Methods for use in a dispersed storage network (DSN) to enable rapid replication of data slices. Following dispersed storage error encoding of a data segment by a device of the DSN, slice naming information is generated for the resulting set of data slices. The slice naming information includes slice names and indicates a naming convention utilized for uniquely identifying replicated data slices. A set of write slice requests, including the slice naming information and the set of data slices, is then generated and broadcast or multicast to multiple sets of storage units of the DSN. Upon receipt of a write slice request, the recipient storage unit generates an updated slice name according to the naming convention and stores both the data slice and updated slice name in local memory. The data slice may be subsequently read from the storage unit by identifying the updated slice name in a read request.
dispersed storage error encode a data segment to produce a set of encoded data slices

generate slice naming information for the set of encoded data slices based on replicated storage of the set of encoded data slices in two or more storage sets

send the set of encoded data slices and the slice naming information to the two or more storage sets

store, by each storage unit of the two or more storage sets receiving an encoded data slice and the slice naming information, the received encoded data slice in a local memory

generate, by each storage unit, an updated slice name based on the slice naming information for association with the received encoded data slice

when retrieving the data segment, issue a read slice request to a storage unit of the two or more storage sets utilizing a slice name in accordance with the slice naming information

FIG. 11
120 determine, by a storage unit of a storage set, a naming convention for uniquely identifying replicated encoded data slices

122 receive a broadcast write slice request to store an encoded data slice of a set of encoded data slices

124 determine that the encoded data slice is to be stored in the storage unit in accordance with the naming convention

126 generate an updated slice name, in accordance with the naming convention, uniquely identifying the encoded data slice in relation to the encoded data slice as replicated in one or more additional storage units

128 store the encoded data slice and updated slice name in local memory of the storage unit

130 receive, from a computing device of the DSN, a read slice request identifying the encoding data slice

132 retrieve the encoded data slice from the local memory and send to requestor

FIG. 12
US 2017/0006099 A1

USING BROADCAST FOR PARALLELIZED AND RAPID SLICE REPLICATION IN A DISPERSED STORAGE NETWORK

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0001] Not applicable.

INCORPORATION-BY-REFERENCE OF MATERIAL SUBMITTED ON A COMPACT DISC

[0002] Not applicable.

BACKGROUND OF THE INVENTION

Technical Field of the Invention

[0003] This invention relates generally to computer networks, and more particularly to replicating error encoded data in a dispersed storage network.

[0004] Description of Related Art

[0005] Computing devices are known to communicate data, process data, and/or store data. Such computing devices range from wireless smart phones, laptops, tablets, personal computers (PC), work stations, and video game devices, to data centers that support millions of web searches, stock trades, or on-line purchases every day. In general, a computing device includes a central processing unit (CPU), a memory system, user input/output interfaces, peripheral device interfaces, and an interconnecting bus structure.

[0006] As is further known, a computer may effectively extend its CPU by using “cloud computing” to perform one or more computing functions (e.g., a service, an application, an algorithm, an arithmetic logic function, etc.) on behalf of the computer. Further, for large services, applications, and/or functions, cloud computing may be performed by multiple cloud computing resources in a distributed manner to improve the response time for completion of the service, application, and/or function. For example, Hadoop is an open source software framework that supports distributed applications enabling application execution by thousands of computers.

[0007] In addition to cloud computing, a computer may use “cloud storage” as part of its memory system. As is known, cloud storage enables a user, via its computer, to store files, applications, etc., on a remote storage system. The remote storage system may include a RAID (redundant array of independent disks) system and/or a dispersed storage system that uses an error correction scheme to encode data for storage.

[0008] In a RAID system, a RAID controller adds parity data to the original data before storing it across an array of disks. The parity data is calculated from the original data such that the failure of a single disk typically will not result in the loss of the original data. While RAID systems can address certain memory device failures, these systems may suffer from effectiveness, efficiency, and security issues. For instance, as more disks are added to the array, the probability of a disk failure rises, which may increase maintenance costs. When a disk fails, for example, it needs to be manually replaced before another disk(s) fails and the data stored in the RAID system is lost. To reduce the risk of data loss, data on a RAID device is often copied to one or more other RAID devices. While this may reduce the possibility of data loss, it also raises security issues since multiple copies of data may be available, thereby increasing the chances of unauthorized access. In addition, co-location of some RAID devices may result in a risk of a complete data loss in the event of a natural disaster, fire, power surge/outage, etc.

BRIEF DESCRIPTION OF THE DRAWING(S)

[0009] FIG. 1 is a schematic block diagram of an embodiment of a dispersed or distributed storage network (DSN) in accordance with the present invention;

[0010] FIG. 2 is a schematic block diagram of an embodiment of a computing core in accordance with the present invention;

[0011] FIG. 3 is a schematic block diagram of an example of dispersed storage error encoding of data in accordance with the present invention;

[0012] FIG. 4 is a schematic block diagram of a generic example of an error encoding function in accordance with the present invention;

[0013] FIG. 5 is a schematic block diagram of a specific example of an error encoding function in accordance with the present invention;

[0014] FIG. 6 is a schematic block diagram of an example of slice naming information for an encoded data slice (EDS) in accordance with the present invention;

[0015] FIG. 7 is a schematic block diagram of an example of dispersed storage error encoding of data in accordance with the present invention;

[0016] FIG. 8 is a schematic block diagram of a generic example of an error decoding function in accordance with the present invention;

