



- (51) **International Patent Classification:**
G06F 11/14 (2006.01) *G06F 21/32* (2013.01)
- (21) **International Application Number:**
PCT/US2015/042831
- (22) **International Filing Date:**
30 July 2015 (30.07.2015)
- (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) **Inventors:** NAZARI, Siamak; 4245 Technology Drive, Fremont, California 94538 (US). WANG, Jin; 4245 Technology Drive, Fremont, California 94538 (US). MURTHY, Srinivasa D.; 4245 Technology Drive, Fremont, California 94538 (US).
- (74) **Agents:** ORTEGA, Arthur S. 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).

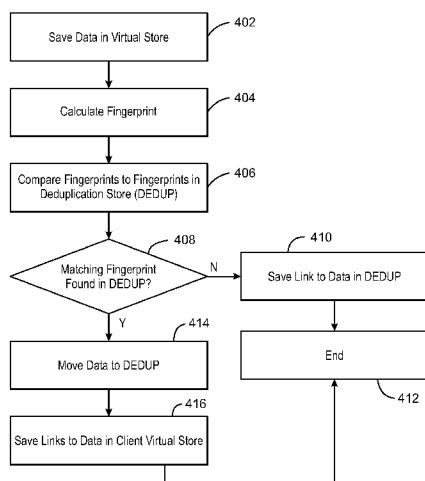
Declarations under Rule 4.17:

- 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:** STORING DATA IN A DEDUPLICATION STORE



400
FIG. 4

(57) **Abstract:** Techniques are provided for storing data in a deduplication store. A method includes calculating a fingerprint for data stored in a client data store. The fingerprint is compared to each of a plurality of fingerprints in a deduplication store. If the data fingerprint matches one of the plurality of fingerprints in the deduplication store, the data is moved to the deduplication store, and a back reference to the data in the deduplication store is placed in the client data store.

STORING DATA IN A DEDUPLICATION STORE

BACKGROUND

[0001] Primary data storage systems provide data services to their clients through the abstraction of data stores, for example, as virtual volumes. These virtual volumes could be of different types, such as fully pre-provisioned or thin-provisioned or thin-provisioned and deduplicated.

DESCRIPTION OF THE DRAWINGS

[0002] Certain exemplary embodiments are described in the following detailed description and in reference to the drawings, in which:

[0003] Fig. 1 is an example of a system for storing deduplicated data;

[0004] Fig. 2 is a schematic example of a system for storing deduplicated data;

[0005] Fig. 3 is a schematic example of a system for storing deduplicated data;

[0006] Fig. 4 is a process flow diagram of an example method for storing deduplicated data;

[0007] Fig. 5A is a block diagram of an example non-transitory, computer readable medium comprising code to direct one or more processors to save deduplicated data; and

[0008] Fig. 5B is a another block diagram of the example non-transitory, computer readable medium comprising code to direct one or more processors to save deduplicated data.

DETAILED DESCRIPTION

[0009] Primary data storage systems provide data services to their clients through the abstraction of data stores, for example, as virtual volumes. These virtual volumes could be of different types, such as fully pre-provisioned or thin-provisioned or thin-provisioned and deduplicated. Such virtual volumes eventually need physical storage to store the data written to the virtual volumes. Normal thin-provisioned volumes can have data stores that are private to each such virtual volume. When a

storage service provides deduplication among multiple virtual volumes, there can be a common deduplication store that is shared among such virtual volumes. Often, all data, whether it is duplicate data with multiple references or not, is saved in the common deduplication store. The virtual volumes only save deduplication collision data on local data stores when the data is different from data already residing in the deduplication store but has the same fingerprint signature.

[0010] Techniques described herein combine data stores, such as virtual volumes, with a deduplication store to efficiently store data. In examples described herein, the common deduplication store is used only to store duplicate data. When new data gets written to a client data store, such as a data store associated with a virtual volume, for the first time, the data gets stored in the client data store. A link to the data in the data store is written to the deduplication store, wherein the link includes the fingerprint, or hash code, associated with the data and a back reference to the data store holding the data. When a subsequent write to any of the other client data stores occurs, a fingerprint of the new data is computed and compared to the fingerprints in the deduplication store. If the new fingerprint matches a fingerprint previously stored in the deduplication store, the new data is moved to the deduplication store. Back references are then written to the associated client data stores to point to the deduplication store.

[0011] In deduplication systems without reference counting, unreferenced pages in the deduplication store are garbage collected periodically. If the deduplication store is used for all data, there can be a lot of data in the deduplication store with only single references. When such singleton data gets overwritten, it will create a lot of unreferenced pages that need to be garbage collected. This demands more aggressive garbage collection, which can adversely impact data services. If garbage collection is not aggressive enough, it may lead to larger deduplication store sizes. Thus, the aggressiveness of the garbage collection is balanced with the size of the storage space. In deduplication systems without reference counts and with background garbage collection, the approach described herein may result in less garbage, e.g., orphaned data occupying system storage space, and fewer singleton references in the deduplication store.

[0012] With data stored in the private data stores, data that is overwritten may be done by replacement of the old data with the new data in place. The new data and old data may have different fingerprints, and when the fingerprint of the new data is calculated, the old link in the deduplication store may be replaced. Further, by storing singleton references in private data stores, better performance may be achieved for sequential writes of singleton references, through coalescing writes to backend disks.

