, via a checksum comparison module of the data storage device, the computed checksum to the expected checksum to generate a comparison result of the file, at 806. For

example, referring to FIG. 1, checksum comparison module 106 may compare computed checksum 122 to expected checksum 116 to generate a comparison result 124. Method 800 may further include transmitting the comparison result to the computing device via a communication interface of the data storage device, at 808. For example, referring to FIG. 1, checksum comparison module 106 may transmit comparison result 124 to computing device 108 via communication interface 102.

[0051] FIG. 9 is a flowchart illustrating a method 900 of operation at a data storage device to generate a checksum of a file based on a checksum request, according to another example. Method 900 may be implemented using data storage device 300 of FIG. 3, data storage device 400 of FIG. 4, and/or data storage device 600 of FIG. 6.

[0052] Method 900 may include receiving, at a data storage device, a checksum request from a computing device, where the checksum request includes file segment information of a file and a file identifier of the file, and where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment, at 902. For example, referring to FIG. 4, data storage device 400 may receive a second checksum request 402 associated with files 112 and 206 from computing device 108. Second checksum request 402 may include file segment information 114, file identifiers 210 and 212, and file segment information 214. Method 900 may also include, in response to a reception of the checksum request, generating, via a checksum calculation module of the data storage device, a computed checksum of the file using the file segment information, at 904. For example, referring to FIG. 3, checksum calculation module 302 may generate computed checksum 122 of first file 112 based on first checksum request 304.

[0053] Method 900 may further include transmitting the computed checksum to the computing device via a communication interface of the data storage device, at 906. For example, referring to FIG. 3, checksum calculation module

302 may transmit computed checksum 122 to computing device 108 via communication interface 102.

[0054] FIG. 10 is a flowchart illustrating a method 1000 of operation at a computing device to generate a checksum request, according to an example. Method 1000 may be implemented using computing device 108 of FIGs. 1-4 and/or computing device 700 of FIG. 7.

[0055] Method 1000 may include transmitting a checksum request to a data storage device, wherein the checksum request includes a file identifier of a file and file segment information of the file, where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment, and where the file is stored at the data storage device, at 1002. For example, referring to FIG. 1, computing device 108 may transmit a first checksum request 110 associated with a first file 112 to data storage device 100 to initiate an integrity verification operation of first file 112. Method 1000 may also include receiving a checksum report associated with the file from the data storage device, where the checksum report is generated based on the checksum request, at 1004. For example, referring to FIG. 1, computing device 108 may receive comparison result 124 and/or checksum report 126 from data storage device 100. Method 1000 may further include determining an integrity of the file using the checksum report, at 1006. For example, referring to FIG. 1, computing device 108 may determine an integrity of first file 112 using comparison result 124.

[0056] The use of "comprising", "including" or "having" are synonymous and variations thereof herein are meant to be inclusive or open-ended and do not exclude additional unrecited elements or method steps.

Claims

What is claimed is:

1. A data storage device comprising:

a communication interface to receive a checksum request from a computing device, wherein the checksum request includes file segment information of a file and an expected checksum of the file, and wherein the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment;

a checksum calculation module to, in response to a reception of the checksum request, generate a computed checksum of the file using the file segment information; and

a checksum comparison module to:

compare the computed checksum to the expected checksum to generate a comparison result of the file; and

transmit the comparison result to the computing device via the communication interface.

2. The data storage device of claim 1, wherein the communication interface is to receive a second checksum request from the computing device, wherein the second checksum request includes file segment information of a second file and an expected checksum of the second file, wherein the second file includes a first segment and a second segment, wherein the segment information of the second file includes a starting LBA of the first segment, a length value of the first segment, a starting LBA of the second segment, and a length value of the second segment.

3. The data storage device of claim 2, wherein the checksum calculation module is to, in response to a reception of the second checksum request:

generate a first segment checksum using the starting LBA of the first segment and the length value of the first segment;

generate a second segment checksum using the starting LBA of the second segment and the length value of the second segment; and

generate a computed checksum of the second file using the first segment checksum and the second segment checksum, and wherein the checksum comparison module is to:

compare the computed checksum of the second file to the expected checksum of the second file to generate a comparison result of the second file; and

transmit the comparison result of the second file to the computing device via the communication interface.

4. The data storage device of claim 1, wherein the comparison result is indicative of an integrity of the file.

5. The data storage device of claim 1, wherein the length value is indicative of a range of LBAs associated with the file.

6. A method comprising:

receiving, at a data storage device, a checksum request from a computing device, wherein the checksum request includes file segment information of a file and a file identifier of the file, and wherein the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment;

in response to receiving the checksum request, generating a computed checksum of the file using the file segment information; and

transmitting the computed checksum to the computing device.

7. The method of claim 6, further comprising receiving a second checksum request from the computing device, wherein the second checksum request includes file segment information of a second file, an expected checksum of the second file, and a file identifier of the second file, wherein the second file includes a first segment and a second segment, wherein the segment information of the second file includes a starting LBA of the first segment, a length value of the first segment, a starting LBA of the second segment, and a length value of the second segment.

8. The method of claim 7, further comprising:

generating a first segment checksum using the starting LBA of the first segment and the length value of the first segment;

generating a second segment checksum using the starting LBA of the second segment and the length value of the second segment;

generating a computed checksum using the first segment checksum and the second segment checksum; and

transmitting the computed checksum to the second computing device.