[0017] FIGS. 9 and 10 are schematic block diagrams of an embodiment of a DSN in accordance with the present invention;

[0018] FIG. 11 is a logic diagram of an example of encoded data slice replication in accordance with the present invention; and

[0019] FIG. 12 is a logic diagram of a further example of encoded data slice replication in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0020] FIG. 1 is a schematic block diagram of an embodiment of a dispersed, or distributed, storage network (DSN) 10 that includes a plurality of computing devices 12 -16, a managing unit 18, an integrity processing unit 20, and a DSN memory 22. The components of the DSN 10 are coupled to a network 24, which may include one or more wireless and/or wire line communication systems; one or more non-public intranet systems and/or public internet systems; and/or one or more local area networks (LAN) and/or wide area networks (WAN).

[0021] The DSN memory 22 includes a plurality of storage units 36 that may be located at geographically different sites (e.g., one in Chicago, one in Milwaukee, etc.), at a common site, or a combination thereof. For example, if the DSN memory 22 includes eight storage units 36, each storage unit is located at a different site. As another example, if the DSN memory 22 includes eight storage units 36, all eight storage units are located at the same site. As yet
another example, if the DSN memory 22 includes eight storage units 36, a first pair of storage units are at a first common site, a second pair of storage units are at a second common site, a third pair of storage units are at a third common site, and a fourth pair of storage units are at a fourth common site. Note that a DSN memory 22 may include more or less than eight storage units 36. Further note that each storage unit 36 includes a computing core (as shown in FIG. 2, or components thereof) and a plurality of memory devices for storing dispersed error encoded data.

Each of the computing devices 12-16, the managing unit 18, and the integrity processing unit 20 include a computing core 26 which includes network interfaces 30-33. Computing devices 12-16 may each be a portable computing device and/or a fixed computing device. A portable computing device may be a social networking device, a gaming device, a cell phone, a smart phone, a digital assistant, a digital music player, a digital video player, a laptop computer, a handheld computer, a tablet, a video game controller, and/or any other portable device that includes a computing core. A fixed computing device may be a computer (PC), a computer server, a cable set-top box, a satellite receiver, a television set, a printer, a fax machine, home entertainment equipment, a video game console, and/or any type of home or office computing equipment. Note that each of the managing unit 18 and the integrity processing unit 20 may be separate computing devices, may be a common computing device, and/or may be integrated into one or more of the computing devices 12-16 and/or into one or more of the storage units 36.

Each interface 30, 32, and 33 includes software and hardware to support one or more communication links via the network 24 indirectly and/or directly. For example, interface 30 supports a communication link (e.g., wired, wireless, direct, via a LAN, via the network 24, etc.) between computing devices 14 and 16. As another example, interface 32 supports communication links (e.g., a wired connection, a wireless connection, a LAN connection, and/or any other type of connection) to/from the network 24 between computing devices 12 and 16 and the DSN memory 22. As yet another example, interface 33 supports a communication link for each of the managing unit 18 and the integrity processing unit 20 to the network 24.

Computing devices 12 and 16 include a dispersed storage (DS) client module 34, which enables the computing device to dispersed storage error encode and decode data (e.g., data object 40) as subsequently described with reference to one or more of FIGS. 3-8. In this example embodiment, computing device 16 functions as a dispersed storage processing agent for computing device 14. In this role, computing device 16 dispersed storage error encodes and decodes data on behalf of computing device 14. With the use of dispersed storage error encoding and decoding, the DSN 10 is tolerant of a significant number of storage unit failures (the number of failures is based on parameters of the dispersed storage error encoding function) without loss of data and without the need for a redundant or backup copies of the data. Further, the DSN 10 stores data for an indefinite period of time without data loss and in a secure manner (e.g., the system is very resistant to unauthorized attempts at accessing the data).

In operation, the managing unit 18 performs DS management services. For example, the managing unit 18 establishes distributed data storage parameters (e.g., vault creation, distributed storage parameters, security parameters, billing information, user profile information, etc.) for computing devices 12-14 individually or as part of a group of user devices. As a specific example, the managing unit 18 coordinates creation of a vault (e.g., a virtual memory block associated with a portion of an overall namespace of the DSN) within the DSN memory 22 for a user device, a group of devices, or for public access and establishes per vault dispersed storage (DS) error encoding parameters for a vault. The managing unit 18 facilitates storage of DS error encoding parameters for each vault by updating registry information of the DSN 10, where the registry information may be stored in the DSN memory 22, a computing device 12-16, the managing unit 18, and/or the integrity processing unit 20.

The managing unit 18 creates and stores user profile information (e.g., an access control list (ACL)) in local memory and/or within memory of the DSN memory 22. The user profile information includes authentication information, permissions, and/or the security parameters. The security parameters may include encryption/decryption scheme, one or more encryption keys, key generation scheme, and/or data encoding/decoding scheme.

The managing unit 18 creates billing information for a particular user, a user group, a vault access, public vault access, etc. For instance, the managing unit 18 tracks the number of times a user accesses a non-public vault and/or public vaults, which can be used to generate per-access billing information. In another instance, the managing unit 18 tracks the amount of data stored and/or retrieved by a user device and/or a user group, which can be used to generate per-data-amount billing information.

