



(51) International Patent Classification:

H04L 29/08 (2006.01) *G06F 17/30* (2006.01)
G06F 11/10 (2006.01) *G06F 3/06* (2006.01)

(21) International Application Number:

PCT/US2017/049507

(22) International Filing Date:

31 August 2017 (31.08.2017)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

15/394,559 29 December 2016 (29.12.2016) US

(71) Applicant: **WESTERN DIGITAL TECHNOLOGIES, INC.** [US/US]; 5601 Great Oaks Parkway, San Jose, CA 95119 (US).

(72) Inventors: **DE KEYSER, Koen**; Western Digital Technologies, INC., 5601 Great Oaks Parkway, San Jose, CA 95119 (US). **DE SCHRIJVER, Frederik, Jacqueline, Luc**; 5601 Great Oaks Parkway, San Jose, CA 95119 (US). **BLYWEERT, Stijn**; Western Digital Technologies, INC., 5601 Great Oaks Parkway, San Jose, CA 95119 (US).

(74) Agent: **BOHN, Michel, E.** et al.; Patent Law Works LLP, 201 S. Main Street, Suite 250, Salt Lake City, UT 84111 (US).

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ,

(54) Title: IMPROVED HIERARCHIC STORAGE POLICY FOR DISTRIBUTED OBJECT STORAGE SYSTEMS

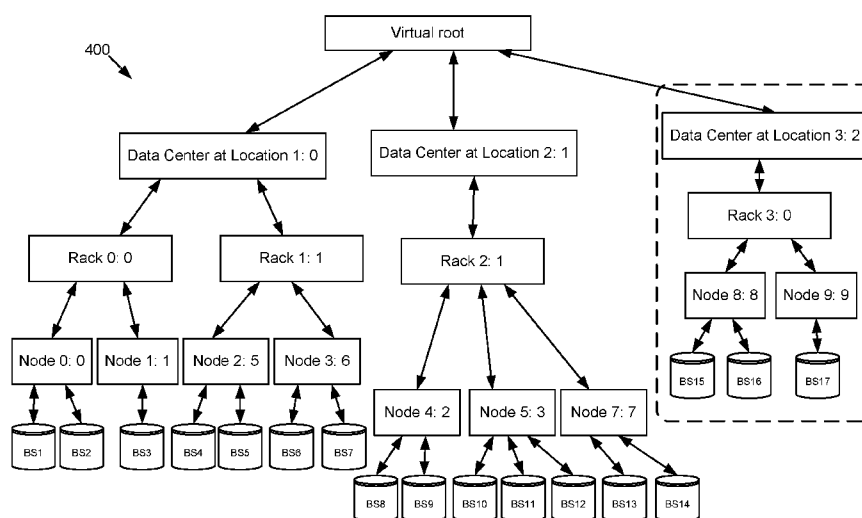


Figure 4

(57) **Abstract:** Techniques for distributing data in a distributed data storage system using a hierarchy rule that is generated based on a spreading policy and a set of tolerable failures specified by a user in absence of system deployment information are disclosed. The system includes a controller node which receives a request including a spreading policy and a protection level for spreading a first data object. The controller node determines a hierarchy rule corresponding to the spreading policy based on the protection level. The controller node distributes the first data object in the system using the hierarchy rule and the spreading policy. The controller node receives a reconfiguration of system deployment. The controller node distributes a second data object in the system based on providing protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object.



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).

Published:

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

IMPROVED HIERARCHIC STORAGE POLICY FOR DISTRIBUTED OBJECT STORAGE SYSTEMS

INVENTORS:

5 KOEN DE KEYSER;
FREDERIK JACQUILINE LUC DE SCHRIJVER; AND
STIJN BLYWEERT;

BACKGROUND

- 10 [0001] The present disclosure relates to a distributed data storage system. In particular, the present disclosure relates to distributing data in the distributed data storage system using a hierarchy rule that is generated based on a spreading policy and a set of tolerable failures specified by a user without knowledge of the deployment of the distributed data storage system.
- 15 [0002] There are many storage systems for storing data in a reliable way, for example, using redundancy. Some data distribution algorithms used in the storage systems even allow a user to define a protection level by describing what kind of failure scenarios can be tolerated, such that data can still be recovered even after such a failure occurs. However, such description is impacted by the layout of the storage system and may cause the storage
- 20 system to malfunction when entities (e.g., data storage devices, storage nodes, racks, data centers) are added to or removed from the system. For example, if a new entity is added to the system, every write of new data and every reconstruction of old data (e.g., the data already stored in the system) would cause the data to be spread out over all entities of the system including the newly added entity. This is problematic because the new entity and the
- 25 old entities are not the same size, and therefore the benefits of these entities are not the same. This is further problematic because the old data may have a different protection guarantee than the new data.

SUMMARY

- 30 [0003] The present disclosure relates to systems and methods for distributing data in a distributed data storage system using a hierarchy rule that is generated based on a spreading policy and a set of tolerable failures specified by a user in absence of deployment information of the distributed data storage system.

[0004] According to one innovative aspect, the subject matter described in this disclosure may be embodied in computer-implemented methods that include receiving a request for spreading data of a first data object, the request including a spreading policy and a protection level, the protection level including a set of tolerable failures from which the first data object can be recovered; determining a hierarchy rule corresponding to the spreading policy based on the protection level; distributing the data of the first data object in a distributed object storage system using the hierarchy rule and the spreading policy; receiving a reconfiguration of a hierarchical deployment of the distributed object storage system, the reconfiguration including adding one or more entities to the distributed object storage system; and distributing data of a second data object in the distributed object storage system based on providing protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object, the protection including recovering the data of both the first data object and the second data object when the set of tolerable failures occur.

[0005] Other implementations of one or more of these aspects include corresponding systems, apparatus, and computer programs, configured to perform the actions of the methods, encoded on computer storage devices. It should be understood that the language used in the present disclosure has been principally selected for readability and instructional purposes, and not to limit the scope of the subject matter disclosed herein.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] The techniques of the present disclosure are illustrated by way of example, and not by way of limitation in the figures of the accompanying drawings in which like reference numerals are used to refer to similar elements.

[0007] Figure 1 is a high-level block diagram illustrating an example distributed object storage system.

[0008] Figure 2 is a block diagram illustrating an example controller node of the distributed object storage system configured to implement the techniques introduced herein.

[0009] Figure 3 is a block diagram illustrating an example hierarchic tree structure of the distributed object storage system.

[0010] Figure 4 is a block diagram illustrating an example hierarchic tree structure of the distributed object storage system reconfigured based on the hierarchic tree structure depicted in Figure 3.

[0011] Figure 5 is a flowchart of an example method for generating a hierarchy rule, according to the techniques described herein.

[0012] Figures 6A and 6B are a flowchart of an example method for distributing a data object in the distributed object storage system using a hierarchy rule and a spreading policy, according to the techniques described herein.

DETAILED DESCRIPTION

[0013] Systems and methods for distributing data in a distributed data storage system using a hierarchy rule that is generated based on a spreading policy and a set of tolerable failures specified by a user without knowledge of the deployment of the distributed data storage system are described below. While the systems and methods of the present disclosure are described in the context of a particular system architecture, it should be understood that the systems and methods can be applied to other architectures and organizations of hardware.

[0014] Figure 1 is a high-level block diagram illustrating an example distributed object storage system 100 that is accessed by an application 102. In some embodiments, the application 102 is a software application running on a computing device that interacts with the system 100. The computer device may be, for example, a laptop computer, a desktop computer, a tablet computer, a mobile telephone, a personal digital assistant, a mobile email device, a portable game player, a portable music player, a television with one or more processors embedded therein or coupled thereto or any other electronic device capable of making requests to the system 100 and receiving responses from the system 100. In other embodiments, the application 102 comprises a file system that enables a general purpose software application to interface with the system 100 or an Application Programming Interface library.

[0015] The application 102 provides the data for storage in the system 100. The application 102 also requests the data stored in the system 100. For example, the application 102 may be a file transfer application that requests for storing a first set of data in the system 100 and retrieving a second set of data from the system 100. In some embodiments, the data is in the form of a data object. The data object comprises the data (e.g., 128-megabyte binary data) and a data object identifier. The data object identifier is a universally unique identifier used for identifying and addressing the data object. Storing data in the form of a data object, also referred to as object storage, is more advantageous than conventional file or block based storage on scalability and flexibility, which are of particular importance to large scale redundant storage in a distributed object storage system as shown in Figure 1.

[0016] The distributed object storage system 100 as depicted in Figure 1 includes a controller node 104, and storage nodes 106a...106n, 108a...108n, and 110a...110n. The controller node 104 may be a computing device configured to make some or all of the storage space for storage of the data provided by the application 102. For example, the controller node 104 generates rules for distributing data of a data object based on user input and determines where to store the data of the data object based on the rules. In some embodiments, the controller node 104 is physically located at a data center, where the controller node 104 along with a plurality of storage nodes 106a...106n, 108a...108n, and 110a...110n are arranged in modular racks as described below.

[0017] The storage nodes 106a...106n, 108a...108n, and 110a...110n are computer devices configured to store the data. For example, the storage nodes 106a...106n, 108a...108n, and 110a...110n comprise a plurality of storage elements (e.g., data storage devices or blockstores) for storing the data. In some embodiments, the storage nodes 106a...106n, 108a...108n, and 110a...110n are divided into groups based on, for example, whether the storage nodes are housed in a single rack. In the example of Figure 1, the storage nodes 106a...106n are grouped into rack 112, the storage nodes 108a...108n are grouped into rack 114, and the storage nodes 110a...110n are grouped into rack 116. In Figure 1, the controller node 104 is also located in rack 114 as indicated by the dash-lined box of rack 114. The racks can be geographically dispersed across different data centers, for example, racks 112 and 114 can be located at a data center in Europe, while rack 116 can be located at a data center in the United States. Although a single controller node 104 and storage nodes of three racks are shown in Figure 1, it should be understood that there may be any number of controller nodes 104, storage nodes, or racks. Also the storage nodes 106a...106n may be collectively referred as storage nodes 106. Similarly, the storage nodes 108a...108n and 110a...110n may be respectively referred as storage nodes 108 and 110.