9. The method of claim 6, wherein the length value is indicative of a range of LBAs associated with the segment.

10. A computer-readable storage medium comprising instructions when executed cause a processor of a computing device to:

transmit a checksum request to a data storage device, wherein the checksum request includes a file identifier of a file and file segment information of the file, wherein the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment, and wherein the file is stored at the data storage device;

receive a checksum report associated with the file from the data storage device, wherein the checksum report is generated based on the checksum request; and

determine an integrity of the file using the checksum report.

11. The computer-readable storage medium of claim 10, wherein the checksum request further includes an expected checksum of the file, wherein the checksum report includes a comparison result between the expected checksum and a computed checksum of the file.

12. The computer-readable storage medium of claim 10, wherein the checksum report includes a computed checksum of the file, and wherein the instructions when executed further cause the processor to:

compare the computed checksum with an expected checksum of the file to generate a comparison result; and

determine the integrity based on the comparison result.

13. The computer-readable storage medium of claim 12, wherein the instructions when executed further cause the processor to:

in response to a determination that the expected checksum is different from the computed checksum, determine that the file is corrupted; and

transmit a replacement copy of the file to the data storage device.

14. The computer-readable storage medium of claim 12, wherein the file includes a plurality of segments, and wherein each of the plurality of segments is stored at a distinct range of logical block addresses.

15. The computer-readable storage medium of claim 12, wherein the data storage device is connected to the computing device via a network.

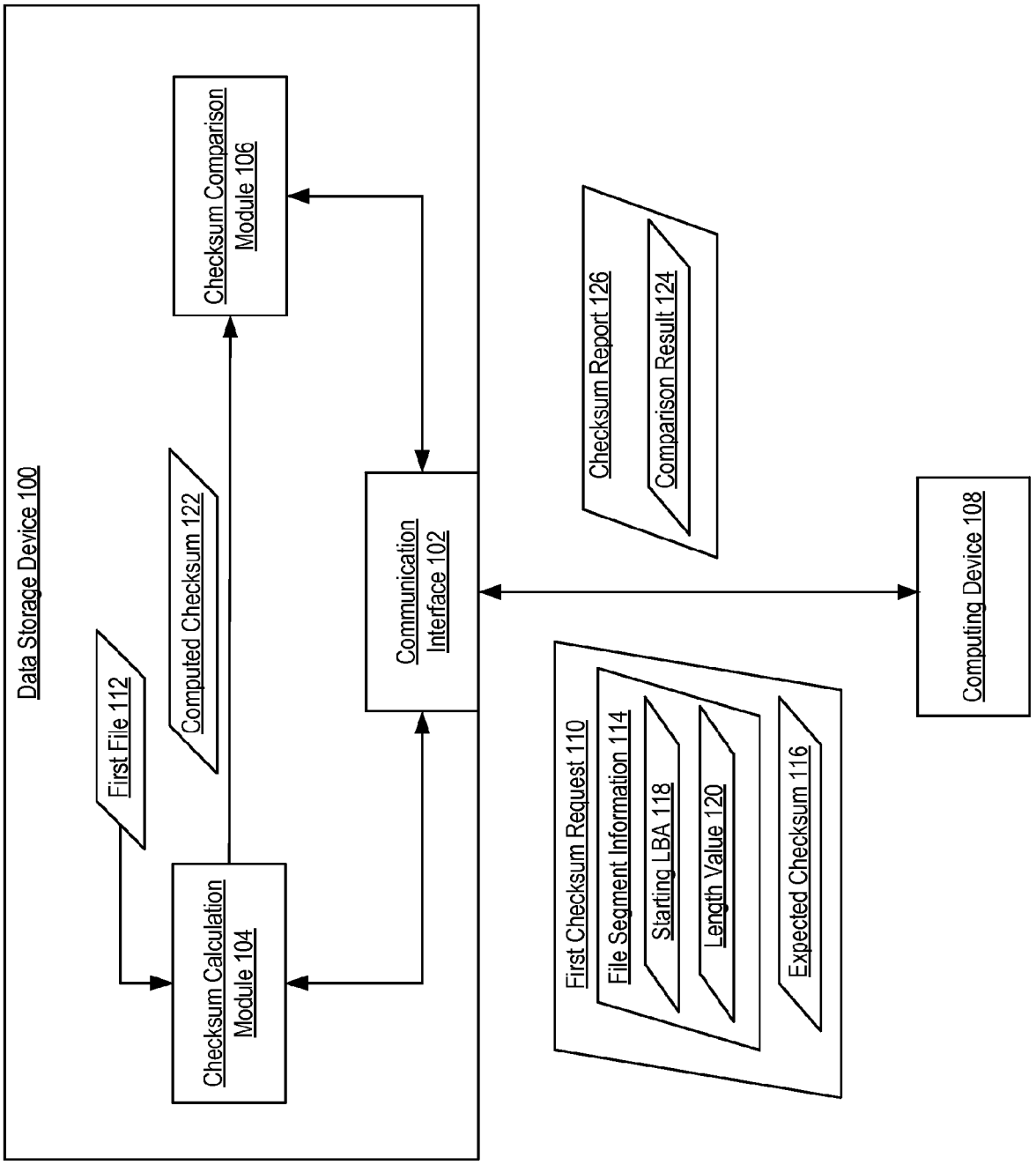
Data Storage Device 100
Checksum Comparison Module 106
Computed Checksum 122
First File 112
Checksum Calculation Module 104
Communication Interface 102
Checksum Report 126
Comparison Result 124
First Checksum Request 110
File Segment Information 114
Starting LBA 118
Length Value 120
Expected Checksum 116
Computing Device 108