[0013] Fig. 1 is an example of a system 100 for storing deduplicated data. In this example, a server 102 may perform the functions described herein. The server 102 may host a number of client data stores 104-110, as well as a deduplication store 112. The client data stores 104-110 may be part of virtual machines 114-120 or may be separate virtual drives, or physical drives, controlled by the server 102.

[0014] The server 102 may include a processor (or processors) 122 that is configured to execute stored instructions, as well as a memory device (or memory devices) 124 that stores instructions that are executable by the processor 122. The processor 122 can be a single core processor, a dual-core processor, a multi-core processor, a computing cluster, a cloud sever, or the like. The processor 122 may be coupled to the memory device 124 by a bus 126 where the bus 126 may be a communication system that transfers data between various components of the server 102. In embodiments, the bus 126 may be a PCI, ISA, PCI-Express, or the like.

[0015] The memory device 124 can include random access memory (RAM), e.g., static RAM, DRAM, zero capacitor RAM, eDRAM, EDO RAM, DDR RAM, RRAM, PRAM, read only memory (ROM), e.g., Mask ROM, PROM, EPROM, EEPROM, flash memory, or any other suitable memory systems. The memory device 124 may store code and links configured to administer the data stores 104-110.

[0016] The server 102 may also include a storage device 128. In some examples, multiple storage devices 128 are used, such as in a storage attached network (SAN). The storage device 128 may include non-volatile storage devices, such as a solid-state drive, a hard drive, an optical drive, a flash drive, an array of drives, or any combinations thereof. In some examples, the storage device 128 may

include non-volatile memory, such as non-volatile RAM (NVRAM), battery backed up DRAM, and the like.

[0017] A network interface controller (NIC) 130 may also be linked to the processor 122. The NIC 130 may link the server 102 to a network 132, for example, to couple the server to clients located in a computing cloud 134. Further, the network 132 may couple the server 102 to management devices 136 in a data center to set up and control the client data stores 104-110.

[0018] The storage device 128 may include a number of modules configured to provide the server 102 with the deduplication functionality. For example, a fingerprint generator (FG) 138, which may be located in the client data stores 104-110, may be utilized to calculate a fingerprint, e.g., a hash code, for new data written to the client data store. A fingerprint comparator (FC) 140 may be used to compare the fingerprints generated to fingerprints in the deduplication store, e.g., associated with either links 142 and 144 or data 146 and 148. If a fingerprint matches, a data mover (DM) 150 may then be used to move the data to the deduplication store 112, if it is not already present. If the data is already in the deduplication store 112, the DM 150 may be used to copy a back reference to the client data store 104-110 to point to the data in the deduplication store 112 and remove the data from the client data store 104-110. The process is explained further with respect to the schematic drawings of Figs. 2 and 3 and the method of Fig. 4.

[0019] In the present example, a single copy of data D1 152 is saved to client data store 106 in virtual machine 2 116. An associated link L1 144, including a fingerprint of the data D1 152 and a backreference to the data D1 152 in the client data store 106 is in the deduplication store 112. A single copy of a second piece of data D2 154 is saved to client data store 108 in virtual machine 3 118. An associated link L2 142, including a fingerprint of the data D2 154 and a backreference to the data D2 154 in the client data store 108 is in the deduplication store 112.

[0020] Further, in this example, data D3 146 is duplicate data that has been written to more than one client data store. A single copy of the data D3 146 is saved to the deduplication store 112 along with the fingerprint of the data. Links L3 156 to this data D3 146, are saved to the associated client data stores 104 and 110.

Similarly, data D4 148 is duplicate data, in which a single copy is saved to the deduplication store 112 along with the fingerprint of the data. Links L4 158 to this data D3 148, are in the associated client data stores 106 and 108. It may be noted that this example has been simplified for clarity. In a real system, there may be many thousands of individual data blocks and links.

[0021] The block diagram of Fig. 1 is not intended to indicate that the system 100 is arranged as shown in Fig. 1. For example, the virtual machines 114-120 may not be present. The client data stores 104-110 may be virtual drives distributed among drives in a storage attached network, as mentioned above. Further, the various operational modules used to provide the deduplication functionality, such as the FG 138, the FC 140, and the DM 150 may be located in the deduplication store 112, or in another location, such as in a separate area of the storage device 128 itself or in a management device 136. In some examples, the deduplication store 112 may include a link generator to associate a matching fingerprint and a back reference to a location for the data in the deduplication store. Further, the deduplication store 112 may include a link saver to save a link to matched data in the deduplication store to a data store.

[0022] The techniques described herein may be clarified by stepping through individual data writes. This is described with respect to Figs. 2 and 3. Although these examples include virtual machines, it can be understood that the present techniques apply to any deduplicated data stores, including virtual drives or deduplicated physical drives.

[0023] Fig. 2 is a schematic example 200 of storing deduplicated data. Like numbered items are as described with respect to Fig. 1. In this example, new data, DATA1 202 is written 204 to virtual machine 2 116. A fingerprint for the stored DATA1 206 is calculated and compared to fingerprints in the deduplication store 112. Since DATA1 206 is new (unmatched) data, a link, Link1 208, is stored to the deduplication store 112. Link1 208 has the calculated fingerprint associated with DATA1 206, and a backreference 210 to the location of DATA1 206 in the client data store 106.