As another example, the managing unit 18 performs network operations, network administration, and/or network maintenance. Network operations includes authenticating user data allocation requests (e.g., read and/or write requests), managing creation of vaults, establishing authentication credentials for user devices, adding/deleting components (e.g., user devices, storage units, and/or computing devices with a DS client module 34) to/from the DSN 10, and/or establishing authentication credentials for the storage units 36. Network administration includes monitoring devices and/or units for failures, maintaining vault information, determining device and/or unit activation status, determining device and/or unit loading, and/or determining any other system level operation that affects the performance level of the DSN 10. Network maintenance includes facilitating replacing, upgrading, repairing, and/or expanding a device and/or unit of the DSN 10.

To support data storage integrity verification within the DSN 10, the integrity processing unit 20 (and/or other devices in the DSN 10) may perform rebuilding of ‘bad’ or missing encoded data slices. At a high level, the integrity processing unit 20 performs rebuilding by periodically attempting to retrieve/list encoded data slices, and/or slice names of the encoded data slices, from the DSN memory 22. Retrieved encoded slices are checked for errors due to data corruption, outdated versioning, etc. If a slice includes an error, it is flagged as a ‘bad’ or ‘corrupt’ slice. Encoded data slices that are not received and/or not listed may be flagged as missing slices. Bad and/or missing slices may be subsequently rebuilt using other retrieved encoded data slices that are deemed to be good slices in order to produce rebuilt slices. A multi-stage decoding process may be employed in
certain circumstances to recover data even when the number of valid encoded data slices of a set of encoded data slices is less than a relevant decode threshold number. The rebuilt slices may then be written to DSN memory 22. Note that the integrity processing unit 20 may be a separate unit as shown, included in DSN memory 22, included in the computing device 16, and/or distributed among the storage units 36.

[0030] FIG. 2 is a schematic block diagram of an embodiment of a computing core 26 that includes a processing module 50, a memory controller 52, main memory 54, a video graphics processing unit 55, an input/output (I/O) controller 56, a peripheral component interconnect (PCI) interface 58, an IO interface module 60, at least one IO device interface module 62, a read only memory (ROM) basic input output system (BIOS) 64, and one or more memory interface modules. The one or more memory interface module(s) includes one or more of a universal serial bus (USB) interface module 66, a host bus adapter (HBA) interface module 68, a network interface module 70, a flash interface module 72, a hard drive interface module 74, and a DSN interface module 76.

[0031] The DSN interface module 76 functions to mimic a conventional operating system (OS) file system interface (e.g., network file system (NFS), flash file system (FFS), disk file system (DFS), file transfer protocol (FTP), web-based distributed authoring and versioning (WebDAV), etc.) and/or a block memory interface (e.g., small computer system interface (SCSI), internet small computer system interface (iSCSI), etc.). The DSN interface module 76 and/or the network interface module 70 may function as one or more of the interface 30-33 of FIG. 1. Note that the IO device interface module 62 and/or the memory interface modules 66-76 may be collectively or individually referred to as IO ports.

[0032] FIG. 3 is a schematic block diagram of an example of dispersed storage error encoding of data. When a computing device 12 or 16 has data to store it disperses storage error encodes the data in accordance with a dispersed storage error encoding process based on dispersed storage error encoding parameters. The dispersed storage error encoding parameters include an encoding function (e.g., information dispersal algorithm, Reed-Solomon, Cauchy Reed-Solomon, systematic encoding, non-systematic encoding, on-line codes, etc.), a data segmenting protocol (e.g., data segment size, fixed, variable, etc.), and per data segment encoding values. The per data segment encoding values include a total, or pillar, width, number (T) of encoded data slices per encoding of a data segment (i.e., in a set of encoded data slices); a decode threshold number (D) of encoded data slices of a set of encoded data slices that are needed to recover the data segment; a read threshold number (R) of encoded data slices to indicate a number of encoded data slices per set to be read from storage for decoding of the data segment; and/or a write threshold number (W) to indicate a number of encoded data slices per set that must be accurately stored before the encoded data segment is deemed to have been properly stored. The dispersed storage error encoding parameters may further include slicing information (e.g., the number of encoded data slices that will be created for each data segment) and/or slice security information (e.g., per encoded data slice encryption, compression, integrity checksum, etc.).

[0033] In the present example, Cauchy Reed-Solomon has been selected as the encoding function (a generic example is shown in FIG. 4 and a specific example is shown in FIG. 5); the data segmenting protocol is to divide the data object into fixed sized data segments; and the per data segment encoding values include: a pillar width of 5, a decode threshold of 3, a read threshold of 4, and a write threshold of 4. In accordance with the data segmenting protocol, the computing device 12 or 16 divides the data (e.g., a file, (e.g., text, video, audio, etc.), a data object, or other data arrangement) into a plurality of fixed sized data segments (e.g., 1 through Y of a fixed size in range of Kilo-bytes to Tera-bytes or more). The number of data segments created is dependent of the size of the data and the data segmenting protocol.

[0034] The computing device 12 or 16 then disperses storage error encodes a data segment using the selected encoding function (e.g., Cauchy Reed-Solomon) to produce a set of encoded data slices. FIG. 4 illustrates a generic Cauchy Reed-Solomon encoding function, which includes an encoding matrix (EM), a data matrix (DM), and a coded matrix (CM). The size of the encoding matrix (EM) is dependent on the pillar width number (T) and the decode threshold number (D) of selected per data segment encoding values. To produce the data matrix (DM), the data segment is divided into a plurality of data blocks and the data blocks are arranged into D number of rows with Z data blocks per row. Note that Z is a function of the number of data blocks created from the data segment and the decode threshold number (D). The coded matrix is produced by matrix multiplying the data matrix by the encoding matrix.