[0018] The application 102, the controller node 104, and the storage nodes 106, 108, 110 are interconnected in a data communication network for distributing data of a data object. The data communication network can be a conventional type, wired or wireless, and may have numerous different configurations including a star configuration, token ring configuration, or other configurations. Furthermore, the data communication network may include a local area network (LAN), a wide area network (WAN) (e.g., the internet), and/or other interconnected data paths across which multiple devices (e.g., a computing device comprising the application 102, the controller node 104, the storage nodes, etc.) may

communicate. In some embodiments, the data communication network may be a peer-to-peer network. The data communication network may also be coupled with or include portions of a telecommunications network for sending data using a variety of different communication protocols. In some embodiments, the data communication network may include Bluetooth
5 (or Bluetooth low energy) communication networks or a cellular communications network for sending and receiving data, including via short messaging service (SMS), multimedia messaging service (MMS), hypertext transfer protocol (HTTP), direct data connection, wireless application protocol (WAP), email, etc.

[0019] Figure 2 is a block diagram illustrating an example controller node 104 of the
10 distributed object storage system 100 in Figure 1. In the example of Figure 2, the controller node 104 includes a processor 202, a memory 204, a network interface (I/F) module 206, and an optional storage element interface 208. The components of the controller node 104 are communicatively coupled to a bus or software communication mechanism 220 for communication with each other.

15 [0020] The processor 202 may include an arithmetic logic unit, a microprocessor, a general purpose controller or some other processor array to perform computations and provide electronic display signals to a display device. In some embodiments, the processor 202 is a hardware processor having one or more processing cores. The processor 202 is coupled to the bus 220 for communication with the other components. Processor 202
20 processes data signals and may include various computing architectures including a complex instruction set computer (CISC) architecture, a reduced instruction set computer (RISC) architecture, or an architecture implementing a combination of instruction sets. Although only a single processor is shown in the example of Figure 2, multiple processors and/or processing cores may be included. It should be understood that other processor
25 configurations are possible.

[0021] The memory 204 stores instructions and/or data that may be executed by the processor 202. In the illustrated embodiment, the memory 204 includes an encoding module 212, a rules engine 214, a spreading module 216, and a user interface engine 218. The memory 204 is coupled to the bus 220 for communication with the other components of the
30 controller node 104. The instructions and/or data stored in the memory 204 may include code for performing the techniques described herein. The memory 204 may be, for example, non-transitory memory such as a dynamic random access memory (DRAM) device, a static random access memory (SRAM) device, flash memory, or some other memory device. In

some embodiments, the memory 204 also includes a non-volatile memory or similar permanent storage device and media, for example, a hard disk drive, a floppy disk drive, a compact disc read only memory (CD-ROM) device, a digital versatile disc read only memory (DVD-ROM) device, a digital versatile disc random access memories (DVD-RAM) device, a digital versatile disc rewritable (DVD-RW) device, a flash memory device, or some other non-volatile storage device.

[0022] The network interface module 206 is configured to connect the controller node 104 to a data communication network. For example, the network interface module 208 may enable communication through one or more of the internet, cable networks, and wired networks. The network interface module 206 links the processor 202 to the data communication network that may in turn be coupled to other processing systems. The network interface module 206 also provides other conventional connections to the data communication network for distribution and/or retrieval of data objects (e.g., files and/or media objects) using standard network protocols such as Transmission Control Protocol/Internet Protocol (TCP/IP), Hypertext Transfer Protocol (HTTP), Secure Hypertext Transfer Protocol (HTTPS), and Simple Mail Transfer Protocol (SMTP) as will be understood. In some embodiments, the network interface module 206 includes a transceiver for sending and receiving signals using WiFi, Bluetooth®, or cellular communications for wireless communication.

[0023] In some embodiments, the controller node 104 may include or be included in one of the storage nodes 106, 108, or 110 that performs both the function of a controller node and a storage node. In such a case, the controller node 104 includes a storage element interface 208 and one or more storage elements 210a...210n connected via the storage element interface 208 to perform the functions of a storage node. For example, the storage element interface 208 may comprise a Serial Advanced Technology Attachment (SATA) interface or a Small Computer System Interface (SCSI) for connecting the storage elements 210a...210n (e.g., ten 2TB SATA-II disk drives) to other components of the controller node 104. The storage element interface 208 is configured to control the reading and writing of data to/from the storage elements 210a...210n. For example, the controller node 104 can use the storage element interface 208 to retrieve the data requested by the application 102 from the storage elements 210a...210n that store the data.

[0024] The distributed object storage system 100 in Figure 1 includes redundant and independently operated storage elements 210 such that, if one particular storage element fails,

the function of the failed storage element can easily be taken on by another storage element. In addition, the types, capacity, manufacturers, hardware technology, storage interfaces, etc. of the storage elements can be different based on the storage elements being redundant and independently operated, which benefits the scalability and flexibility of the distributed object storage system 100. For example, a storage element can be easily added or removed without correlating to other storage elements already in use in the distributed object storage system 100. When the system 100 is reconfigured (e.g., with the number of the storage elements changed), it is advantageous that the reconfiguration does not affect a protection level applied to the data that was already stored on the storage elements of the system 100. The protection level includes a set of failures that can be tolerated (“tolerable failures”), such that a data object can still be recovered even after such a failure occurs. For example, a protection level can provide that a data object stored on storage elements of storage nodes 106, 108, and 110 can be recovered from two concurrent data storage device failures. When new entities (e.g., data storage devices, storage nodes, racks, data centers, etc.) are added to the system 100, the distributing algorithm described herein guarantees that data objects stored before the new entities were added to the system 100 are still under the same level of protection (e.g., the data objects can still survive from two concurrent data storage device failures).

[0025] Software communication mechanism 220 may be an object bus (e.g., Common Object Request Broker Architecture (CORBA)), direct socket communication (e.g., TCP/IP sockets) among software modules, remote procedure calls, User Datagram Protocol (UDP) broadcasts and receipts, HTTP connections, function or procedure calls, etc. Further, any or all of the communication could be secure (e.g., Secure Shell (SSH), HTTPS, etc.). The software communication mechanism 220 can be implemented on any underlying hardware, for example, a network, the Internet, a bus, a combination thereof, etc.

[0026] In the illustrated embodiment of Figure 2, the controller node 104 comprises an encoding module 212, a rule engine 214, a spreading module 216, and a user interface engine 218. The encoding module 212, stored on memory 204 and configured to be executed by the processor 202, disassembles a data object received from the application 102 into a predetermined number of redundant sub-blocks or pieces to be stored across storage elements of the distributed object storage system. A distributed object storage system not only stores a data object on a plurality of storage elements, but also guarantees that the data object can be correctly retrieved when a certain number of the plurality of storage elements are unavailable (e.g., inaccessible, damaged). The encoding module 212 uses erasure coding techniques to

disassemble a data object to achieve acceptable reliability with considerably less overhead than a standard replication scheme.

[0027] In some embodiments, the encoding module 212 disassembles the data object into data pieces based on a spreading policy included in a storage request. The spreading policy may be defined as a spreading width (W) over a maximum concurrent failure tolerance (F). The spreading width indicates the number of data storage devices that store the pieces of the data object, where each data storage device stores a piece of data object. The maximum concurrent failure tolerance indicates a number of data storage devices that store the pieces of the data object that are allowed to fail concurrently. The spreading width and the maximum concurrent failure tolerance satisfy the relationship $W=K+F$, where K is a minimum spreading requirement (i.e., the minimum number of data storage devices that store pieces of the data object and are not allowed to fail to successfully recreate the data object).

[0028] The encoding module 212 using a W/F encoding scheme greatly reduces the overhead as compared to standard replication schemes. In some embodiments, the encoding module 212 communicates with the user interface engine 218 to receive a spreading policy from a user. The user specifies a spreading policy for an individual object or a group of objects. For example, a user may specify a spreading policy for a group of objects for simplicity. In some embodiments, a decoding module (not shown) assembles the data pieces of a data object based on a unique object identifier associated with each piece of the data object to recover the data object.

[0029] The rules engine 214, stored on memory 204 and configured to be executed by the processor 202, generates a hierarchy rule corresponding to a spreading policy. The hierarchy rule identifies a maximum number of data storage devices on each hierarchy level of a hierarchic tree structure of the distributed object storage system 100 for spreading the data of the data object to.

[0030] Referring now to Figure 3, an example hierarchic tree structure 300 of a distributed object storage system is shown. In the example of Figure 3, the three levels of a hierarchical configuration of the distributed object storage system form a tree structure. The virtual root in 302 is not part of the three levels as it represents an interface to access the data that is stored or retrieved to/from the distributed object storage system. At the bottom level of the hierarchy 300 (i.e., the node level), the storage elements (e.g., data storage devices or blockstores) are grouped according to their respective storage nodes. In the example of Figure 3, 14 data storage devices are grouped into 7 storage nodes at the node level. At the middle level of the hierarchy 300 (i.e., the rack level), the storage nodes are grouped

according to their respective racks. In the example of Figure 3, seven storage nodes are grouped into three racks at the rack level. At the top level of the hierarchy 300 (i.e., the data center level), the racks are grouped according to their respective data centers. In the example of Figure 3, three racks are grouped into two data centers at the data center level.

5 [0031] Each entity in the hierarchy 300 has a unique name and a unique identifier. An entity can be a data center, a rack, a storage node, a data storage device, etc.. For example, a data center 304 at the top level has a name “Data Center at Location 1” and an identifier “0,” a rack 306 at the middle level has a name “Rack 1” and an identifier “1,” and a storage node 308 at the bottom level has a name “Node 4” and an identifier “2.” A data
10 storage device has a hierarchy identifier comprising an array of integers. Each integer of a hierarchy identifier corresponds to an identifier of a data center, a rack and a node at a respective level of the hierarchy 300. These entities form a branch of the tree that ends up at the data storage device. For example, in Figure 3, the rightmost data storage device at the bottom level has a name “BS14” and a hierarchy identifier “[1,1,7].” The numbers “1,” “1,”
15 and “7” from left to right respectively correspond to identifiers of the “Data Center at Location 2,” “Rack 2” and “Node 7” from top to bottom of a branch that ends at the data storage device BS14. In Figure 3, the name and the identifier of an entity is separated by a colon.