[0024] Similarly, more new (unmatched) data, DATA2 212 is written 214 to virtual machine 3 118, and saved to the client data store 108 as DATA2 216. A fingerprint

is generated for DATA2 216, but since there are no matching fingerprints in the deduplication store 112, a link, Link2 218 is saved in the deduplication store 112. As for Link1 208, Link2 218 includes the fingerprint of DATA2 216 and a backreference 220 to the location of DATA2 216 in the client data store 108.

[0025] Fig. 3 is a schematic example 300 of storing deduplicated data. Like numbered items are as described with respect to Figs. 1 and 2. This example takes place after the example shown in Fig. 2, when DATA1 202 is written 302 to virtual machine 4 120 and is temporarily saved (not shown). In this example, a fingerprint is generated for DATA1 202, which matches the fingerprint saved in Link1 208 of Fig. 2. Accordingly, the matched data is moved to the deduplication store 112, and saved as DATA1 304. A link to DATA1 304, Link 1A 306 is saved to the client data store 110 for virtual machine 4 120 and to the client data store 106 for virtual machine 2 116. Link 1A may include the fingerprint of DATA1 304 and a backreference 308 to the location of DATA1 304 in the deduplication store 112. The associated fingerprint for DATA1 304 may also be kept in the deduplication store 112 for further comparisons in case the data is written to other virtual machines.

[0026] Fig. 4 is a process flow diagram of an example method 400 for storing deduplicated data. The method 400 begins at block 402, with the data being saved to a client data store, for example, in a virtual machine, a virtual drive, or a deduplicated physical drive. At block 404, a fingerprint is calculated for the data, for example, by the generation of a hash code from the data. At block 406, the fingerprint is compared to fingerprints saved in the deduplication store.

[0027] If, at block 408, a matching fingerprint is not found in the deduplication store, process flow proceeds to block 410. At block 410, a link to the data in the client data store is saved in the deduplication store. The link includes the fingerprint of the data and a backreference to the location of the data in the client data store. If there is an old link associated with old data, it should be removed after the new link to new data is created in DEDUP. The method 400 then ends at block 412.

[0028] If a matching fingerprint is found at block 408, at block 414, the data is moved to the deduplication store. In one example, the data already exists in the deduplication store, in which case, no data is moved. At block 416, links to the data are saved to the associated client data stores. These links may include the

fingerprint of the data and a backreference to the data saved in the deduplication store. The original fingerprint of the data may also be retained in the deduplication store for further comparisons.

[0029] If the data is removed from all but one client, it may be left in the deduplication store to minimize unnecessary data moves that consume resources. If the data is deleted from that final client, then garbage collection may be used to remove the data from the deduplication store.

[0030] Fig. 5A is a block diagram of an example non-transitory, computer readable medium 500 comprising code or computer readable instructions to direct one or more processors to save deduplicated data. The computer readable medium 500 is coupled to one or more processors 502 over a bus 504. The processors 502 and bus 504 may be as described with respect to the processors 122 and bus 126 of Fig. 1.

[0031] The computer readable medium 500 includes a block 506 of code to direct one of the one or more processors 502 to calculate a fingerprint for data written to a client data store. Another block 508 of code directs one of the one or more processors 502 to compare the fingerprint to fingerprints stored in the deduplication store. The computer readable medium 500 also includes a block 510 of code to direct one of the one or more processors 502 to move data to the deduplication store. A block 512 of code may direct one of the one or more processors 502 to write links to the data to each client data store that is associated with that data. Further, a block 514 of code may direct one of the one or more processors 502 to erase the linked data from the client data stores. In one example, the data that is no longer needed in the client data store, e.g., because it is duplicate data saved in the deduplication store, may be marked and removed to free storage space as part of the normal garbage collection functions in the data store.

[0032] The code blocks above do not have to be separated as shown, the functions may be recombined into different blocks that perform the functions. Further, the computer readable medium does not have to include all of the blocks shown in Fig. 5A.

[0033] Fig. 5B is a another block diagram of the example non-transitory, computer readable medium comprising code to direct one or more processors to

save deduplicated data. Like numbered items are as described with respect to Fig. 5A. This simpler arrangement, includes the core code blocks that may be used to perform the functions described herein in some examples.

[0034] While the present techniques may be susceptible to various modifications and alternative forms, the exemplary examples discussed above have been shown only by way of example. It is to be understood that the technique is not intended to be limited to the particular examples disclosed herein. Indeed, the present techniques include all alternatives, modifications, and equivalents falling within the scope of the present techniques.

CLAIMS

What is claimed is:

1. A method for storing data in a deduplication store, comprising
calculating a fingerprint for data stored in a client data store;
comparing the fingerprint to each of a plurality of fingerprints in the
deduplication store; and, if the fingerprint matches one of the plurality
of fingerprints in the deduplication store:
moving the data to the deduplication store; and
placing a back reference to the data in the deduplication store in the
client data store.
2. The method of claim 1, wherein calculating the fingerprint comprises
generating a hash code for the data.
3. The method of claim 1, comprising:
removing the data from a second client data store after saving the data to the
deduplication store; and
placing the back reference to the data in the deduplication store to the second
client data store.
4. The method of claim 1, comprising associating each of a plurality of
client data stores with the deduplication store.
5. The method of claim 1, comprising, if the fingerprint does not match
one of the plurality of fingerprints in the deduplication store, saving a link to the data
in the deduplication store.
6. The method of claim 5, wherein the link comprises a back reference to
the data in the client data store and an associated fingerprint.