[0035] FIG. 5 illustrates a specific example of Cauchy Reed-Solomon encoding with a pillar number (T) of five and decode threshold number of three. In this example, a first data segment is divided into twelve data blocks (D1-D12). The coded matrix includes five rows of coded data blocks, where the first row of X11-X14 corresponds to a first encoded data slice (EDS 1_1), the second row of X21-X24 corresponds to a second encoded data slice (EDS 2_1), the third row of X31-X34 corresponds to a third encoded data slice (EDS 3_1), the fourth row of X41-X44 corresponds to a fourth encoded data slice (EDS 4_1), and the fifth row of X51-X54 corresponds to a fifth encoded data slice (EDS 5_1). Note that the second number of the EDS designation corresponds to the data segment number. In the illustrated example, the value X11=mD3+nD7+oD11, and X54=nD4+nD8+oD12.

[0036] Returning to the discussion of FIG. 3, the computing device also creates a slice name (SN) for each encoded data slice (EDS) in the set of encoded data slices. A typical format for a slice name 80 is shown in FIG. 6. As shown, the slice name (SN) 80 includes a pillar number of the encoded data slice (e.g., 1 of 1-T), a data segment number (e.g., 1 of 1-Y), a vault identifier (ID), a data object identifier (ID), and may further include revision level information of the encoded data slices. The slice name functions as at least part of a DSN address for the encoded data slice for storage and retrieval from the DSN memory 22. As discussed more fully below in conjunction with FIGS. 9-12, a slice name 80 may be accompanied by an indication of naming convention 82 (collectively referred to herein as slice naming information) for use in replicated storage of encoded data slices in accordance with the present disclosure.

[0037] As a result of encoding, the computing device 12 or 16 produces a plurality of sets of encoded data slices, which are provided with their respective slice names to the storage.
units for storage. As shown, the first set of encoded data slices includes EDS 1 through EDS 5 and the first set of slice names includes SN 1 through SN 5 and the last set of encoded data slices includes EDS 1 through EDS 5 and the last set of slice names includes SN 1 through SN 5.

[0038] FIG. 7 is a schematic block diagram of an example of dispersed storage error decoding of a data object that was dispersed storage error encoded and stored in the example of FIG. 4. In this example, the computing device 12 or 16 retrieves from the storage units at least the decode threshold number of encoded data slices per data segment. As a specific example, the computing device retrieves a read threshold number of encoded data slices.

[0039] In order to recover a data segment from a decode threshold number of encoded data slices, the computing device uses a decoding function as shown in FIG. 8. As shown, the decoding function is essentially an inverse of the encoding function of FIG. 4. The coded matrix includes a decode threshold number of rows (e.g., three in this example) and the decoding matrix in an inversion of the encoding matrix that includes the corresponding rows of the coded matrix. For example, if the coded matrix includes rows 1, 2, and 4, the decoding matrix is reduced to rows 1, 2, and 4, and then inverted to produce the decoding matrix.

[0040] FIGS. 9 and 10 are schematic block diagrams of an embodiment of a dispersed storage network (DSN) in accordance with the present invention. The illustrated DSN includes one or more computing devices 1-D (such as computing device 16 of FIG. 1), the network 24 of FIG. 1, and a plurality of storage sets 1-S. Each computing device includes a DS client module 34. Each storage unit includes a set of storage units 1-n (e.g., storage set includes storage units 1-1-n). Each storage unit may be implemented utilizing the storage unit 36 of FIG. 1. Hereafter, each storage set may be interchangeably referred to as a set of storage units. The DSN of this embodiment functions to replicate encoded data slices as described more fully below.

[0041] Under certain circumstances, it may be desirable to enable rapid replication of a particular encoded data slice or set of encoded data slices in a DSN 10. Such circumstances may involve, by way of example and without limitation, a need for extreme storage reliability, legal compliance considerations, data objects subject to frequent read slice requests, etc. Novel methodologies utilizing broadcast/multicast transmissions and replicated slice naming conventions are described herein to enable rapid and efficient replication of an encoded data slice in two or more storage units via a single write slice request.

[0042] In an example of wireless broadcasting of a set of write slice requests, including slice naming information and a set of encoded data slices, any storage units with an appropriately tuned antenna may simultaneously receive the set of encoded data slices for storage in accordance with the slice naming information. In certain embodiments, each of a pillar width number of encoded data slices may be broadcast over a different frequency or group of frequencies, such that different storage units serving different pillars associated with an information dispersal algorithm may be configured to receive a different slice. Alternatively, different encoded data slices may be sent at varied times (or prefixed with a slice name/index) such that only certain storage units (e.g., storage units owning a relevant slice name range) process and store the slices.

[0043] In one approach, a “slice name template” or like naming convention is broadcast to the storage sets (e.g., as a separate broadcast, as a preamble to a broadcast set of write slice requests, or embedded within slice naming information of a broadcast set of write slice requests), and each storage unit generates an updated slice name based on that template, but one that is unique and falls into a slice name range that the storage unit serves/owns. In this manner, many uniquely named slices, perhaps existing in different vaults, may all be created from the same broadcast message. In certain embodiments, one or more naming conventions/addressing schemes for replicated storage of data slices may be pre-arranged between storage units and other devices of a DSN, with the slice naming information of a broadcast set of write slice requests (e.g., within slice naming information) providing an indication of the relevant naming convention. Alternatively, the indication of a naming convention may include omission of certain slice name information in accordance with a predetermined slice naming convention.