FIG. 1

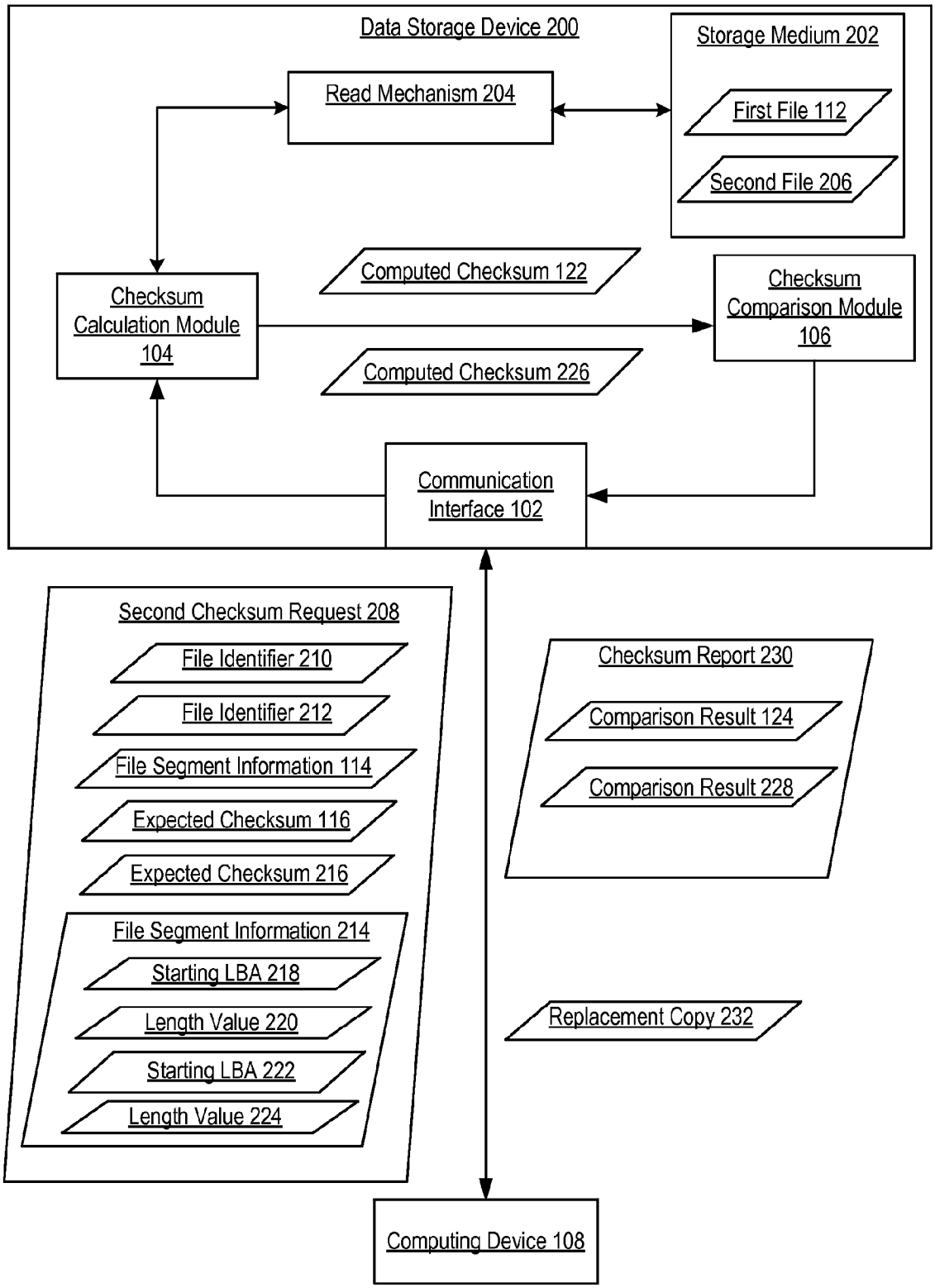
Data Storage Device 200
Storage Medium 202
Read Mechanism 204
First File 112
Second File 206
Computed Checksum 122
Checksum Calculation Module 104
Checksum Comparison Module 106
Computed Checksum 226
Communication Interface 102
Second Checksum Request 208
File Identifier 210
File Identifier 212
File Segment Information 114
Expected Checksum 116
Expected Checksum 216
File Segment Information 214
Starting LBA 218
Length Value 220
Starting LBA 222
Length Value 224
Checksum Report 230
Comparison Result 124
Comparison Result 228
Replacement Copy 232
Computing Device 108