7. A system for storing data in a deduplication store, comprising:
a plurality of data stores, each data store comprising:
a deduplication link to matched data in the deduplication store that has
a matching fingerprint to data from a second data store; and
unmatched data that does not have a matching fingerprint to data in
any other data store;
the deduplication store, comprising:
matched data that is linked to two or more data stores; and
a singleton link to the unmatched data in the data store that does not
have a matching fingerprint to data in any other data store.
8. The system of claim 7, the data store comprising a fingerprint
generator to calculate a hash code for new data stored in the data store.
9. The system of claim 7, the data store comprising a fingerprint
comparator to compare a fingerprint for new data saved in the data store to a
fingerprint in the deduplication store.
10. The system of claim 7, the data store comprising a data mover to copy
new data that has a matching fingerprint to the data store.
11. The system of claim 7, the deduplication store comprising a link
generator to associate the matching fingerprint and a back reference to a location for
the data in the deduplication store.
12. The system of claim 7, the deduplication store comprising a link saver
to save a link to the matched data in the deduplication store to the second data
store.

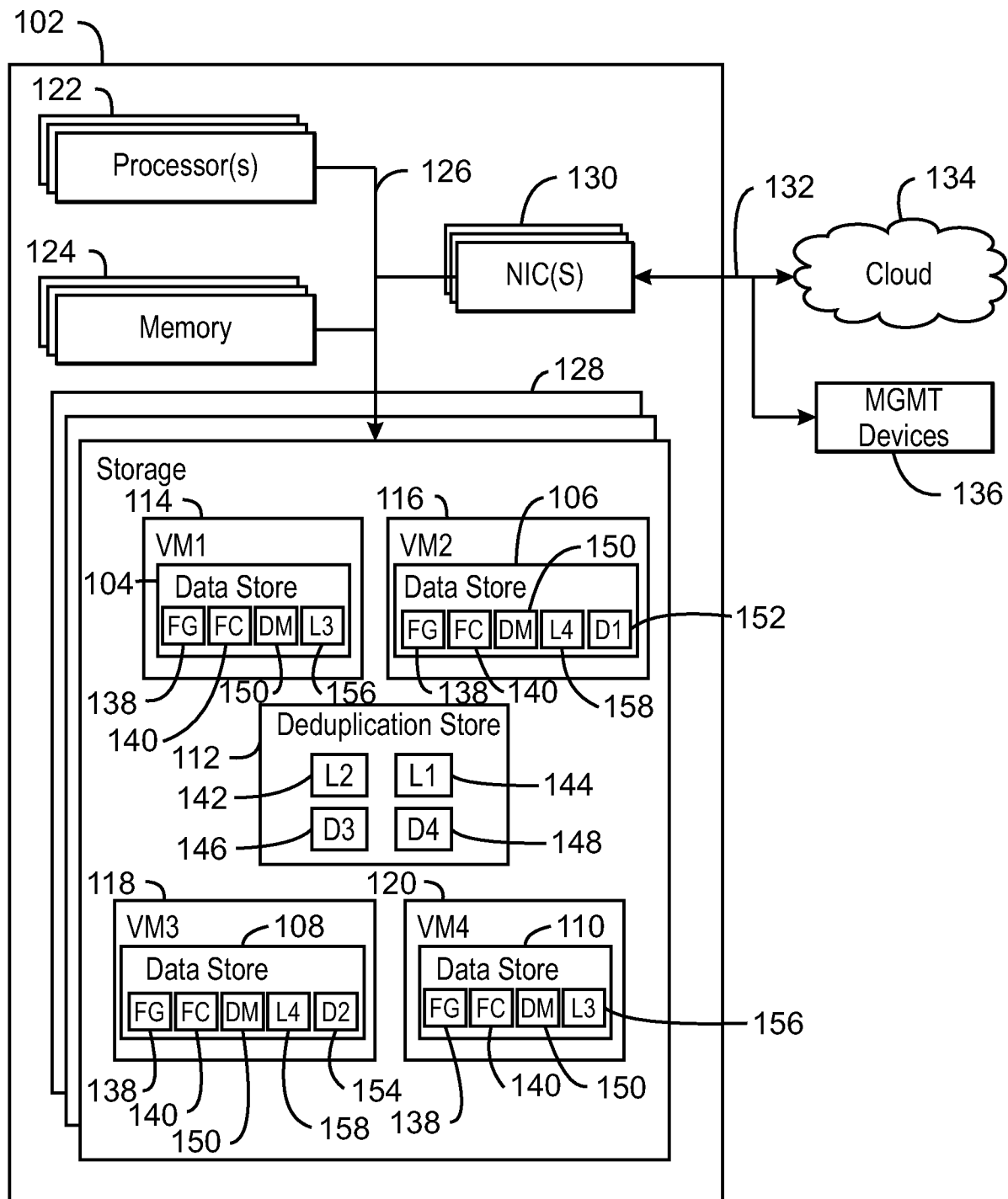
13. A non-transitory, computer readable medium comprising code for storing data in a deduplication store, the code configured to direct one or more processors to:

- calculate a fingerprint for data stored in a client data store;
- compare the fingerprint to each of a plurality of fingerprints in a deduplication store; and
- moving the data to the deduplication store.