[0044] Referring more particularly to FIG. 9, in an example of operation of the replication of data in a DSN 10, the DSN client module 34 dispersed storage error encodes a data segment into a set of encoded data slices 1-n, where a data object is divided into a plurality of data segments that includes the data segment. Having generated the set of encoded data slices, the DSN client module 34 generates slice naming information for the set of encoded data slices. For example, the DSN client module 34 generates a set of slice names and indicates a naming convention for uniquely identifying replicated encoded data slices as replicated in two or more storage units. The naming convention may include, for example appending a prefix or suffix to a given slice name, where the prefix or suffix is based on a unique identifier for a storage unit. In the illustrated embodiment, for instance, storage set 1 refers to encoded data slice 1 as encoded data slice 1-1, storage set 2 refers to encoded data slice 1 as encoded data slice 2-1, storage set 3 refers to encoded data slice 1 as encoded data slice 3-1, etc.

[0045] Having generated the slice naming information, the DSN client module 34 sends, via the network 24, the set of encoded data slices and the slice naming information to two or more sets of storage units. For example, the DSN client module 34 sends, via a wireless broadcast connection of the network 24, a simultaneous broadcast of slices 1-n to each of the storage sets 1-S. As another example, the DSN client module 34 sends, via a wireline connection of the network 24, a substantially simultaneous multicast (e.g., internet protocol (IP) multicast transmissions) of slices 1-n to each of the storage sets 1-S.

[0046] Each storage unit receiving an encoded data slice and the slice naming information stores the received encoded data slice in a local memory. Each storage unit also generates an updated slice name, based on the slice naming information, for association with a received encoded data slice. For example, storage unit 2-1 modifies the slice name for encoded data slice 1 to produce a slice name of 2-1 for the encoded data slice 1. As another example, storage unit 3-1 modifies the slice name for encoded data slice 1 to produce a slice name of 3-1 for the encoded data slice 1.

[0047] As illustrated in FIG. 10, when retrieving the data segment from one of the storage sets, a DSN client module 34 may issue, via the network 24, a read slice request to a desired target storage unit for slice retrieval utilizing a slice name in accordance with the slice naming information. For
example, when retrieving the data segment from the storage set 3, the DS client module 34 of computing device 2 issues, via the network 24, a read slice request that includes a slice name of 3-1 to retrieve the encoded data slice 1. Other DS client modules 34 may issue substantially simultaneous read slice requests using the naming convention to retrieve slices from specific storage sets/storage units. Such an approach to recovery of the stored data may provide system-level performance improvements by providing substantially simultaneous recovery of one or more data segments in parallel across the plurality of storage sets utilizing different slice names for the replicated sets of encoded data slices.

[0048] In order to enhance security and prevent unauthorized storage and retrieval of encoded data slices, write slice requests, read slice requests, and other communications from the computing devices 1-D may be digitally signed or broadcast with an appropriate message authentication code (MAC) to prove that such communications originated from an authorized entity. Broadcast write/read slice requests may also be encrypted (e.g., authenticated encryption (AE) or authenticated encryption with associated data (AEAD)) using key information known only internally to a location housing relevant storage units, such that the broadcasts are not considered public and confidentiality is maintained for slice content.

[0049] FIG. 11 is a logic diagram of an example of encoded data slice replication in accordance with the present invention. In particular, a method is presented for use in conjunction with one or more functions and features described in conjunction with FIGS. 1-10. The method begins at step 100 where a processing module (e.g., of a distributed storage (DS) client module of a computing device 16) disperses storage error encodes a data segment of a data object to produce a set of encoded data slices. The method continues at step 102 where the processing module generates slice naming information for the set of encoded data slices based on replicated storage of the set of encoded data slices in two or more storage sets. For example, the processing module generates a set of slice names, indicates a naming convention for uniquely identifying replicated encoded data slices (e.g., appending a storage unit identifier), and generates an indication of which storage units are to replicate which encoded data slices of the set of encoded data slices.

[0050] The method continues at step 104 where the processing module sends the set of encoded data slices and the slice naming information to the two or more storage sets. For example, the processing module facilitates broadcast (e.g., via a wireless connection) of a set of write slice requests that includes the set of encoded data slices and the slice naming information. As another example, the processing module facilitates multicast transmission (e.g., in accordance with an IP multicast standard) of the set of write slice requests.

[0051] The method continues at step 106 where each storage unit of the two or more storage sets receiving an encoded data slice and a slice naming information stores the received encoded data slice in a local memory. Next, at step 108, each storage unit generates an updated slice name based on the slice naming information for association with a received encoded data slice. For example, each storage unit modifies a received slice name in accordance with the slice naming information (e.g., appends a unique identifier of the storage unit as a prefix/suffix to the slice name) to produce the updated slice name.

[0052] When retrieving the data segment, the method continues at step 110 where a processing module issues a read slice request to a target storage unit of the two or more storage sets utilizing a slice name (or other slice identification information) in accordance with the relevant naming convention. For example, the processing module may generate the slice name for the read slice request based on the identifier of the storage unit (e.g., generates slice name 3-1 when requesting slice 1 from storage unit 3-1).