FIG. 2

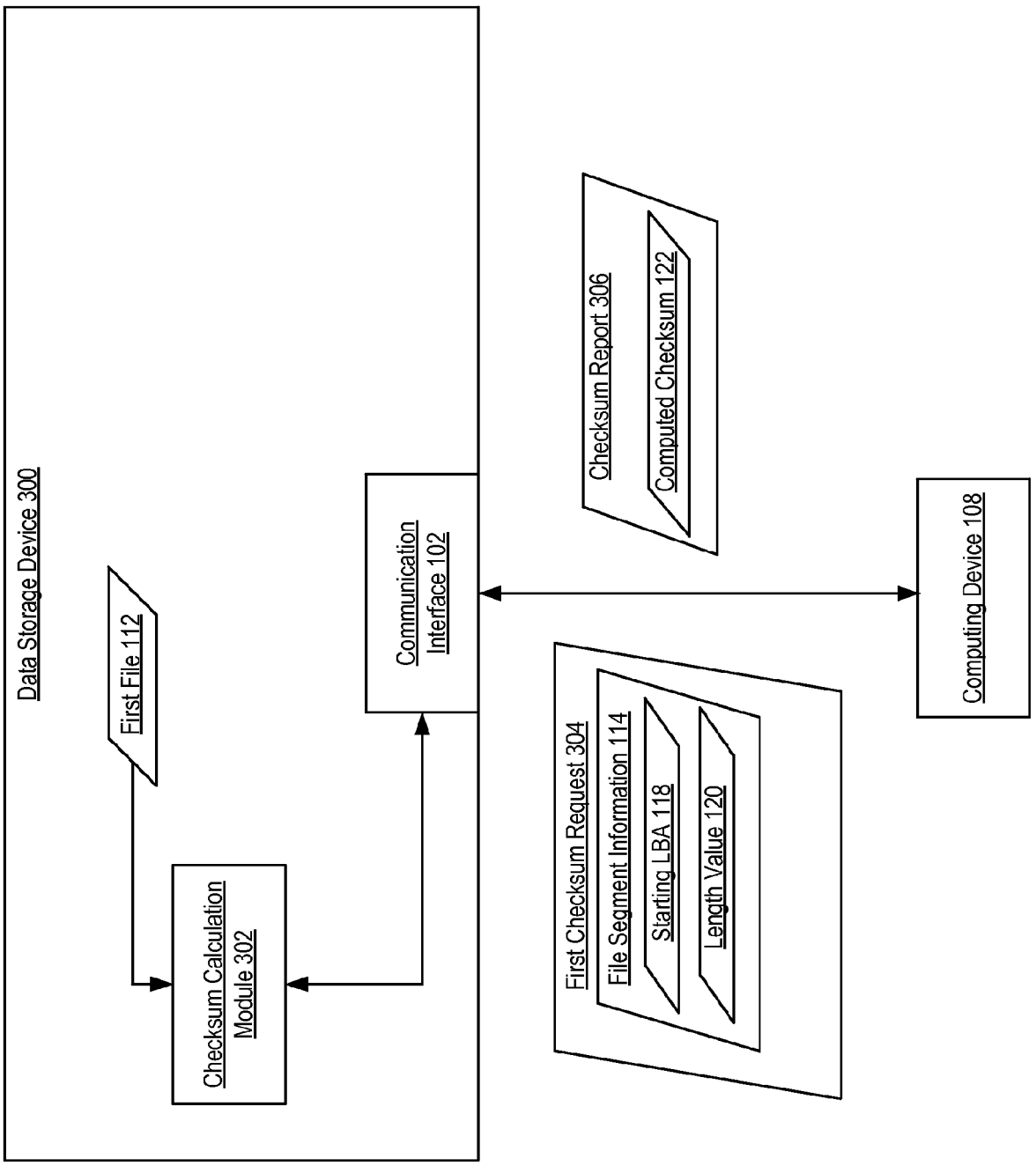
Data Storage Device 300
First File 112
Checksum Calculation Module 302
Communication Interface 102
Checksum Report 306
Computed Checksum 122
First Checksum Request 304
File Segment Information 114
Starting LBA 118
Length Value 120
Computing Device 108