[0032] A name of an entity is unique so that no two entities have a same name. An
20 identifier of a data center at the top level is unique so that no data centers have a same identifier. For example, data centers in Figure 3 are given unique identifiers 0 and 1. An identifier of a rack or a node is unique within a next higher level. In other words, racks are given unique identifiers within a data center. For example, in data center 0, the racks (i.e., Rack 0 and Rack 1) can have identifiers 0 and 1. In data center 1, the rack (i.e., Rack 2) can
25 also have identifier 1. So every rack has a unique number within a specific data center. Similarly, down to the node level, the nodes are given unique identifiers within a rack (i.e., every node has a unique number within a specific rack). The unique names and identifiers associated with entities provide layout information of a distributed object storage system, which is useful for distributing a data object in the system.

30 [0033] One way to distribute data across multiple entities of the hierarchic tree structure of a distributed object storage system is based on a hierarchy rule. In some embodiments, a hierarchy rule is in the form of $[n1, n2, n3]$, where the numbers $n1$, $n2$, and $n3$ respectively indicate a maximum number of data storage devices on each hierarchy level of the hierarchic tree structure (e.g., the data center level, the rack level, and the node level),

for spreading data of a data object to. For example, if the hierarchy rule is [100, 90, 1], it means that a maximum 1 data storage device per node, a maximum 90 data storage devices per rack, and a maximum 100 data storage devices per data center can be used to spread the data of a data object in a storage operation. The rule engine 214 generates the hierarchy rule
5 based on a spreading policy and a set of tolerable failures specified by a user.

[0034] The spreading policy W/F determines that W pieces of the data object need to be stored on W data storage devices with a tolerance for F concurrent data storage device failures. The protection level includes a set of tolerable failure scenarios specified by the user. The rule engine 214 translates the spreading policy into the number associated with
10 each hierarchy level in the hierarchy rule based on the protection level according to certain logic. The four main points of the logic are described below with an example spreading policy W/F=18/5 and an example hierarchy rule [100, 100, 1].

[0035] First, if the maximum number associated with a hierarchy level in the hierarchy rule is greater than W of the spreading policy, the rule engine 214 determines that
15 there is no restriction applied. Since the first 100 associated with the data center level and the second 100 associated with the rack level in the hierarchy rule [100, 100, 1] are greater than W=18 of the spreading policy, the rule engine 214 determines that no restriction is applied to the data center level and the rack level. In other words, there is no need to select multiple data centers or multiple racks to store the data. The 18 data pieces of a data object can be put
20 on a single data center and a single rack because each data center and each rack can store at most 100 data pieces.

[0036] Second, if the maximum number associated with a hierarchy level in the hierarchy rule is less than W of the spreading policy, the rule engine 214 determines that not
all data can be spread on a single entity of this hierarchy level. The number 1 associated with
25 the node level in the hierarchy rule [100, 100, 1] is less than W=18 of the spreading policy, which means that 18 pieces of the data object cannot be stored on a single node. In fact, the number 1 at the node level indicates that a maximum 1 data storage device per node can be used to store the data, and therefore the rule engine 214 determines that the data object should be distributed to 18 nodes, with each node storing one piece of data.

[0037] Third, if the maximum number associated with a hierarchy level in the hierarchy rule is greater than F of the spreading policy, the rule engine 214 determines that a
failure of a single entity at this hierarchy level cannot be tolerated. Since the number 100
30 associated with the data center level and the rack level in the hierarchy rule [100, 100, 1] is greater than F=5 of the spreading policy, a single data center failure and a single rack failure

would violate the spreading policy. If a distributed object storage system includes only a data center and/or a rack, in such case, the original data object cannot be recovered.

[0038] Fourth, if the maximum number associated with a hierarchy level in the hierarchy rule is less than F of the spreading policy, the rule engine 214 determines that, in addition to a failure of a single entity at this hierarchy level, at least one further concurrent data storage device failure can be tolerated. The number 1 in the hierarchy rule [100, 100, 1] is less than $F=5$ of the spreading policy, which means that a single node failure at the node level as well as at least an additional concurrent data storage device failure can be tolerated. $F=5$ of the spreading policy indicates that up to five pieces of data can be lost without affecting recovery of the data object using the remaining 13 pieces of data. The number 1 in the hierarchy rule [100, 100, 1] indicates that each node stores only one piece of data (e.g., on a single data storage device of the node). Therefore, in this case, in addition to a single node failure, up to four other nodes can fail until a total of five data pieces are lost. Assuming that the hierarchy rule is changed to [100, 100, 2]. The number 2 indicates that at most two data pieces can be stored on a single node (e.g., one piece each on two data storage devices of the node). In this case, in addition to a single node failure (i.e., two data storage device failures) up to three other data storage devices or one other node can fail until a total of five data pieces is lost. Two node failures lead to the loss of four data pieces, which is still tolerable. However, three node failures will cause six data pieces to be lost, which exceeds the maximum number of concurrent failures $F=5$ and results in incapability of reconstructing and retrieving the data object.

[0039] In some embodiments, the rule engine 214 receives a set of tolerable failure scenarios specified by a user. The rule engine 214 generates the hierarchy rule based on a spreading policy and the set of tolerable failures specified by the user. For example, if the spreading policy is $W/F=18/5$, the rule engine 214 may determine multiple hierarchy rules such as [100, 100, 1], [100, 2, 100], [100, 100, 3] using the logic described above. Responsive to receiving a set of tolerable failure scenarios, the rule engine 214 refines the hierarchy rules. If a user specifies that up to two racks can fail, the rule engine 214 determines the hierarchy rule to be [100, 2, 100], for example. Since the rack level element of the hierarchy rules (i.e., the second number 100) [100, 100, 1] and [100, 100, 3] is greater than $M=18$, the rule engine 214 determines that there is no restriction. The 18 data pieces of the data object can be stored on a single rack. However, there is no guarantee that two rack failures can be tolerated, as requested by the user in the failure scenario, since the entire data object (18 pieces) may be stored on a single rack. The rule engine 214 therefore determines

that the hierarchy rules [100, 100, 1] and [100, 100, 3] do not fulfill the user's requirement. On the other hand, since the product of the rack level element of the hierarchy rule [100, 2, 100] (i.e., 2) and the number of rack failures specified by the user (i.e., 2) is less than $F=5$ (i.e., 4), the rule engine 214 determines that the hierarchy rule [100, 2, 100] is sufficient when the user requires that up to two rack failures be tolerated.

[0040] It is advantageous that determining a spreading policy, a failure tolerance, and/or a hierarchy rule does not require information about the layout or deployment of a distributed object storage system. For example, a user may specify what kind of failures from which a data object can survive without knowledge of the hierarchical tree structure of the storage system and save the time associated with retrieving the extensive deployment information. In practice, the user may be provided a set of predefined policies that cover most common use cases and a description about what kind of failures that a data object may survive, which further simplifies the user's task for specifying failures and minimizes the user's need for knowledge about how the storage system works.

[0041] The spreading module 216, stored on memory 204 and configured to be executed by the processor 202, selects data storage devices in the distributed object storage system 100 to store a data object using a hierarchy rule and a spreading policy. In some embodiments, responsive to receiving a request for spreading a data object in a distributed object storage system 100, the rule engine 214 determines a hierarchy rule $[n1, n2, n3]$ based on a spreading policy W/F and a protection level included in the request, and transfers the hierarchy rule to the spreading module 216. The spreading module 216 identifies a hierarchical deployment configuration of the system 100, and determines whether the hierarchy rule is compatible with the hierarchical deployment configuration. Responsive to the hierarchy rule being compatible with the hierarchical deployment configuration, the spreading module 216 identifies which data storage devices in the system 100 should be used for storing the data and transfers the data to the identified data storage devices for storing.

[0042] As described above, given a number of data pieces for a data object W and a number of data pieces that can be lost F, and given a set of tolerable failure scenarios that the data object is able to survive, the rule engine 214 generates the maximum number of data centers, racks, and nodes used to store a data object (i.e., the hierarchy rule). The generation of the hierarchy rule does not relate to the hierarchical deployment configuration of a distributed object storage system. Therefore, it is possible that the hierarchy rule may not be compatible with the hierarchical deployment configuration. The spreading module 216 determines whether the hierarchy rule is compatible with the hierarchical deployment

configuration by determining whether each entity of a hierarchy level meets the maximum number of data storage devices associated with the hierarchy level without receiving information of data storage devices used at other entities of the hierarchy level. For example, a user selects a spreading policy $W/F=18/5$ that defines a data object be distributed into 18 pieces such that failure of up to five pieces can be tolerated and a protection level that defines that the failure of a single data center can be tolerated. As a result, the rule engine 214 determines that no more than 5 pieces of data can be put on any single data center, and creates a maximum number of 5 on the data center level of a hierarchy rule. The spreading module 216 receives this hierarchy rule and identifies that the actual hierarchical deployment configuration of the distributed object storage system includes only two data centers to fulfill this rule. Because of the maximum number of 5 on the data center level of the hierarchy rule, at most 10 data pieces can be stored between the two data centers, which is less than 18 data pieces selected in the spreading policy. The spreading module 216 therefore determines that the hierarchy rule is incompatible with the hierarchical deployment of the system because the entire data object cannot be stored according to the user selected protection level. When the spreading module 216 determines that the hierarchy rule corresponding to a spreading policy cannot be fulfilled by the hierarchical layout or deployment of the distributed object storage system, the spreading module 216 communicates with the user interface engine 218 to notify the user of the incompatibility and instruct the user to modify at least one of the spreading policy and the protection level such that the hierarchy rule is compatible with the hierarchical deployment configuration.

[0043] If the spreading module 216 determines that the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system, the spreading module 216 selects the data storage devices for storing the pieces of the data object using the hierarchy rule and the spreading policy, and distributes the pieces to the selected data storage devices. Referring back to Figure 3, the spreading module 216 identifies that the hierarchical deployment of the system as shown in Figure 3 includes two data centers, three racks, and seven nodes that comprise 14 data storage devices. Suppose that a user requests to spread a first data object using a spreading policy $W/F=6/3$. The user also specifies a protection level that the first data object survive a single node failure. Based on the spreading policy and the protection level, the rule engine 214 determines that the hierarchy rule is [100, 100, 2]. The number 2 associated with the node level in [100, 100, 2] indicates that each node can use up to two data storage devices to store two pieces of data. The spreading module 216 determines that multiple ways to store the pieces of the first data object are

possible. For example, the spreading module 216 may determine, to use both data centers at the data center level, the three racks at the rack level, and any six of the seven nodes at the node level to store the six pieces of data with each node storing one piece of data.