[0053] FIG. 12 is a logic diagram of a further example of encoded data slice replication in accordance with the present invention. In particular, a method is presented for use in conjunction with one or more functions and features described in conjunction with FIGS. 1-10. The method begins at step 120 where a storage unit (e.g., a storage unit of a storage set such as illustrated in FIG. 9) determines a naming convention for uniquely identifying replicated encoded data slices. Upon receiving a write slice request, such as a broadcast or multicast write slice request, to store an encoded data slice of a set of encoded data slices as shown in step 122, the storage unit (step 124) determines that the encoded data slice is to be stored in the storage unit in accordance with the naming convention. Such a determination may be based on information (e.g., slice naming information) contained within the write slice request.

[0054] Next, in step 126, the storage unit generates an updated slice name, in accordance with the naming convention, that uniquely identifies the encoded data slice in relation to the name of the slice as it may be replicated in one or more additional storage units of the DSN. The encoded data slice and updated slice name are then stored in local memory of the storage unit in step 128. As previously described, updating the slice name may include appending storage unit identification information to the slice name, where the storage unit identification information is distinct from storage unit identification information utilized by one or more additional storage units of the DSN.

[0055] The method continues at step 130 where the storage unit receives a read slice request from a computing device of the DSN, the read slice request identifying the encoded data slice using the naming convention. In response, at step 132 the storage unit retrieves the encoded data slice from the local memory and sends it to the requesting device.

[0056] In various embodiments, the storage unit may apply integrity verification information to the encoded data slice before sending it to the requesting device. For example, when reading an encoded data slice from local memory, the storage unit may determine that the slice data does not match a corresponding stored integrity check value. This can result in an error being returned to the requestor instead of the slice. For example, a storage unit detecting a bad data slice may return a read slice response including the data slice and a flag indicating that the data slice is likely corrupt (e.g., the storage unit may not be able to determine whether the data within the slice is corrupted or the integrity check value is wrong). Upon receiving such an error indication, the requesting device may attempt to correct the encoded data slice, ignore it if a decode threshold number of encoded data slices is received, or send a revised read slice request to another storage unit that is storing a replicated copy of the encoded data slice.

[0057] The methods described above in conjunction with the computing device and the storage units can alternatively
be performed by other modules of the dispersed storage network or by other devices. For example, any combination of a first module, a second module, a third module, a fourth module, etc. of the computing device and the storage units may perform the method described above. In addition, at least one memory section (e.g., a first memory section, a second memory section, a third memory section, a fourth memory section, a fifth memory section, a sixth memory section, etc. of a non-transitory computer readable storage medium) that stores operational instructions can, when executed by one or more processing modules of one or more computing devices and/or by the storage units of the dispersed storage network (DSN), cause the one or more computing devices and/or the storage units to perform any or all of the method steps described above.

[0058] As may be used herein, the terms “substantially” and “approximately” provide an industry-accepted tolerance for its corresponding term and/or relativity between items. Such an industry-accepted tolerance ranges from less than one percent to fifty percent. Such relativity between items ranges from a difference of a few percent to magnitude differences. As may also be used herein, the term(s) “configured to”, “operably coupled to”, “coupled to”, and/or “coupling” includes direct coupling between items and/or indirect coupling between items via an intervening item (e.g., an item includes, but is not limited to, a component, an element, a circuit, and/or a module) where, for example, an indirect coupling, the intervening item does not modify the information of a signal but may adjust its current level, voltage level, and/or power level. As may further be used herein, inferred coupling (i.e., where one element is coupled to another element by inference) includes direct and indirect coupling between two items in the same manner as “coupled to”. As may even further be used herein, the term “configured to”, “operable to”, “coupled to”, or “operably coupled to” indicates that an item includes one or more of power connections, input(s), output(s), etc. to perform, when activated, one or more of its corresponding functions and may further include inferred coupling to one or more other items. As may still further be used herein, the term “associated with”, includes direct and/or indirect coupling of separate items and/or one item being embedded within another item.

[0059] As may be used herein, the term “comparably”, indicates that a comparison between two or more items, signals, etc., provides a desired relationship. For example, when the desired relationship is that signal 1 has a greater magnitude than signal 2, a favorable comparison may be achieved when the magnitude of signal 1 is greater than that of signal 2 or when the magnitude of signal 2 is less than that of signal 1. As may be used herein, the term “comparably unfavorably”, indicates that a comparison between two or more items, signals, etc., fails to provide the desired relationship.