FIG. 3

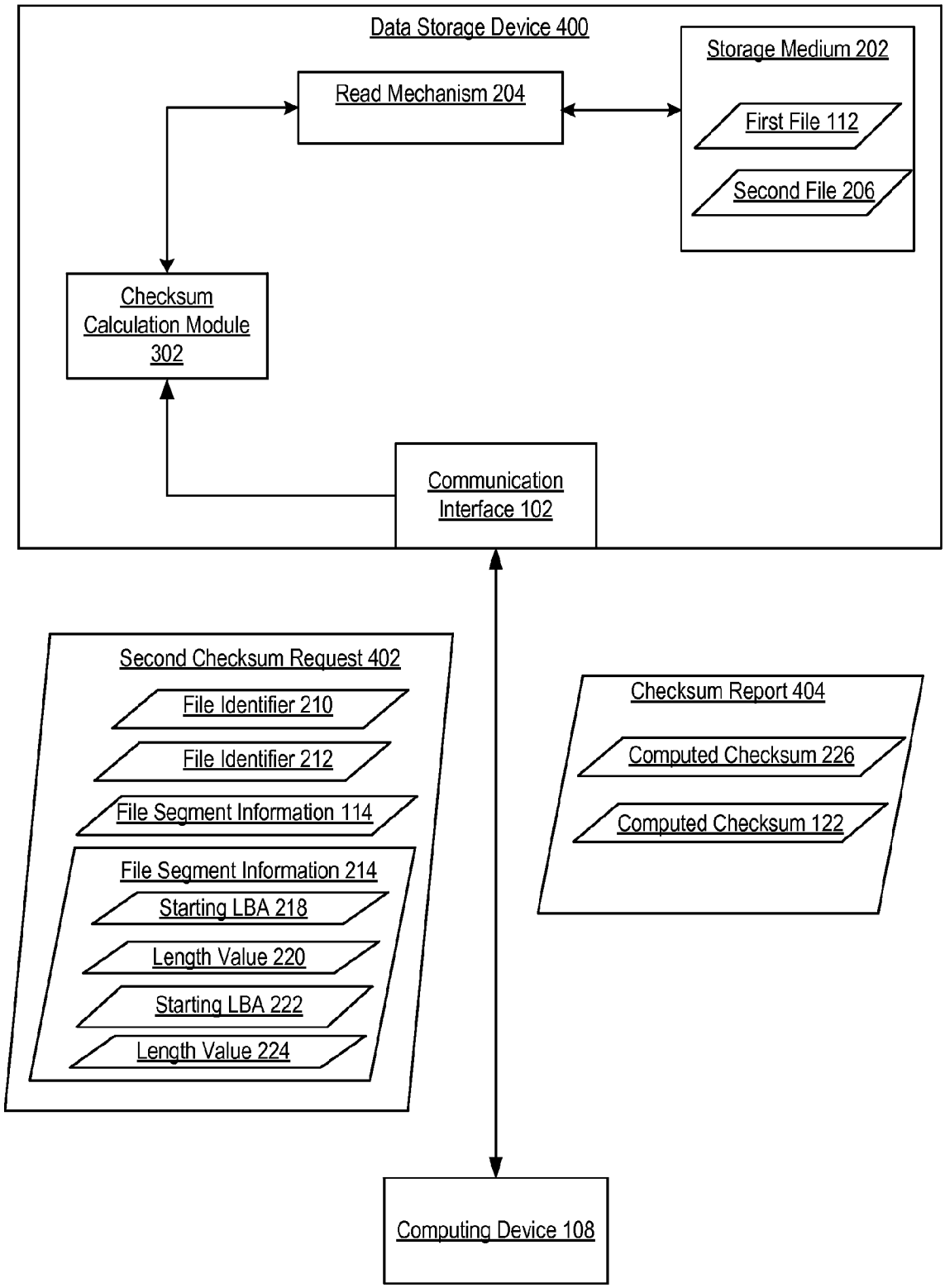
Data Storage Device 400
Storage Medium 202
Read Mechanism 204
First File 112
Second File 206
Checksum Calculation Module 302
Communication Interface 102
Second Checksum Request 402
File Identifier 210
File Identifier 212
File Segment Information 114
File Segment Information 214
Starting LBA 218
Length Value 220
Starting LBA 222
Length Value 224
Checksum Report 404
Computed Checksum 226
Computed Checksum 122
Computing Device 108

FIG. 4

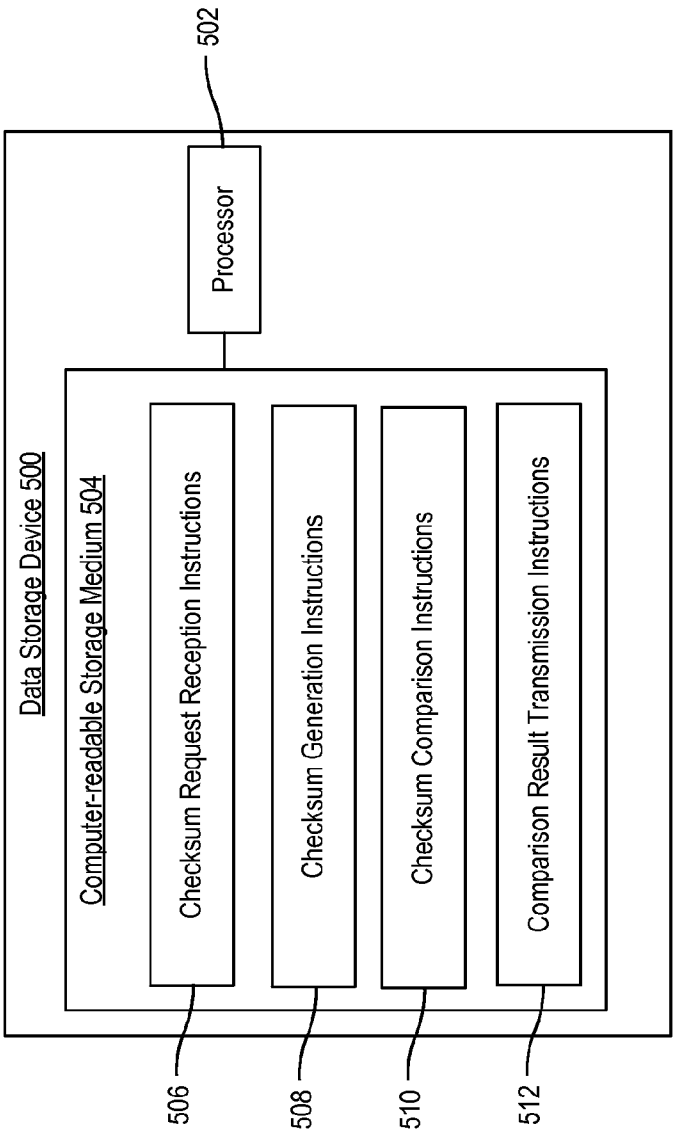
502
Processor
Data Storage Device 500
Computer-readable Storage Medium 504
Checksum Request Reception Instructions
Checksum Generation Instructions
Checksum Comparison Instructions
Comparison Result Transmission Instructions
506
508
510
512