Alternatively, the spreading module 216 may determine to use the two data centers at the data center level, two racks out of the three racks (e.g., Rack 0 and Rack 2) at the rack level, and any six data storage devices of four nodes at the node level to store the six pieces of data (e.g., BS1 and BS2 of Node 0, BS3 of Node 1, BS8 and BS9 of Node 4, and BS14 of Node 7).

[0044] This example illustrates one advantage of the distributing algorithm described herein, that is, a user can require the data to be stored under a certain protection level (e.g., failure scenarios) regardless of the actual layout of the storage system and where the data is stored. The spreading module 216 cooperates with other modules/engines to fulfil the user's requirement by actually distributing the data in the system and providing the required protection to the data. Another advantage of the distributing algorithm described herein is that expansion of the system 100 will not invalidate the protection level applied to the data already stored in the system 100.

[0045] Referring now to Figure 4, an example hierarchic tree structure 400 of a distributed object storage system reconfigured based on the hierarchic tree structure depicted in Figure 3 is shown. Compared with the deployment in Figure 3 and as shown in the dashed-line box, the distributed object storage system in Figure 4 is expanded to include additional entities: Data Center at Location 3, Rack 3, Node 8 and Node 9, and data storage devices BS15, BS16 and BS17. Continuing with the example above, a first data object was stored in the previous deployment of system 100 (as described by Figure 3) with a protection level that specified the first data object survive a single node failure. Now suppose that a user requests to store a second data object in the current deployment of the system 100 as shown in Figure 4. Responsive to the request, the spreading module 216 distributes the second data object in the system 100 of Figure 4 such that a protection level specified by the user is satisfied. More importantly, the spreading module 216 distributes the second data object without affecting the protection level initially applied to the first data object (i.e., the data of the first data object and the second data object can both recover from a single node failure). In fact, no matter what data (e.g., the second data object) is added or the original data needs to be reconstructed or repaired (e.g., the first data object is recovered from a data storage device failure), the reconfiguration of the system, including the addition, removal, and/or

replacement of one or more entities, does not affect the protection level already applied to existing data on the system.

[0046] In practice, a set of spreading policies, protection levels (e.g., failure scenarios), and hierarchy rules can be predefined for most common use cases of a distributed object storage system when the system is installed. To reduce the incompatibility between the hierarchy rule and the actual deployment configuration and to minimize the information a user needs to know about the system configuration and work mechanism, the user may be provided the predefined policies and protection levels such that the user has a clear vision of what kind of failure scenarios the data object would survive.

[0047] A simple use case is to apply the hierarchy rule [100, 100, 1] in the system 100. In this case, the spreading module 216 distributes a single piece of data to a single data storage device per node to meet concurrent failure requirement F ($F \geq 1$) in the spreading policy W/F . There is no additional restriction for the data center level and the rack level. To satisfy this use case, the first two elements of the hierarchy rule (i.e., the data center and rack elements) can be any number that is greater than the spreading width W of the spreading policy W/F .

[0048] Another use case allows a data object to be reconstructed and recovered from a single node failure and two additional data storage device failures. Suppose the spreading policy is $W/F=18/5$, which means as much as 5 pieces of data can go missing and the remaining 13 pieces of data can be used to correctly reconstruct the data object. According to the protection policy, two data storage devices (whether they are on one node or across multiple nodes) are allowed to fail, which means that two pieces of data stored on the failed data storage devices may be lost. To ensure that the data object can be recovered when taking into account the two data storage device failures, the rule engine 214 determines that any single node can store at most three pieces of data. As a result, the rule engine 214 determines a hierarchy rule [100, 100, 3] that forces each node to store at most three pieces of data across three data storage devices. Losing a single node and two additional data storage devices mean losing at most three pieces of data plus two pieces of data. In total, up to five data pieces are lost, which is within the maximum tolerance $F=5$. The data object is therefore protected from a single node failure and two additional data storage device failures.

[0049] In one embodiment, the system 100 includes six nodes and each node has 98 data storage devices. When spreading the 18 data pieces across the six nodes using the hierarchy rule [100, 100, 3], the spreading module 216 can store these data pieces on a single data center/rack or multiple data centers/racks with no restriction applied on the data center

level and the rack level. To make the hierarchy rule [100, 100, 3] viable over the six nodes, the spreading module 216 uses every node of the six nodes and stores exactly 3 pieces of data on each node based on the maximum number 3 associated with the node level.

[0050] In yet another use case, a user selects a spreading policy of 18 pieces of data and tolerance for loss of 8 pieces of data, i.e., $W/F=18/8$. The protection level includes a single data center failure and two additional data storage devices. The system 100 includes three different data centers. Each data center has six nodes with each node having 98 data storage devices. The rule engine 214 determines the hierarchy rule to be [6, 100, 100] because six additional pieces of data can go missing when two pieces of data on two data storage devices are lost, i.e., each data center should not have more than six pieces of data. In implementing the hierarchy rule, the spreading module 216 spreads out the 18 pieces of data on all three data centers with each data center storing exactly six pieces of data. In some embodiments, the rule engine 214 may also determine the hierarchy rule to be [6, 100, 2]. Since each node can store at most two pieces of data and each data center stores six pieces of data, the spreading module 216 spreads six pieces of data to at least three nodes of each data center with each node storing one or two pieces of data.

[0051] The user interface engine 218, stored on memory 204 and configured to be executed by the processor 202, generates graphical data for displaying a user interface. In some embodiments, the user interface engine 218 communicates with the rule engine 214 and the spreading module 216 to generate graphical data for displaying predefined spreading policies and protection levels including a set of tolerable failure scenarios to a user. In other embodiments, the user interface engine 218 generates a user interface for receiving a selection of a spreading policy and a protection level from a user. In some other embodiments, the user interface engine 218 receives instructions from the rule engine 214 to generate a user interface to notify the user to modify a spreading policy, a protection policy, and/or a set of tolerable failure scenarios. The user interface engine 218 may also communicate with the spreading module 216 to generate a user interface to notify the user of incompatibility between a hierarchy rule and an actual hierarchical deployment configuration of a distributed object storage system, and instruct the user to modify a spreading policy and/or a protection level such that a hierarchy rule that is generated based on the modified spreading policy and/or the protection level is compatible with the hierarchical deployment configuration.

[0052] Figure 5 is a flowchart of an example method 500 for generating a hierarchy rule. At 502, the rule engine 214 receives a spreading policy W/F for spreading a data object.

At 504, the rule engine 214 receives a set of tolerable failures from which the data object can be recovered.

[0053] At 506, the rule engine 214 determines whether the spreading policy can be translated to a number associated with a hierarchy level based on the set of tolerable failures, the number indicating a maximum number of data storage devices on the hierarchy level for spreading the data to. As shown in Figures 3 and 4, a distributed object storage system has a hierarchic tree structure that includes a data center level, a rack level and a node level. For example, a user may specify the spreading policy W/F=15/3 for spreading a data object, and require that the data object be correctly restored after two node failures with each node storing at least two data pieces. However, the rule engine 214 determines that at most one failure can be tolerated if each node stores two data pieces. In such case, the rule engine 214 determines that the spreading policy cannot be translated to the number associated with the hierarchy level, the method 500 returns to step 502.

[0054] Responsive to determining that the spreading policy can be translated to a number associated with the hierarchy level, at 508, the rule engine 214 identifies the number associated with the hierarchy level. For example, if a user specifies that the spreading policy is W/F=18/5 and up to two racks can fail, the rule engine 214 determines that at most two data pieces can be put on each rack to satisfy the requirement of F=5 and two rack failure tolerance, and identifies that the number associated with the rack level is 2.

[0055] At 510, the rule engine 214 determines whether the number associated with each level of a hierarchical configuration, for example, the data center level, the rack level and the node level as shown in Figures 3 and 4, is identified. If the number associated with each level of the hierarchical configuration has not been identified, the method returns to step 506 to determine the number. If the number associated with each level of the hierarchical configuration has been identified, at 512, the rule engine 214 generates a hierarchy rule corresponding to the spreading policy based on the identified number associated with each hierarchy level of the hierarchical configuration.

[0056] Figures 6A and 6B depict a flowchart of an example method 600 for distributing a data object in the distributed object storage system using a hierarchy rule and a spreading policy. At 602, the rule engine 214 receives a request for spreading a first data object, the request including a spreading policy and a protection level, the protection level including a set of tolerable failures from which the first data can be recovered. At 604, the rule engine 214 determines a hierarchy rule corresponding to the spreading policy based on

the protection level. The rule engine 214 sends the hierarchy rule to the spreading module 216.

[0057] At 606, the spreading module 216 identifies a hierarchical deployment configuration of a distributed object storage system. For example, the spreading module 216 identifies that the distributed object storage system in Figure 3 includes two data centers, three racks and seven nodes that comprise 14 data storage devices. At 608, the spreading module 216 determines whether the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system. Responsive to determining that the hierarchy rule is incompatible with the hierarchical deployment configuration of the distributed object storage system, the spreading module 216 notifies a user of the incompatibility and instructs the user to modify the spreading policy and/or the protection level. For example, the rule engine 214 determines a maximum number of 5 on the data center level of a hierarchy rule when a user selects a spreading policy W/F=18/5 to distribute a first data object and specifies that the first data object should be protected from a single data center failure. However, if the spreading module 216 identifies that the actual hierarchical deployment configuration of the distributed object storage system includes only two data centers, the spreading module 216 determines that this hierarchy rule cannot be fulfilled because at most 10 data pieces can be stored on the two data centers (e.g., 5 on each data center), which is less than 18 data pieces selected in the spreading policy. The spreading module 216 notifies the user to change either the spreading policy or the protection level.

[0058] Responsive to determining that the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system, the spreading module 216 distributes the data of the first data object in the distributed object storage system using the hierarchy rule and the spreading policy.