14. The non-transitory, computer readable medium of claim 13, comprising code configured to direct one of the one or more processors to place a back reference to the data in the deduplication store in the client data store.

15. The non-transitory, computer readable medium of claim 13, comprising code configured to direct one of the one or more processors to write a link to the data in the deduplication store to another client data store.

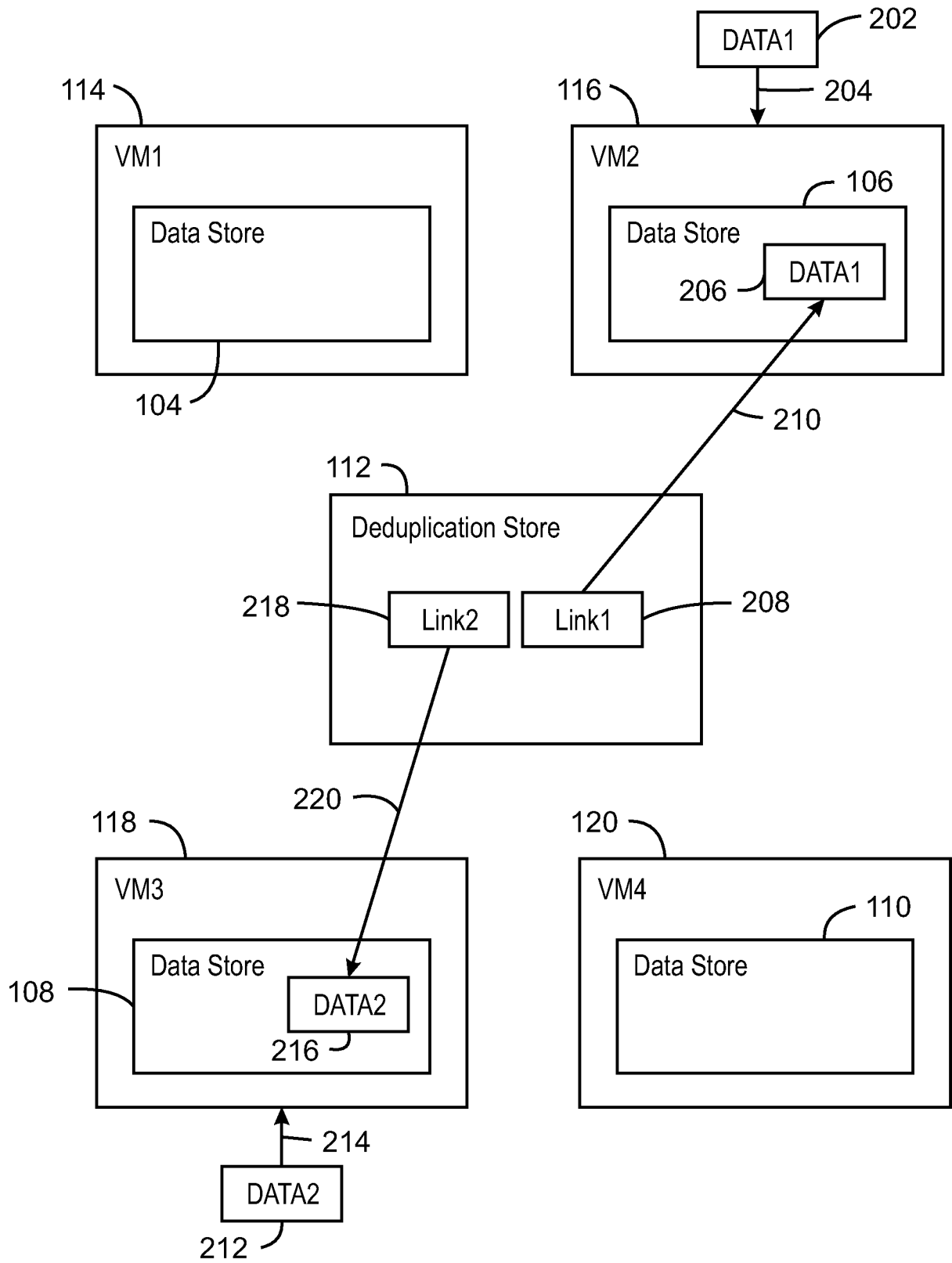
1/6



100
FIG. 1

⌕

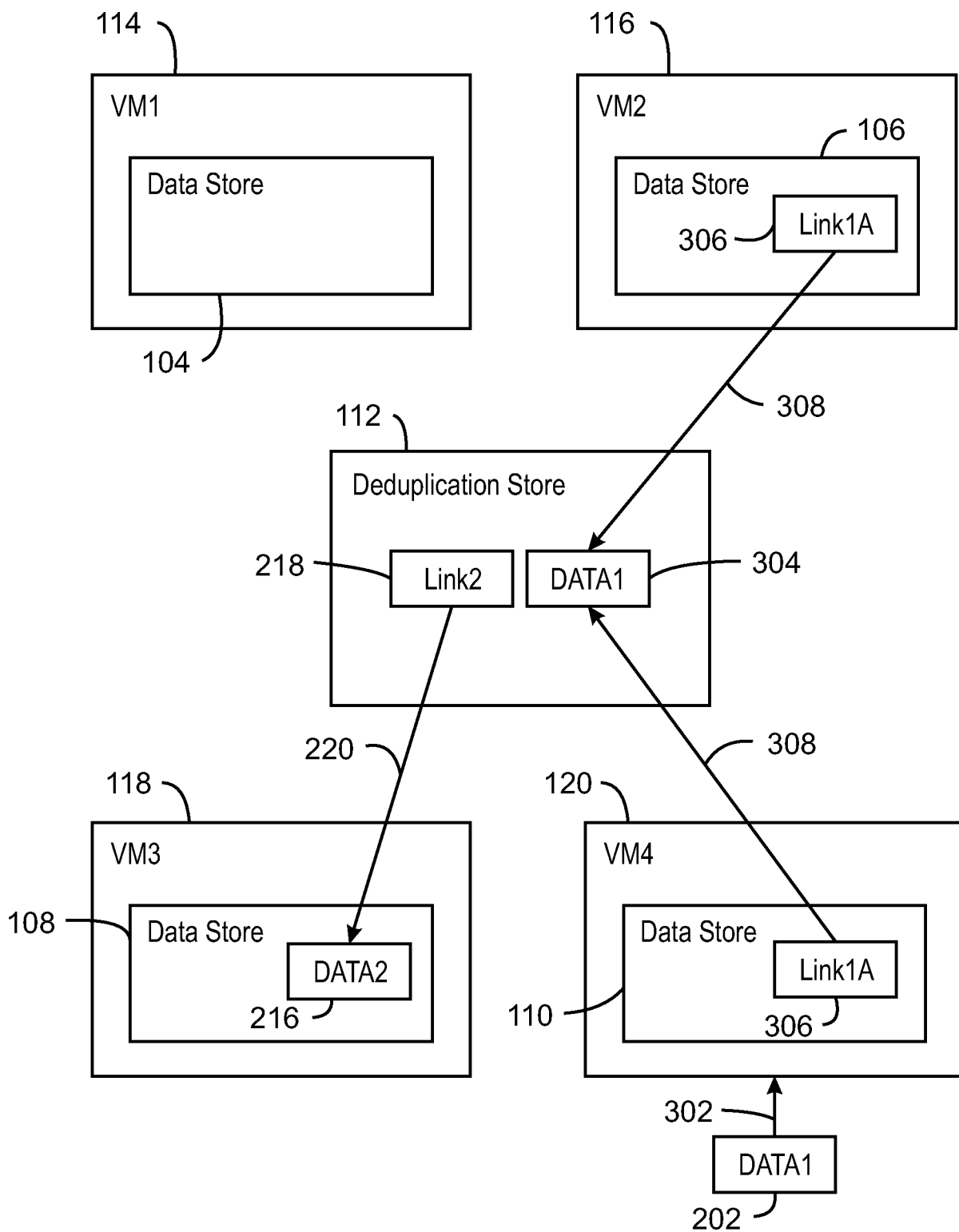
2/6



200
FIG. 2

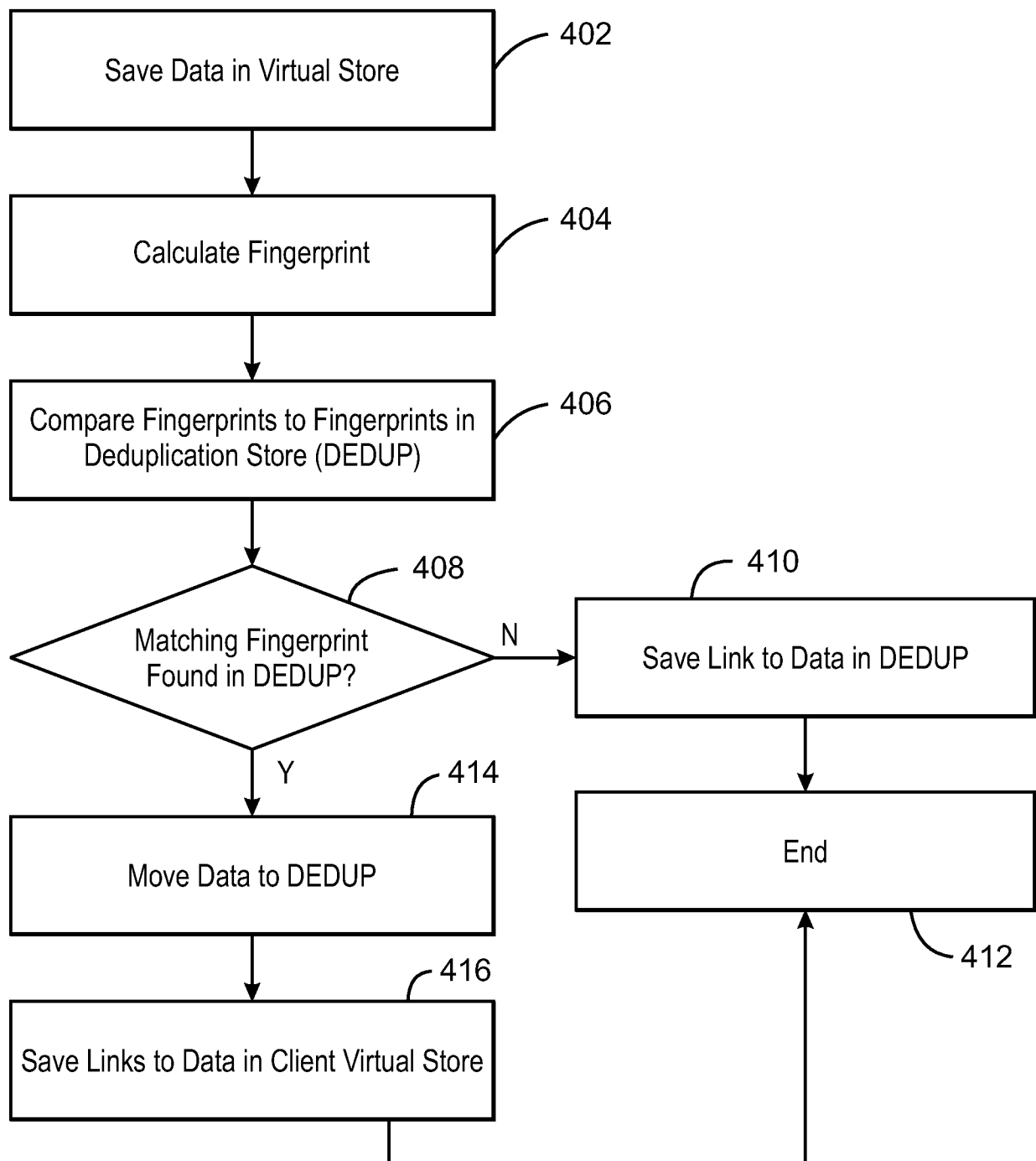
⌕

3/6



300
FIG. 3

4/6



400
FIG. 4



5/6

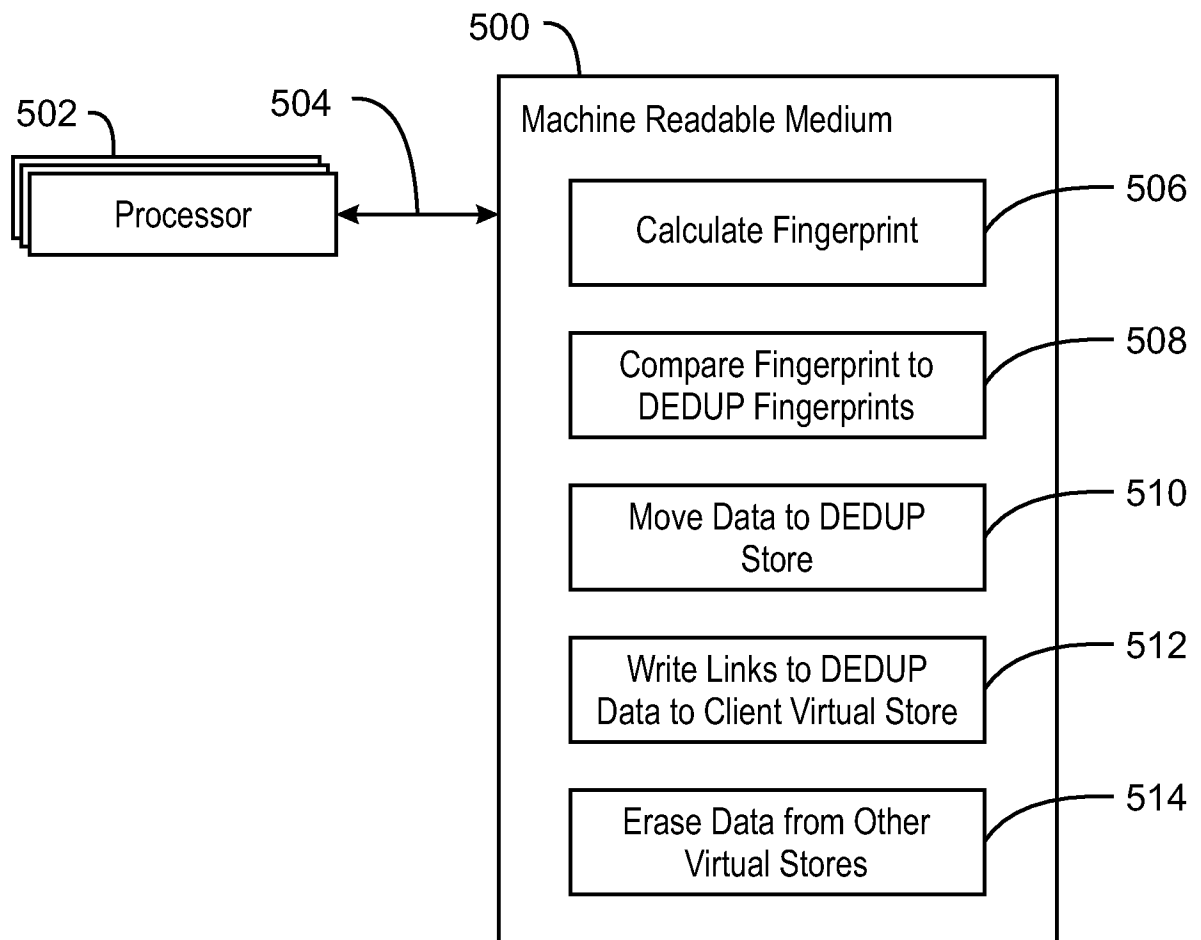


FIG. 5A



6/6