FIG. 5

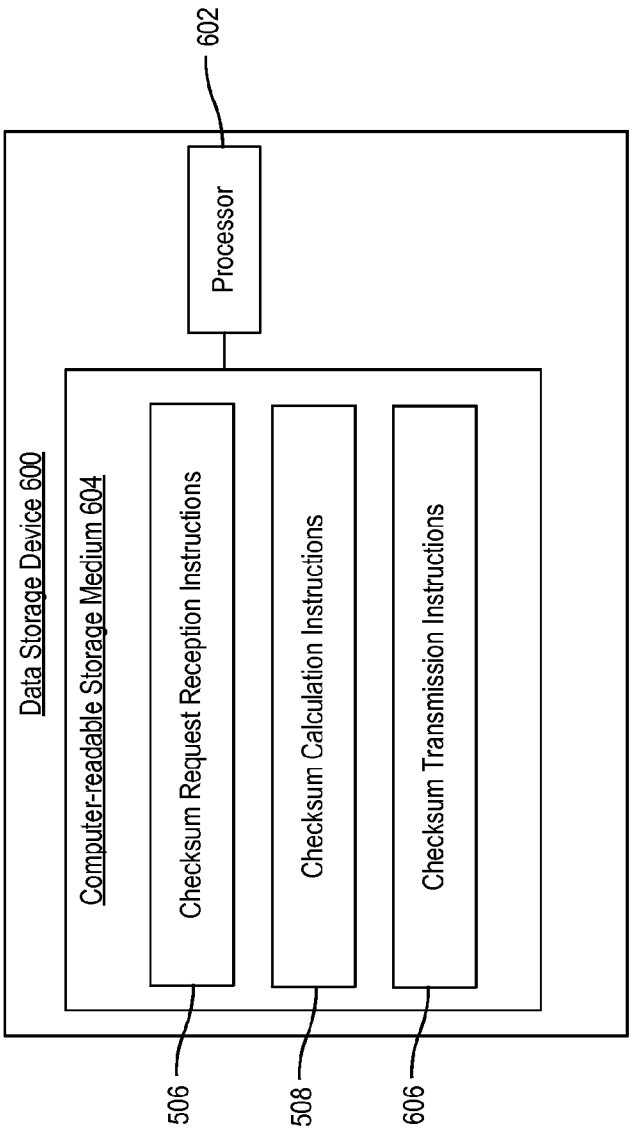
602
Processor
Data Storage Device 600
Computer-readable Storage Medium 604
Checksum Request Reception Instructions
Checksum Calculation Instructions
Checksum Transmission Instructions
506
508
606

FIG. 6

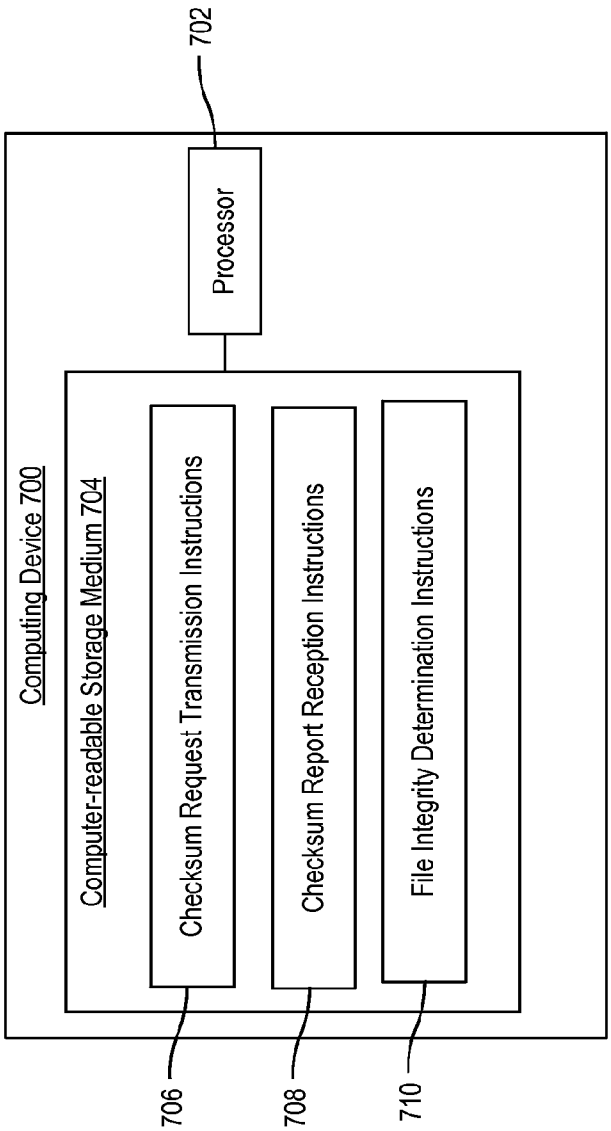
Computing Device 700
Processor
702
Computer-readable Storage Medium 704
Checksum Request Transmission Instructions
Checksum Report Reception Instructions
File Integrity Determination Instructions
706
708
710