[0059] Referring now to Figure 6B, at 616, the rule engine 214 receives a reconfiguration of the hierarchical deployment of the distributed object storage system, the reconfiguration including adding one or more entities to the distributed object storage system. For example, the system is updated to include a third data center as depicted in the example of Figure 4. At 618, the rule engine 214 receives a request for spreading data of a second data object. At 620, the spreading module 216 distributes the data of the second data object in the distributed object system, wherein distributing the data of the second data object provides protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object, the protection including recovering the data of both the first data object and the second data object when the

set of tolerable failures occur. In other words, the layout change of the distributed object storage system does not invalidate the protection level applied to the data already stored in the system.

[0060] Systems and methods for distributing a data object in a distributed data storage system using a hierarchy rule that is generated based on a spreading policy and a set of tolerable failures specified by a user in absence of deployment information of the distributed data storage system has been described. In the above description, for purposes of explanation, numerous specific details were set forth. It will be apparent, however, that the disclosed technologies can be practiced without any given subset of these specific details. In other instances, structures and devices are shown in block diagram form. For example, the disclosed technologies are described in some implementations above with reference to user interfaces and particular hardware. Moreover, the technologies disclosed above primarily in the context of on line services; however, the disclosed technologies apply to other data sources and other data types (e.g., collections of other resources for example images, audio, web pages).

[0061] Reference in the specification to “one implementation” or “an implementation” means that a particular feature, structure, or characteristic described in connection with the implementation is included in at least one implementation of the disclosed technologies. The appearances of the phrase “in one implementation” in various places in the specification are not necessarily all referring to the same implementation.

[0062] Some portions of the detailed descriptions above were presented in terms of processes and symbolic representations of operations on data bits within a computer memory. A process can generally be considered a self-consistent sequence of steps leading to a result. The steps may involve physical manipulations of physical quantities. These quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. These signals may be referred to as being in the form of bits, values, elements, symbols, characters, terms, numbers or the like.

[0063] These and similar terms can be associated with the appropriate physical quantities and can be considered labels applied to these quantities. Unless specifically stated otherwise as apparent from the prior discussion, it is appreciated that throughout the description, discussions utilizing terms for example “processing” or “computing” or “calculating” or “determining” or “displaying” or the like, may refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system’s

registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0064] The disclosed technologies may also relate to an apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may include a general-purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable storage medium, for example, but is not limited to, any type of data storage device including floppy disks, optical disks, CD-ROMs, and magnetic disks, read-only memories (ROMs), random access memories (RAMs), erasable programmable read-only memories (EPROMs), electrically erasable programmable read-only memories (EEPROMs), magnetic or optical cards, flash memories including universal serial bus (USB) keys with non-volatile memory or any type of media suitable for storing electronic instructions, each coupled to a computer system bus.

[0065] The disclosed technologies can take the form of an entirely hardware implementation, an entirely software implementation or an implementation containing both hardware and software elements. In some implementations, the technology is implemented in software, which includes but is not limited to firmware, resident software, microcode, etc.

[0066] Furthermore, the disclosed technologies can take the form of a computer program product accessible from a non-transitory computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any instruction execution system. For the purposes of this description, a computer-usable or computer-readable medium can be any apparatus that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

[0067] A computing system or data processing system suitable for storing and/or executing program code will include at least one processor (e.g., a hardware processor) coupled directly or indirectly to memory elements through a system bus. The memory elements can include local memory employed during actual execution of the program code, bulk storage, and cache memories which provide temporary storage of at least some program code in order to reduce the number of times code must be retrieved from bulk storage during execution.

[0068] Input/output (I/O) devices (including but not limited to keyboards, displays, pointing devices, etc.) can be coupled to the system either directly or through intervening I/O controllers.

[0069] Network adapters may also be coupled to the system to enable the data
5 processing system to become coupled to other data processing systems or remote printers or storage devices through intervening private or public networks. Modems, cable modems and Ethernet cards are just a few of the currently available types of network adapters.

[0070] Finally, the processes and displays presented herein may not be inherently
10 related to any particular computer or other apparatus. Various general-purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatus to perform the required method steps. The required structure for a variety of these systems will appear from the description below. In addition, the disclosed technologies were not described with reference to any particular programming language. It will be appreciated that a variety of programming languages may be used to
15 implement the teachings of the technologies as described herein.

[0071] The foregoing description of the implementations of the present techniques
and technologies has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the present techniques and technologies to the precise form disclosed. Many modifications and variations are possible in light of the above
20 teaching. It is intended that the scope of the present techniques and technologies be limited not by this detailed description. The present techniques and technologies may be implemented in other specific forms without departing from the spirit or essential characteristics thereof. Likewise, the particular naming and division of the modules, routines, features, attributes, methodologies and other aspects are not mandatory or significant, and the
25 mechanisms that implement the present techniques and technologies or its features may have different names, divisions and/or formats. Furthermore, the modules, routines, features, attributes, methodologies and other aspects of the present technology can be implemented as software, hardware, firmware or any combination of the three. Also, wherever a component, an example of which is a module, is implemented as software, the component can be
30 implemented as a standalone program, as part of a larger program, as a plurality of separate programs, as a statically or dynamically linked library, as a kernel loadable module, as a device driver, and/or in every and any other way known now or in the future in computer programming. Additionally, the present techniques and technologies are in no way limited to implementation in any specific programming language, or for any specific operating system

or environment. Accordingly, the disclosure of the present techniques and technologies is intended to be illustrative, but not limiting.

WHAT IS CLAIMED IS:

1. A computer-implemented method comprising:
receiving, by one or more processors of a controller node, a request for spreading data of a first data object, the request including a spreading policy and a protection level, the protection level including a set of tolerable failures from which the first data object can be recovered;
determining, by the one or more processors, a hierarchy rule corresponding to the spreading policy based on the protection level;
distributing, by the one or more processors, the data of the first data object in a distributed object storage system using the hierarchy rule and the spreading policy;
receiving, by the one or more processors, a reconfiguration of a hierarchical deployment of the distributed object storage system, the reconfiguration including adding one or more entities to the distributed object storage system;
and
distributing, by the one or more processors, data of a second data object in the distributed object storage system based on providing protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object, the protection including recovering the data of both the first data object and the second data object when the set of tolerable failures occur.
2. The computer-implemented method of claim 1, wherein determining the hierarchy rule corresponding to the spreading policy comprises:
determining whether the spreading policy can be translated to a number associated with a hierarchy level based on the set of tolerable failures, the number indicating a maximum number of data storage devices on the hierarchy level for spreading the data of the first object to;
in response to determining that the spreading policy can be translated to a number associated with the hierarchy level, identifying the number;
determining whether the number associated with each hierarchy level of a hierarchical configuration has been identified; and

in response to determining that the number associated with each hierarchy level of the hierarchical configuration has been identified, generating the hierarchy rule corresponding to the spreading policy based on the identified number.

3. The computer-implemented method of claim 2, further comprising:
identifying a hierarchical deployment configuration of the distributed object storage system; and
determining whether the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system;
wherein distributing the data of the first data object in the distributed object storage system is based on determining that the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system.

4. The computer-implemented method of claim 3, wherein determining whether the hierarchy rule is compatible with the hierarchical deployment configuration of the storage system comprises determining whether each entity of the hierarchy level meets the maximum number of data storage devices associated with the hierarchy level without receiving information of data storage devices used at other entities of the hierarchy level, an entity including at least one of a storage node, a rack, and a data center.

5. The computer-implemented method of claim 3, further comprising in response to determining that the hierarchy rule is incompatible with the hierarchical deployment configuration of the distributed object storage system:
notifying a user; and
instructing the user to modify at least one of the spreading policy and the protection level.

6. The computer-implemented method of claim 2, wherein the hierarchy level comprises a level where the data storage devices are grouped according to respective storage nodes, a level where the storage nodes are grouped to respective racks, and a level where the racks are grouped according to respective data centers.

7. The computer-implemented method of claim 1, further comprising updating the spreading policy based on the second data object to obtain an updated spreading policy,

and wherein distributing the data of the second data object in the distributed object storage system uses the hierarchy rule and the updated spreading policy.

8. The computer-implemented method of claim 1, wherein the spreading policy comprises a desired spreading width and a maximum concurrent failure tolerance.

9. A distributed object storage system comprising:
a set of non-volatile data storage devices; and
a controller node having a memory and one or more processors configured to execute instructions stored on the memory, wherein the controller node is configured to perform operations comprising:
receiving a request for spreading data of a first data object, the request including a spreading policy and a protection level, the protection level including a set of tolerable failures from which the first data object can be recovered;
determining a hierarchy rule corresponding to the spreading policy based on the protection level;
distributing the data of the first data object among the set of non-volatile data storage devices in the distributed object storage system using the hierarchy rule and the spreading policy;
receiving a reconfiguration of a hierarchical deployment of the distributed object storage system, the reconfiguration including adding one or more entities to the distributed object storage system; and
distributing data of a second data object among the set of non-volatile data storage devices in the distributed object storage system based on providing protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object, the protection including recovering the data of both the first data object and the second data object when the set of tolerable failures occur.

10. The distributed object storage system of claim 9, wherein to determine the hierarchy rule corresponding to the spreading policy, the operations further comprise:

determining whether the spreading policy can be translated to a number associated with a hierarchy level based on the set of tolerable failures, the number indicating a maximum number of data storage devices on the hierarchy level for spreading the data of the first object to;

in response to determining that the spreading policy can be translated to a number associated with the hierarchy level, identifying the number;

determining whether the number associated with each hierarchy level of a hierarchical configuration has been identified; and

in response to determining that the number associated with each hierarchy level of the hierarchical configuration has been identified, generating the hierarchy rule corresponding to the spreading policy based on the identified number.

11. The distributed object storage system of claim 10, wherein the operations further comprise:

identifying a hierarchical deployment configuration of the distributed object storage system; and

determining whether the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system;

wherein distributing the data of the first data object in the distributed object storage system is based on determining that the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system.

12. The distributed object storage system of claim 11, wherein to determine whether the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system, the controller node is further configured to perform the operations comprising determining whether each entity of the hierarchy level meets the maximum number of data storage devices associated with the hierarchy level without receiving information of data storage devices used at other entities of the hierarchy level, an entity including at least one of a storage node, a rack, and a data center.