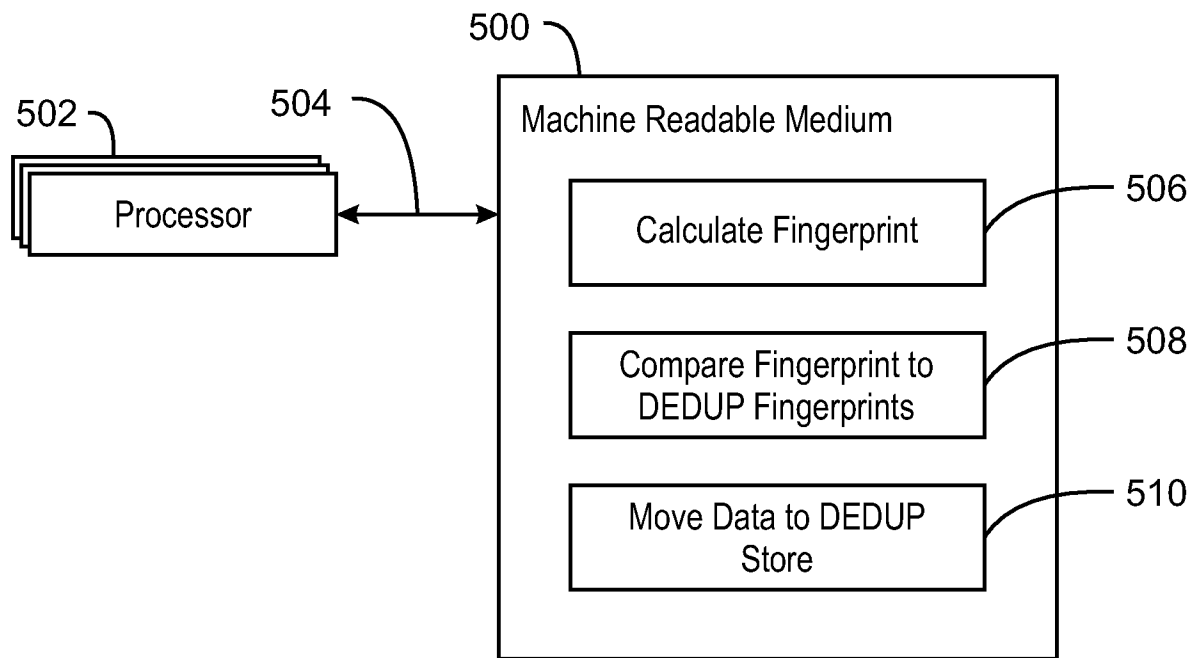


FIG. 5B

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/042831**A. CLASSIFICATION OF SUBJECT MATTER****G06F 11/14(2006.01)i, G06F 21/32(2013.01)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 11/14; G06F 12/06; G06F 12/12; G06F 17/00; G06F 17/30; G06F 7/00; G06F 12/00; G06F 11/00; G06F 21/32

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: duplication store, calculating, fingerprint, client data store, back reference, 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 2013-0086006 A1 (JOHN COLGROVE et al.) 04 April 2013 See paragraphs [0028], [0048], [0057]–[0059], and [0096]; claims 1, 8, and 15; and figures 1–2.	1–15