FIG. 7

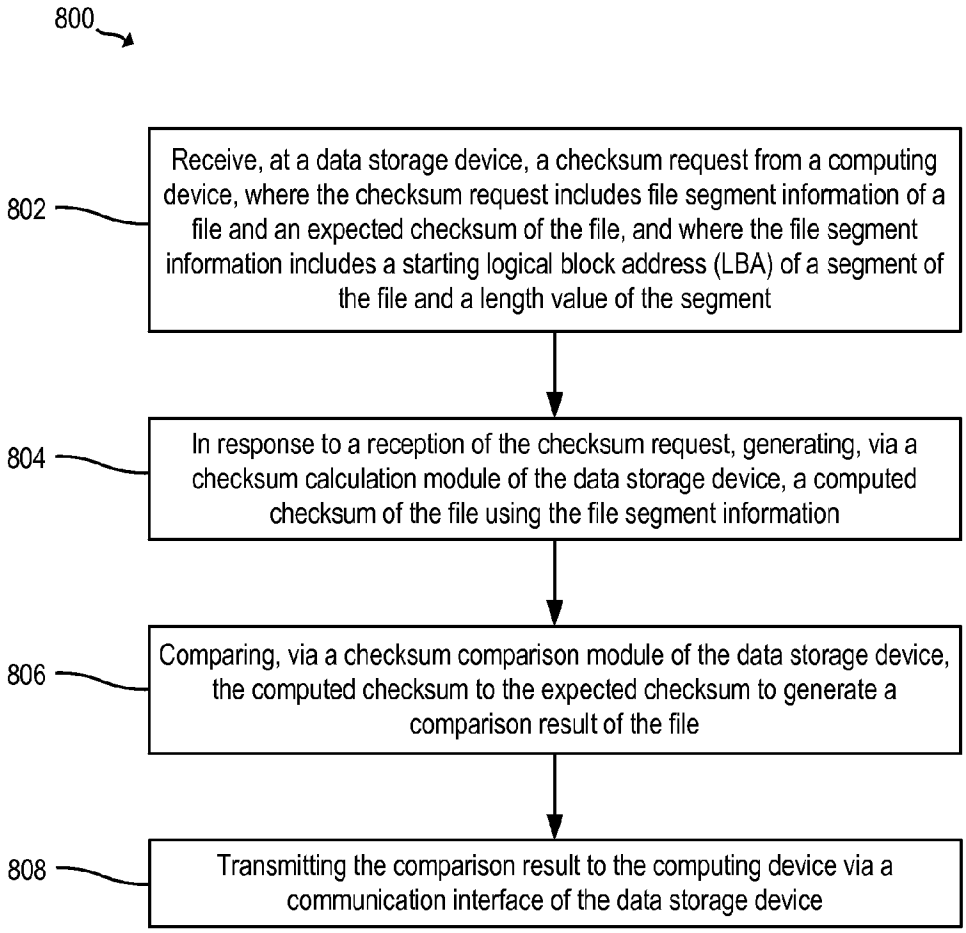
800
802
Receive, at a data storage device, a checksum request from a computing device, where the checksum request includes file segment information of a file and an expected checksum of the file, and where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment
804
In response to a reception of the checksum request, generating, via a checksum calculation module of the data storage device, a computed checksum of the file using the file segment information
806
Comparing, via a checksum comparison module of the data storage device, the computed checksum to the expected checksum to generate a comparison result of the file
808
Transmitting the comparison result to the computing device via a communication interface of the data storage device

FIG. 8

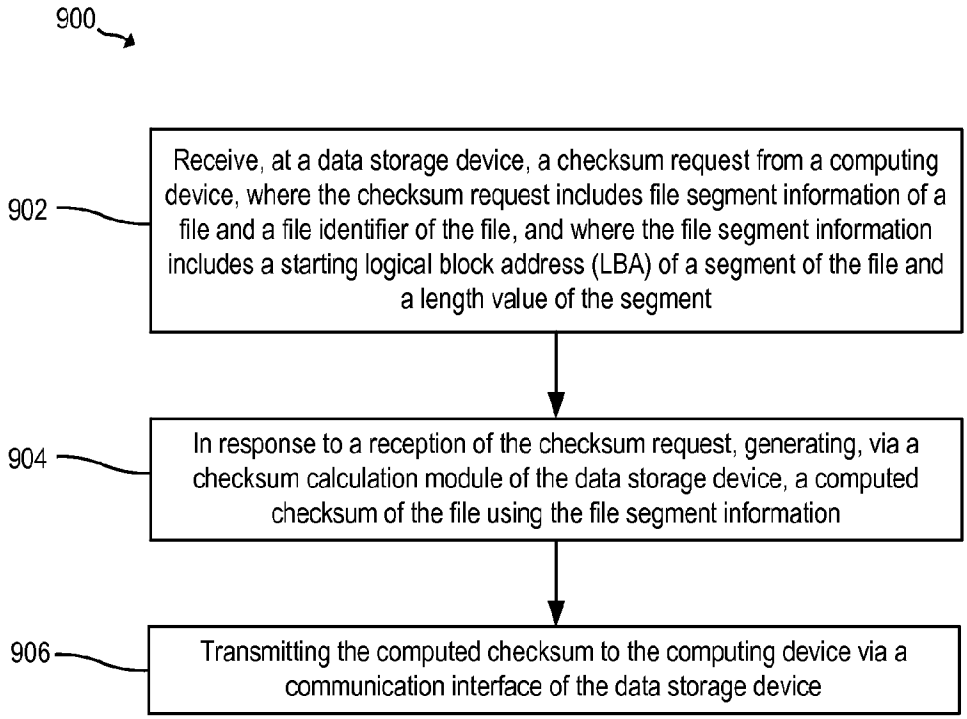
900
902
Receive, at a data storage device, a checksum request from a computing device, where the checksum request includes file segment information of a file and a file identifier of the file, and where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment
904
In response to a reception of the checksum request, generating, via a checksum calculation module of the data storage device, a computed checksum of the file using the file segment information
906
Transmitting the computed checksum to the computing device via a communication interface of the data storage device