13. The distributed object storage system of claim 11, wherein the operations further comprise in response to determining that the hierarchy rule is incompatible with the hierarchical deployment configuration of the distributed object storage system:

notifying a user; and

instructing the user to modify at least one of the spreading policy and the protection level.

14. The distributed object storage system of claim 10, wherein the hierarchy level comprises a level where the data storage devices are grouped according to respective storage nodes, a level where the storage nodes are grouped to respective racks, and a level where the racks are grouped according to respective data centers.

15. The distributed object storage system of claim 9, wherein the spreading policy comprises a desired spreading width and a maximum concurrent failure tolerance.

16. A distributed object storage system comprising:

means for receiving a request for spreading data of a first data object, the request including a spreading policy and a protection level, the protection level including a set of tolerable failures from which the first data object can be recovered;

means for determining a hierarchy rule corresponding to the spreading policy based on the protection level;

means for distributing the data of the first data object among the set of non-volatile data storage devices in the distributed object storage system using the hierarchy rule and the spreading policy;

means for receiving a reconfiguration of a hierarchical deployment of the distributed object storage system, the reconfiguration including adding one or more entities to the distributed object storage system; and

means for distributing data of a second data object among the set of non-volatile data storage devices in the distributed object storage system based on providing protection of the protection level to the second data object without affecting protection of the same protection level applied to the first data object, the protection including recovering the data of both the first data object and the second data object when the set of tolerable failures occur.

17. The distributed object storage system of claim 16, wherein to determine the hierarchy rule corresponding to the spreading policy, the distributed object storage system further comprises:

means for determining whether the spreading policy can be translated to a number associated with a hierarchy level based on the set of tolerable failures, the number indicating a maximum number of data storage devices on the hierarchy level for spreading the data of the first object to;

in response to determining that the spreading policy can be translated to a number associated with the hierarchy level, means for identifying the number;

means for determining whether the number associated with each hierarchy level of a hierarchical configuration has been identified; and

in response to determining that the number associated with each hierarchy level of the hierarchical configuration has been identified, means for generating the hierarchy rule corresponding to the spreading policy based on the identified number.

18. The distributed object storage system of claim 17, further comprising:

means for identifying a hierarchical deployment configuration of the distributed object storage system; and

means for determining whether the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system;

wherein distributing the data of the first data object in the distributed object storage system is based on determining that the hierarchy rule is compatible with the hierarchical deployment configuration of the distributed object storage system.

19. The distributed object storage system of claim 18, further comprising:

means for notifying a user in response to determining that the hierarchy rule is incompatible with the hierarchical deployment configuration of the distributed object storage system; and

means for instructing the user to modify at least one of the spreading policy and the protection level.

20. The distributed object storage system of claim 17, wherein the hierarchy level comprises a level where the data storage devices are grouped according to respective storage nodes, a level where the storage nodes are grouped to respective racks, and a level where the racks are grouped according to respective data centers.