[0060] As may also be used herein, the terms “processing module”, “processing circuit”, “processor”, and/or “processing unit” may be a single processing device or a plurality of processing devices. Such a processing device may be a microprocessor, microcontroller, digital signal processor, microcomputer, central processing unit, field programmable gate array, programmable logic device, state machine, logic circuitry, analog circuitry, digital circuitry, and/or any device that manipulates signals (analog and/or digital) based on hard coding of the circuitry and/or operational instructions. The processing module, module, processing circuit, and/or processing unit may be, or further include, memory and/or an integrated memory element, which may be a single memory device, a plurality of memory devices, and/or embedded circuitry of another processing module, module, processing circuit, and/or processing unit. Such a memory device may be a read-only memory, random access memory, volatile memory, non-volatile memory, static memory, dynamic memory, flash memory, cache memory, and/or any device that stores digital information. Note that if the processing module, module, processing circuit, and/or processing unit includes more than one processing device, the processing devices may be centrally located (e.g., directly coupled together via a wired and/or wireless bus structure) or may be distributedly located (e.g., cloud computing via indirect coupling via a local area network and/or a wide area network). Further note that if the processing module, module, processing circuit, and/or processing unit implements one or more of its functions via a state machine, analog circuitry, digital circuitry, and/or logic circuitry, the memory and/or memory element storing the corresponding operational instructions may be embedded within, or external to, the circuitry comprising the state machine, analog circuitry, digital circuitry, and/or logic circuitry. Still further note that, the memory element may store, and the processing module, module, processing circuit, and/or processing unit executes, hard coded and/or operational instructions corresponding to at least some of the steps and/or functions illustrated in one or more of the Figures. Such a memory device or memory element can be included in an article of manufacture.

[0061] One or more embodiments have been described above with the aid of method steps illustrating the performance of specified functions and relationships thereof. The boundaries and sequence of these functional building blocks and method steps have been arbitrarily defined herein for convenience of description. Alternate boundaries and sequences can be defined so long as the specified functions and relationships are appropriately performed. Any such alternate boundaries or sequences are thus within the scope and spirit of the claims. Further, the boundaries of these functional building blocks have been arbitrarily defined for convenience of description. Alternate boundaries could be defined as long as the certain significant functions are appropriately performed. Similarly, flow diagram blocks may also have been arbitrarily defined herein to illustrate certain significant functionality.

[0062] To the extent used, the flow diagram block boundaries and sequence could have been defined otherwise and still perform the certain significant functionality. Such alternate definitions of both functional building blocks and flow diagram blocks and sequences are thus within the scope and spirit of the claims. One of average skill in the art will also recognize that the functional building blocks, and other illustrative blocks, modules and components herein, can be implemented as illustrated or by discrete components, application specific integrated circuits, processors executing appropriate software and the like or any combination thereof.

[0063] In addition, a flow diagram may include a “start” and/or “continue” indication. The “start” and “continue” indications reflect that the steps presented can optionally be incorporated in or otherwise used in conjunction with other routines. In this context, “start” indicates the beginning of the first step presented and may be preceded by other activities not specifically shown. Further, the “continue”
indication reflects that the steps presented may be performed multiple times and/or may be succeeded by other activities not specifically shown. Further, while a flow diagram indicates a particular ordering of steps, other orderings are likewise possible provided that the principles of causality are maintained.

[0064] The one or more embodiments are used herein to illustrate one or more aspects, one or more features, one or more concepts, and/or one or more examples. A physical embodiment of an apparatus, an article of manufacture, a machine, and/or of a process may include one or more of the aspects, features, concepts, examples, etc. described with reference to one or more of the embodiments discussed herein. Further, from Figure to Figure, the embodiments may incorporate the same or similarly named functions, steps, modules, etc. that may use the same or different reference numbers and, as such, the functions, steps, modules, etc. may be the same or similar functions, steps, modules, etc. or different ones.

[0065] Unless specifically stated to the contrary, signals to, from, and/or between elements in a figure of any of the figures presented herein may be analog or digital, continuous time or discrete time, and single-ended or differential. For instance, if a signal path is shown as a single-ended path, it also represents a differential signal path. Similarly, if a signal path is shown as a differential path, it also represents a single-ended signal path. While one or more particular architectures are described herein, other architectures can likewise be implemented that use one or more data buses not expressly shown, direct connectivity between elements, and/or indirect coupling between other elements as recognized by one of average skill in the art.

[0066] The term “module” is used in the description of one or more of the embodiments. A module implements one or more functions via a device such as a processor or other processing device or other hardware that may include or operate in association with a memory that stores operational instructions. A module may operate independently and/or in conjunction with software and/or firmware. As also used herein, a module may contain one or more sub-modules, each of which may be one or more modules.

[0067] As may further be used herein, a computer readable memory includes one or more memory elements. A memory element may be a separate memory device, multiple memory devices, or a set of memory locations within a memory device. Such a memory device may be a read-only memory, random access memory, volatile memory, non-volatile memory, static memory, dynamic memory, flash memory, cache memory, and/or any device that stores digital information. The memory device may be in a form a solid state memory, a hard drive memory, cloud memory, thumb drive, server memory, computing device memory, and/or other physical medium for storing digital information. A computer readable memory/storage medium, as used herein, is not to be construed as being transitory signals per se, such as radio waves or other freely propagating electromagnetic waves, electromagnetic waves propagating through a waveguide or other transmission media (e.g., light pulses passing through a fiber-optic cable), or electrical signals transmitted through a wire.

[0068] While particular combinations of various functions and features of the one or more embodiments have been expressly described herein, other combinations of these features and functions are likewise possible. The present disclosure is not limited by the particular examples disclosed herein and expressly incorporates these other combinations.

What is claimed is:

1. A method for execution by one or more processing modules of one or more computing devices of a dispersed storage network (DSN), the method comprises:
   - dispersed storage error encoding a data segment to produce a set of encoded data slices for replicated storage in sets of storage units of the DSN, where a data object is segmented into a plurality of data segments that include the data segment;
   - generating slice naming information for encoded data slices of the set of encoded data slices, the slice naming information indicating a naming convention utilized by the sets of storage units of the DSN for uniquely identifying encoded data slices as replicated in two or more sets of storage units;
   - generating a set of write slice requests, the set of write slice requests including the slice naming information and the set of encoded data slices; and
   - sending the set of write slice requests to the two or more sets of storage units of the DSN, wherein an individual write slice request of the set of write slice requests functions to initiate replicated storage of an associated encoded data slice in the two or more sets of storage units of the DSN in accordance with the naming convention.