Y	US 2011-0131390 A1 (KIRAN SRINIVASAN et al.) 02 June 2011 See paragraphs [0102], [0129], and [0133]–[0134]; claim 1; and figures 2 and 13A–13B.	1–15
A	US 2013-0013865 A1 (CHETAN VENKATESH et al.) 10 January 2013 See paragraphs [0028]–[0031] and figure 3A.	1–15
A	US 8,898,114 B1 (DAVID ALLEN FEATHERGILL et al.) 25 November 2014 See column 8, lines 26–46; and figure 3.	1–15
A	WO 2012-173859 A2 (NETAPP, INC.) 20 December 2012 See paragraphs [0067]–[0069] and figure 8.	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

25 April 2016 (25.04.2016)

Date of mailing of the international search report

26 April 2016 (26.04.2016)

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. +82-42-481-8578

Authorized officer

NHO, Ji Myong

Telephone No. +82-42-481-8528



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/042831

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2013-0086006 A1	04/04/2013	US 8930307 B2 WO 2013-049319 A1	06/01/2015 04/04/2013
US 2011-0131390 A1	02/06/2011	US 7908436 B1 US 8145843 B2	15/03/2011 27/03/2012
US 2013-0013865 A1	10/01/2013	US 2013-013844 A1 US 2014-237181 A1 US 2014-237182 A1 US 2014-237183 A1 US 8732401 B2 US 8868884 B2 US 8874851 B2 US 8874877 B2 US 8996800 B2	10/01/2013 21/08/2014 21/08/2014 21/08/2014 20/05/2014 21/10/2014 28/10/2014 28/10/2014 31/03/2015
US 8898114 B1	25/11/2014	None	
WO 2012-173859 A2	20/12/2012	EP 2721495 A2 EP 2721495 A4 US 2012-0323860 A1 US 9292530 B2 WO 2012-173859 A3	23/04/2014 26/08/2015 20/12/2012 22/03/2016 25/04/2013