FIG. 9

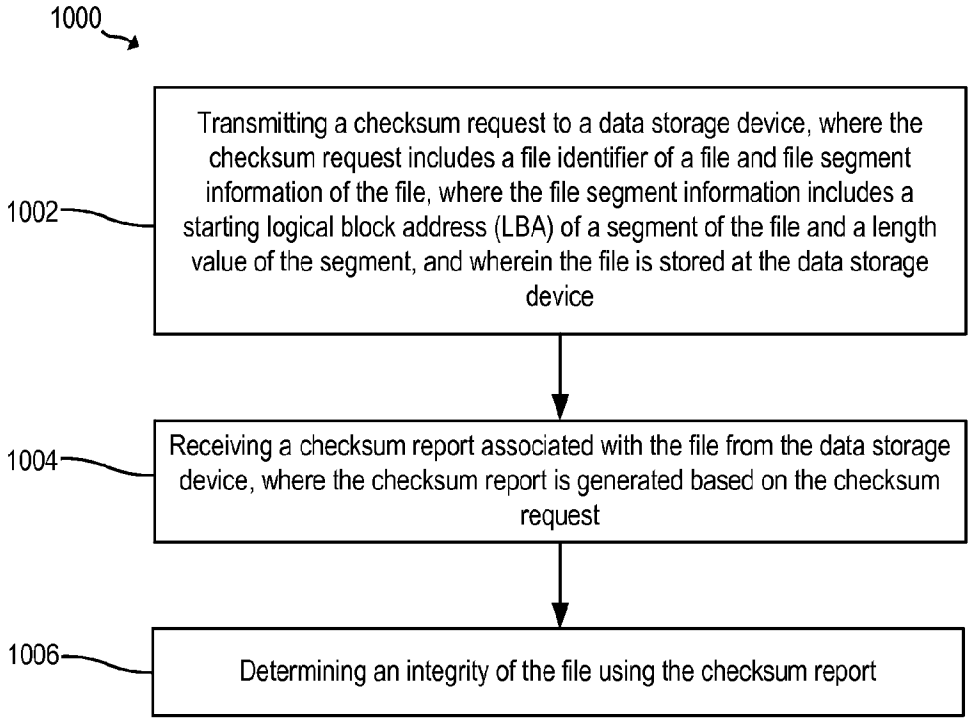
1000
Transmitting a checksum request to a data storage device, where the checksum request includes a file identifier of a file and file segment information of the file, where the file segment information includes a starting logical block address (LBA) of a segment of the file and a length value of the segment, and wherein the file is stored at the data storage device
1002
Receiving a checksum report associated with the file from the data storage device, where the checksum report is generated based on the checksum request
1004
Determining an integrity of the file using the checksum report
1006

FIG. 10

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/017676

A. CLASSIFICATION OF SUBJECT MATTER

G06F 11/10(2006.01)i, G06F 12/06(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 11/10; H03M 13/00; G06F 11/08; G06F 13/16; G06F 11/00; G11C 29/00; G06F 12/06

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: storage device, expected checksum, comparison, LBA, request, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2005-0289436 A1 (JOHN REDFORD) 29 December 2005 See paragraphs [0027], [0059] and [0062]; claims 1, 8, 19, and 22; and figure 1.	1-15
Y	US 8,484,537 B1 (TANG HENG et al.) 09 July 2013 See column 3, lines 50-60; claims 1-3; and figure 3B.	1-15
A	US 2006-0136780 A1 (YI MENG et al.) 22 June 2006 See paragraphs [0057]-[0060] and [0063]-[0072]; and figures 2-3.	1-15
A	WO 2010-096153 A2 (MICRON TECHNOLOGY, INC.) 26 August 2010 See paragraphs [0006] and [0013]; and figure 1.	1-15
A	US 2004-0049726 A1 (GREGG S. GOYINS et al.) 11 March 2004 See paragraphs [0032]-[0034] and figure 2.	1-15

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier 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
15 December 2015 (15.12.2015)	**15 December 2015 (15.12.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
NHO, Ji Myong
Telephone No. +82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/017676

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2005-0289436 A1	29/12/2005	US 7415654 B2	19/08/2008
US 8484537 B1	09/07/2013	US 7840878 B1	23/11/2010
		US 9129654 B1	08/09/2015
US 2006-0136780 A1	22/06/2006	US 2004-0003316 A1	01/01/2004
		US 7020798 B2	28/03/2006
		US 7441159 B2	21/10/2008
WO 2010-096153 A2	26/08/2010	CN 102317919 A	11/01/2012
		CN 102317919 B	11/03/2015
		EP 2399194 A2	28/12/2011
		EP 2399194 A4	31/10/2012
		JP 2012-518224 A	09/08/2012
		JP 5776107 B2	09/09/2015
		KR 10-1351754 B1	14/01/2014
		KR 10-1457518 B1	10/11/2014
		KR 10-2011-0118168 A	28/10/2011
		KR 10-2013-0124989 A	15/11/2013
		TW 201035987 A	01/10/2010
		TW 201434051 A	01/09/2014
		TW I451434 B	01/09/2014
		US 2010-0211834 A1	19/08/2010
		US 2013-0283124 A1	24/10/2013
		US 2015-0220386 A1	06/08/2015
		US 8468417 B2	18/06/2013
		US 9015553 B2	21/04/2015
		WO 2010-096153 A3	25/11/2010
US 2004-0049726 A1	11/03/2004	GB 2394331 A	21/04/2004
		GB 2394331 B	21/12/2005
		JP 2004-103005 A	02/04/2004
		US 6938201 B2	30/08/2005