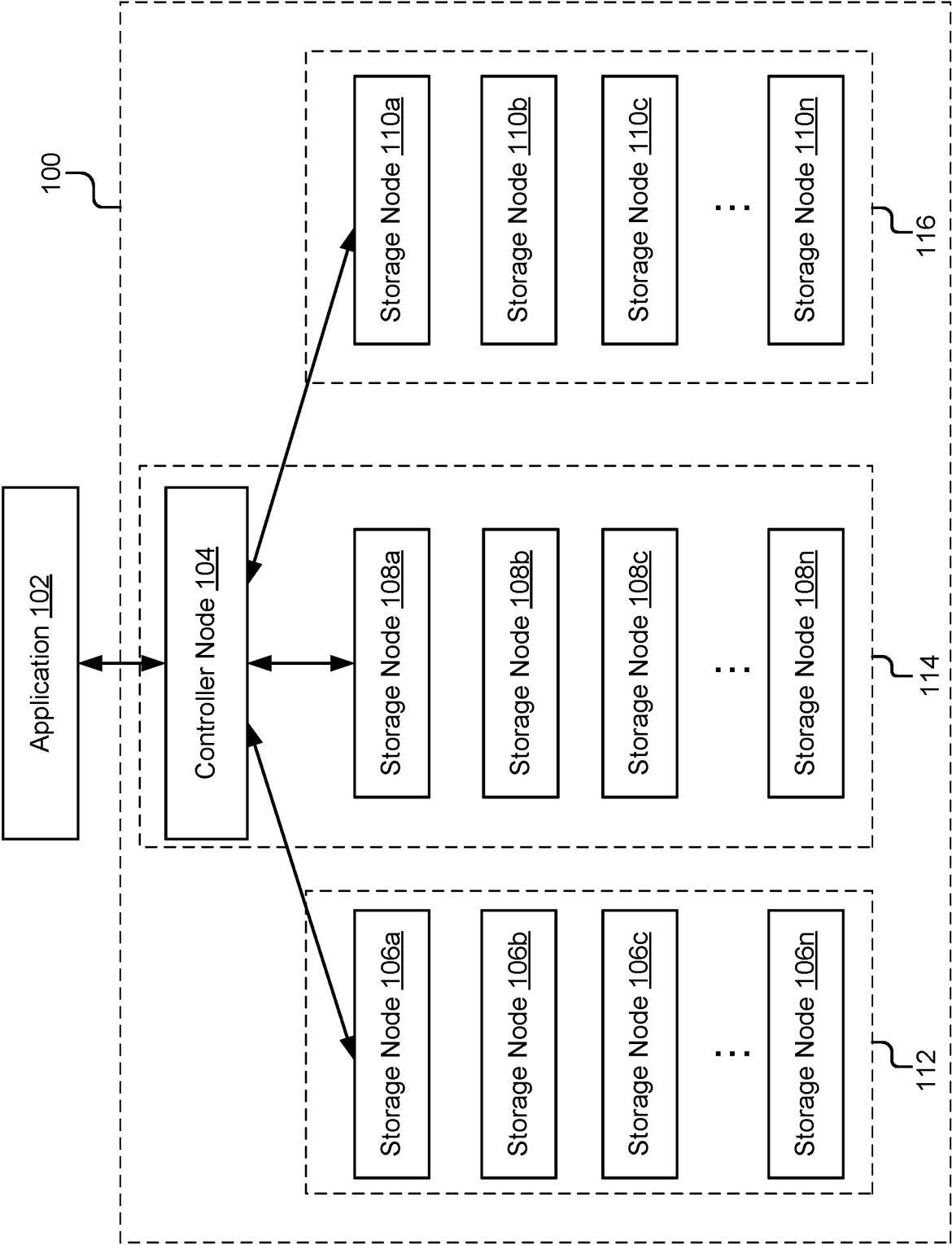


Figure 1

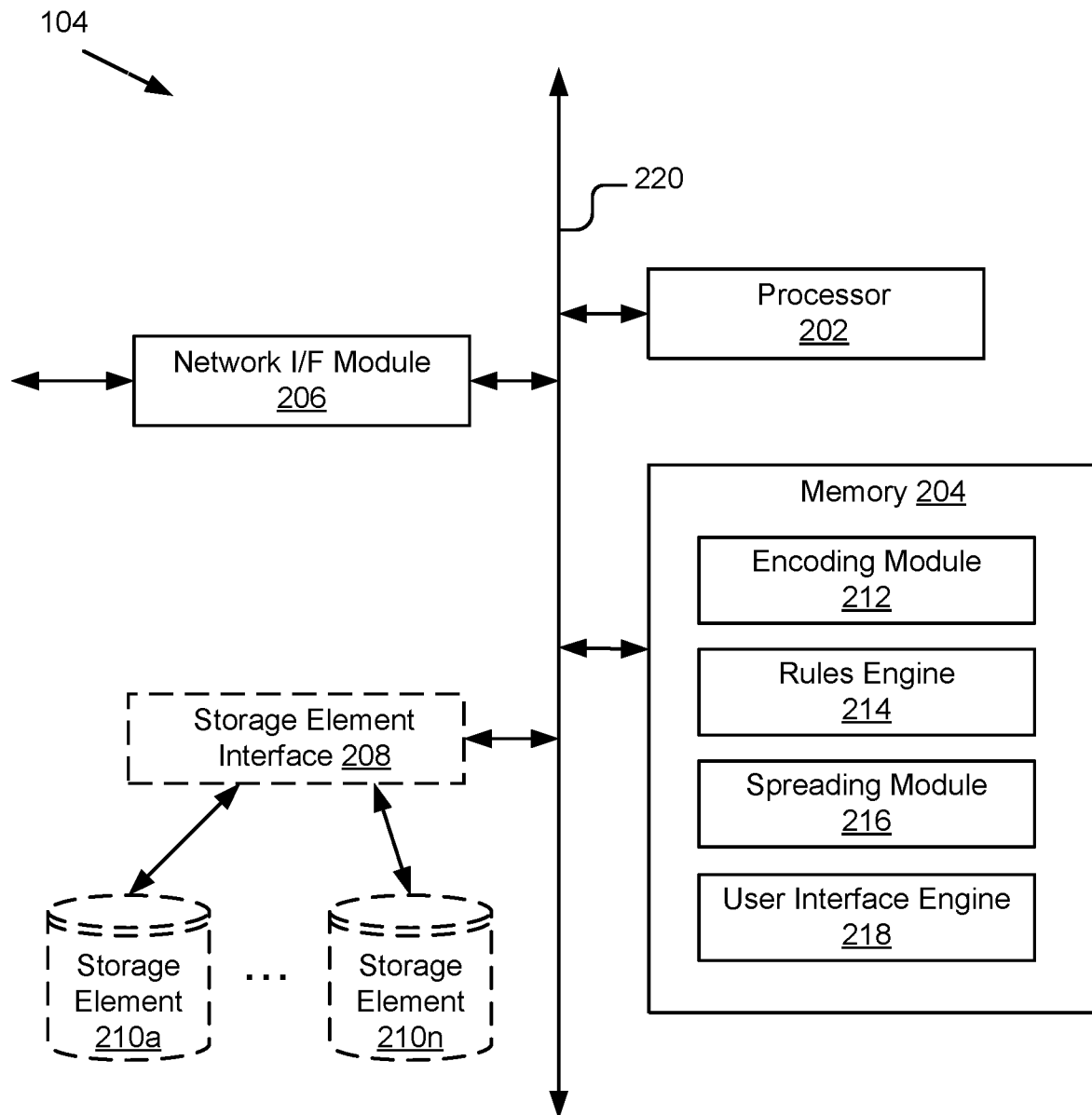


Figure 2

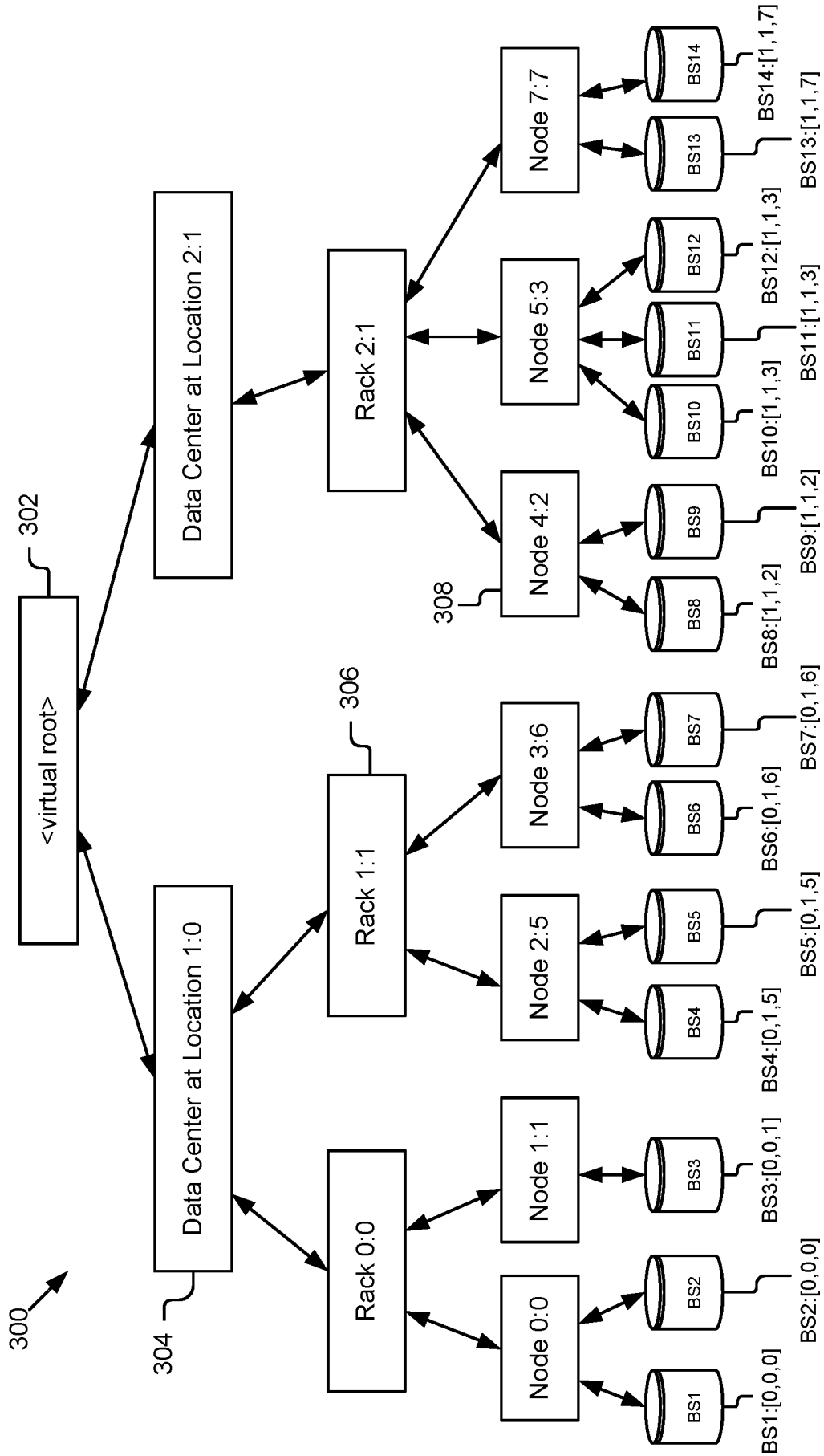


Figure 3

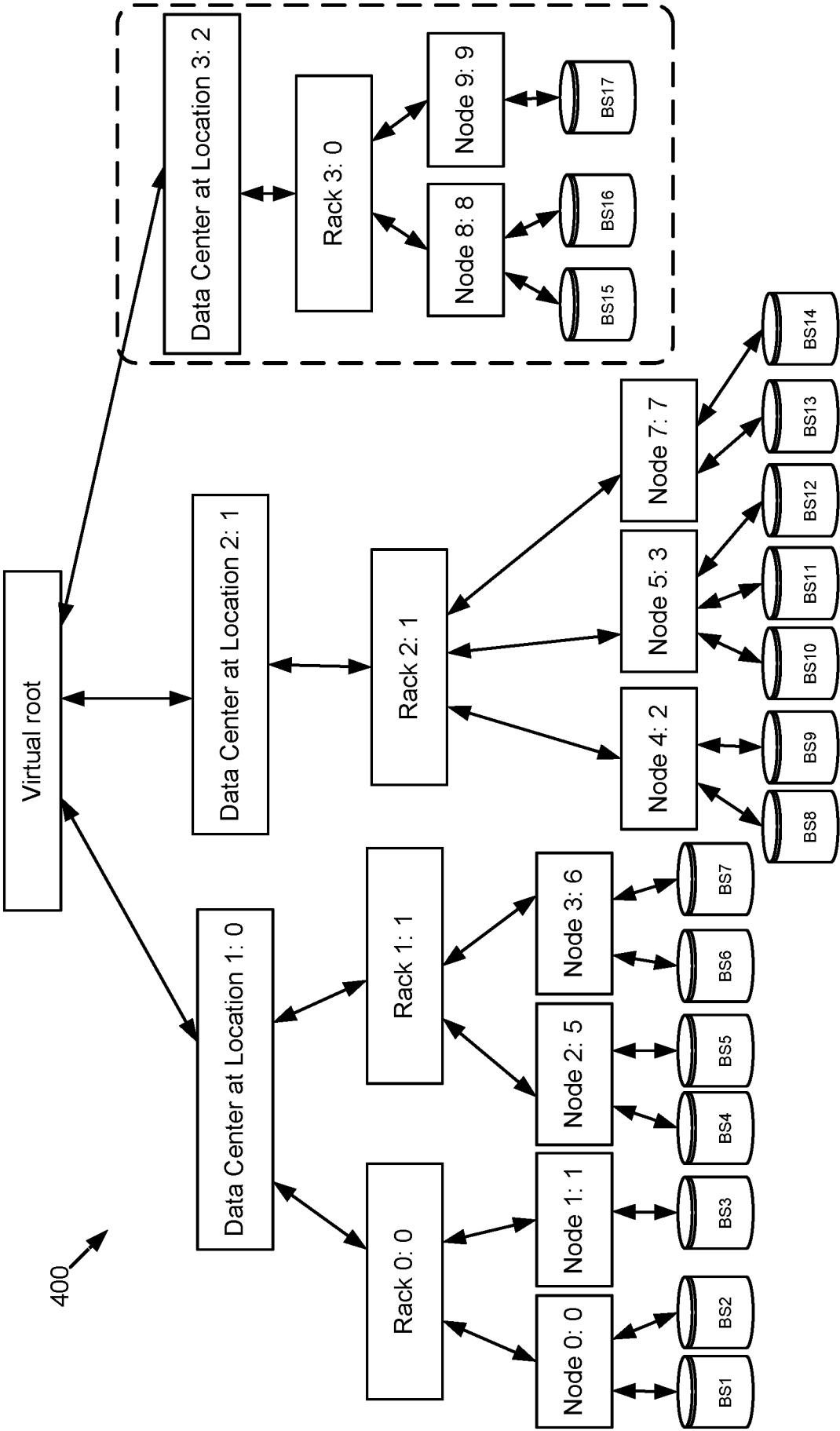


Figure 4

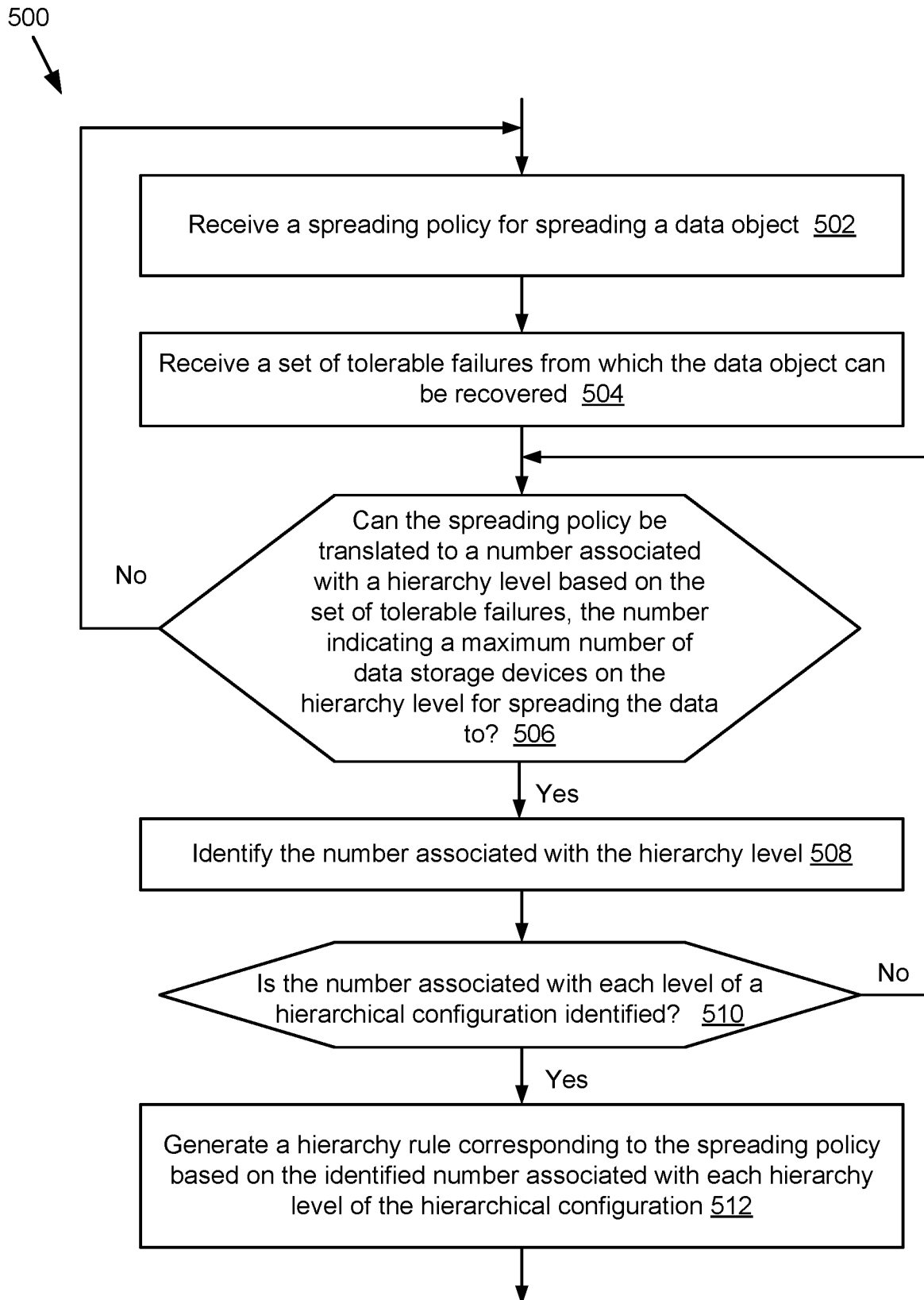


Figure 5

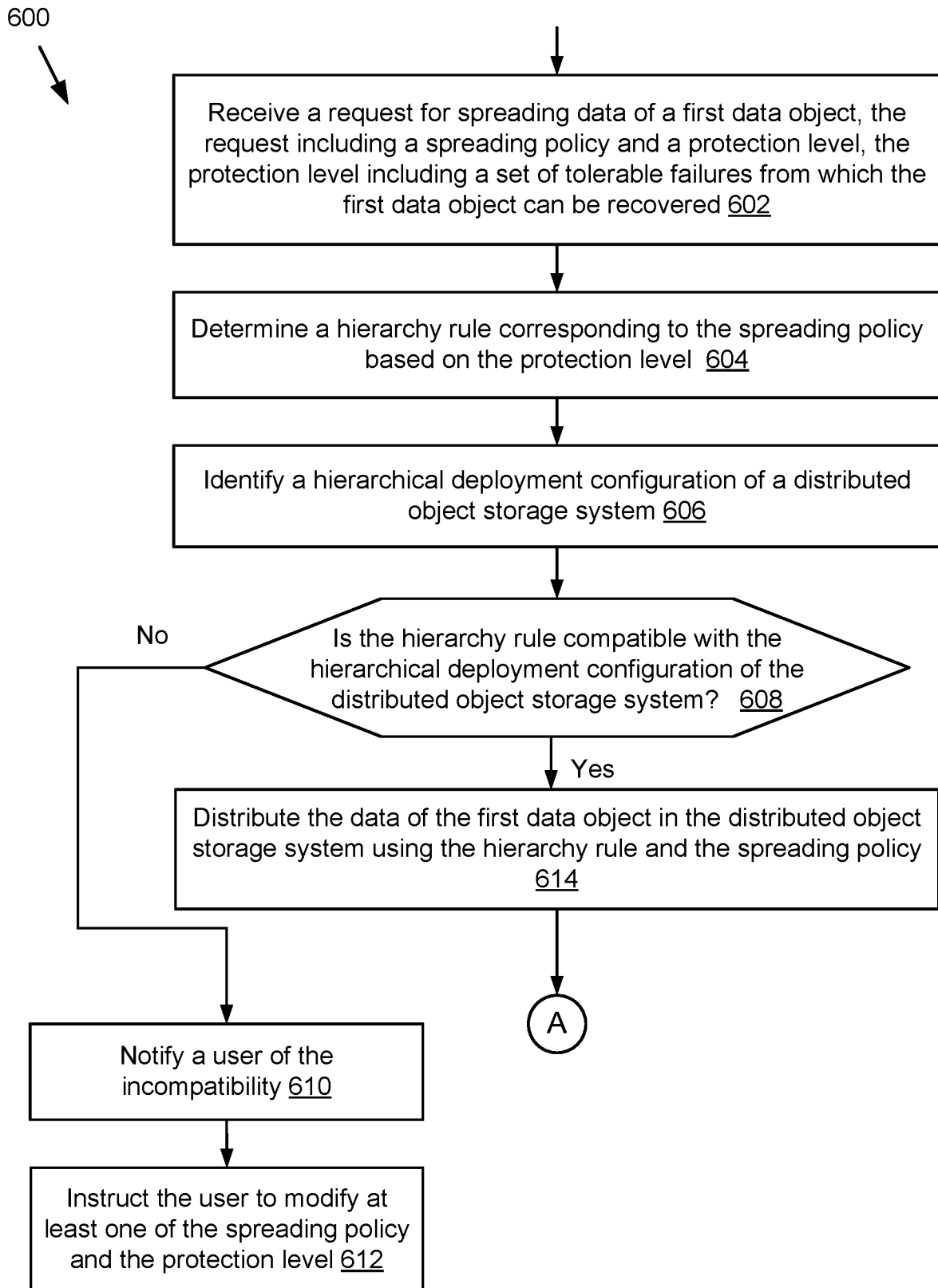


Figure 6A

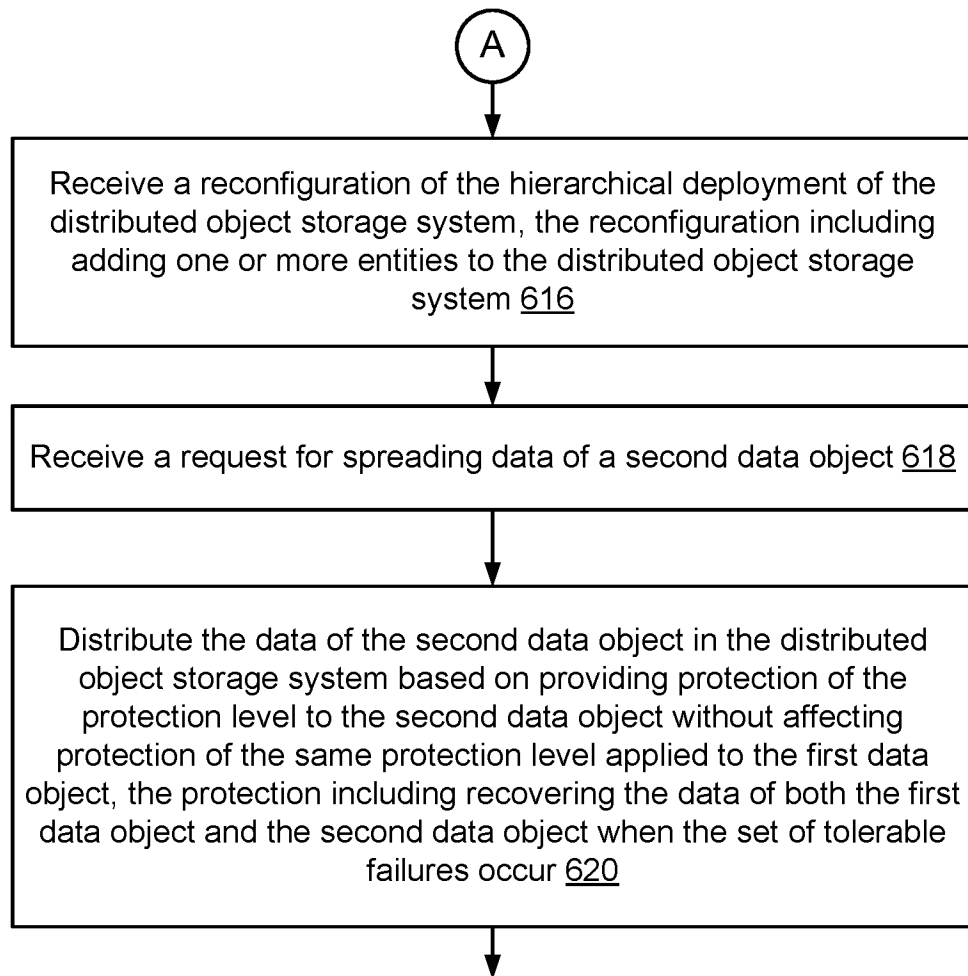


Figure 6B

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2017/049507

A. CLASSIFICATION OF SUBJECT MATTER

INV. H04L29/08 G06F11/10 G06F17/30 G06F3/06
ADD.

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)

H04L G06F

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, COMPENDEX, INSPEC, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X A A	US 2012/166726 A1 (DE SCHRIJVER FREDERIK [BE] ET AL) 28 June 2012 (2012-06-28) paragraph [0004] - paragraph [0005] paragraph [0024] paragraph [0043] - paragraph [0066] figures 1,6-8 -----	1,8,9, 14-16,20 2-6, 10-13, 17-19
A	US 2016/246676 A1 (BAKRE AJAY [IN] ET AL) 25 August 2016 (2016-08-25) abstract paragraph [0005] paragraph [0008] paragraph [0034] - paragraph [0049] figure 4 -----	1-20
A	US 2016/070617 A1 (ALGIE TEAGUE SCOTT [US] ET AL) 10 March 2016 (2016-03-10) abstract paragraph [0273] - paragraph [0289] -----	1-20



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

31 October 2017

Date of mailing of the international search report

08/11/2017

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Hes, Ronald

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2017/049507

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2012166726 A1	28-06-2012	EP 2469413 A1	27-06-2012
		US 2012166726 A1	28-06-2012

US 2016246676 A1	25-08-2016	US 2016246676 A1	25-08-2016
		US 2017139640 A1	18-05-2017

US 2016070617 A1	10-03-2016	US 2016070617 A1	10-03-2016
		US 2017116081 A1	27-04-2017