2. The method of claim 1, wherein sending the set of write slice requests to the two or more sets of storage units of the DSN includes broadcasting the set of write slice requests over at least one wireless communication link of the DSN.

3. The method of claim 1, wherein sending the set of write slice requests to the two or more sets of storage units of the DSN includes multicasting, in a substantially simultaneous manner, the set of write slice requests to the two or more sets of storage units.

4. The method of claim 3, the multicasting including an internet protocol multicast transmission occurring, at least in part, over a wired communication link of the DSN.

5. The method of claim 1 further comprises:
   - generating a read slice request for an encoded data slice of the set of encoded data slices, the read slice request identifying, in accordance with the naming convention, a target storage unit of the plurality of sets of storage units of the DSN; and
   - sending the read slice request for processing by the target storage unit.

6. The method of claim 1, wherein sending the set of write slice requests to the two or more sets of storage units of the DSN includes generating and sending a digital signature or message authentication code for individual ones of the set of write slice requests.

7. The method of claim 1, prior to sending the set of write slice requests to the two or more sets of storage units of the DSN, encrypting the set of write slice requests for authentication by the two or more sets of storage units.

8. The method of claim 1, the slice naming information further indicating particular sets of storage units for replicated storage of particular encoded data slices of the set of encoded data slices.

9. The method of claim 1, the method repeated for additional data segments of the plurality of data segments.
10. A method for execution by one or more storage units of a dispersed storage network (DSN), the method comprises:

determining a naming convention for uniquely identifying replicated encoded data slices;

receiving, by a storage unit of the DSN, a write slice request to store an encoded data slice of a set of encoded data slices of a data segment, where a data object is segmented into a plurality of data segments that include the data segment, the write slice request including a slice name and the encoded data slice; the write slice request having been broadcast for reception by one or more additional storage units of the DSN;

determining that the encoded data slice is to be stored in the storage unit in accordance with the naming convention;

generating an updated slice name in accordance with the naming convention, the updated slice name uniquely identifying the encoded data slice in relation to the encoded data slice as replicated in the one or more additional storage units of the DSN receiving the write slice request; and

storing the encoded data slice and updated slice name in local memory of the storage unit.

11. The method of claim 10, wherein determining that the encoded data slice is to be stored in the storage unit in accordance with the naming convention is based on information contained within the write slice request.

12. The method of claim 10, wherein generating an updated slice name includes appending storage unit identification information to the slice name, the storage unit identification information distinct from storage unit identification information utilized by the one or more additional storage units of the DSN.

13. The method of claim 10 further comprises:

receiving, from a computing device of the DSN, a read slice request identifying the encoded data slice in accordance with the naming convention;

retrieving the encoded data slice from the local memory of the storage unit; and

sending the encoded data slice to the computing device.

14. The method of claim 13, wherein retrieving the encoded data slice includes utilizing stored integrity information corresponding to the encoded data slice to determine if the stored data of the encoded data slice is valid.

15. A computing device of a group of computing devices of a dispersed storage network (DSN), the computing device comprises:

an interface;

a local memory; and

a processing module operably coupled to the interface and the local memory, wherein the processing module operates to:

dispersed storage error encode a data segment to produce a set of encoded data slices for replicated storage in sets of storage units of the DSN, where a data object is segmented into a plurality of data segments that include the data segment;

generate slice naming information for encoded data slices of the set of encoded data slices, the slice naming information indicating a naming convention utilized by the sets of storage units of the DSN for uniquely identifying encoded data slices as replicated in two or more sets of storage units;

generate a set of write slice requests, the set of write slice requests including the slice naming information and the set of encoded data slices; and

send, via the interface, the set of write slice requests for reception by the two or more sets of storage units of the DSN, wherein an individual write slice request of the set of write slice requests functions to initiate replicated storage of an associated encoded data slice in the two or more sets of storage units of the DSN in accordance with the naming convention.

16. The computing device of claim 15, wherein sending, via the interface, the set of write slice requests to the two or more sets of storage units of the DSN includes wirelessly broadcasting the set of write slice requests over at least one wireless communication link of the DSN.

17. The computing device of claim 15, wherein sending, via the interface, the set of write slice requests to the two or more sets of storage units of the DSN includes multicasting, in a substantially simultaneous manner, the set of write slice requests to the two or more sets of storage units.

18. The computing device of claim 15, wherein the processing module further functions to:

generate a read slice request for an encoded data slice of the set of encoded data slices, the read slice request identifying, in accordance with the naming convention, a set of storage units of the plurality of sets of storage units of the DSN; and

send, via the interface, the read slice request for processing by the identified set of storage units.

19. The computing device of claim 15, wherein generating the set of write slice requests includes generating a digital signature or message authentication code for individual ones of the set of write slice requests, and appending the digital signature or message authentication code thereto.

20. The computing device of claim 15, wherein generating the set of write slice requests includes encrypting the set of write slice requests for authentication by the two or more sets of storage units.

* * * * *