White Paper

EMC ISILON ONEFS: A TECHNICAL OVERVIEW

Abstract

This white paper provides an in-depth look at the major components of the EMC Isilon OneFS operating and file system. It details the software, hardware, distributed architecture and various data protection mechanisms that make OneFS a very flexible, easy to manage and a highly scalable enterprise storage solution both in terms of capacity as well as performance.

November 2013

Part Number H10719.3

Table of Contents

Introduction

In 2000, seeing the challenges with traditional storage architectures, and the pace at which file-based data was increasing, the founders of Isilon Systems began work on a revolutionary new storage architecture—the OneFS® Operating System. The fundamental difference of EMC® Isilon® storage is that it uses intelligent software to scale data across vast quantities of commodity hardware, enabling explosive growth in performance and capacity. The three layers of the traditional storage model—file system, volume manager, and data protection—have evolved over time to suit the needs of small-scale storage architectures, but introduce significant complexity and are not well adapted to petabyte-scale systems. OneFS replaces all of these, providing a unifying clustered file system with built-in scalable data protection, and obviating the need for volume management. OneFS is a fundamental building block for scale-out infrastructures, allowing for massive scale and tremendous efficiency.

Crucially, OneFS is designed to scale not just in terms of machines, but also in human terms—allowing large-scale systems to be managed with a fraction of the personnel required for traditional storage systems. OneFS eliminates complexity and incorporates self-healing and self-managing functionality that dramatically reduces the burden of storage management. OneFS also incorporates parallelism at a very deep-level of the OS, such that virtually every key system service is distributed across multiple units of hardware. This allows OneFS to scale in virtually every dimension as the infrastructure is expanded, ensuring that what works today, will continue to work as the dataset grows.

OneFS is a fully symmetric file system with no single point of failure — taking advantage of clustering not just to scale performance and capacity, but also to allow for any-to-any failover and multiple levels of redundancy that go far beyond the capabilities of RAID. The trend for disk subsystems has been slowly-increasing performance while rapidly-increasing storage densities. OneFS responds to this reality by scaling the amount of redundancy as well as the speed of failure repair. This allows OneFS to grow to multi-petabyte scale while providing greater reliability than small, traditional storage systems.

Isilon scale-out NAS hardware provides the appliance on which OneFS executes. Hardware components are best-of-breed, but commodity-based — ensuring that Isilon hardware benefits from commodity hardware's ever-improving cost and efficiency curves. OneFS allows hardware to be incorporated or removed from the cluster at will and at any time, abstracting the data and applications away from the hardware. Data is given infinite longevity, protected from the vicissitudes of evolving hardware generations. The cost and pain of data migrations and hardware refreshes are eliminated.

OneFS is ideally suited for file-based and unstructured "Big Data" applications in enterprise environments including large-scale home directories, file shares, archives, virtualization and business analytics. As such, OneFS is widely used in many data-intensive industries today, including energy, financial services, Internet and hosting services, business intelligence, engineering, manufacturing, media & entertainment, bioinformatics, scientific research and other high performance computing environments.

EMC Isilon OneFS overview

OneFS combines the three layers of traditional storage architectures—file system, volume manager, and data protection—into one unified software layer, creating a single intelligent distributed file system that runs on an Isilon storage cluster.

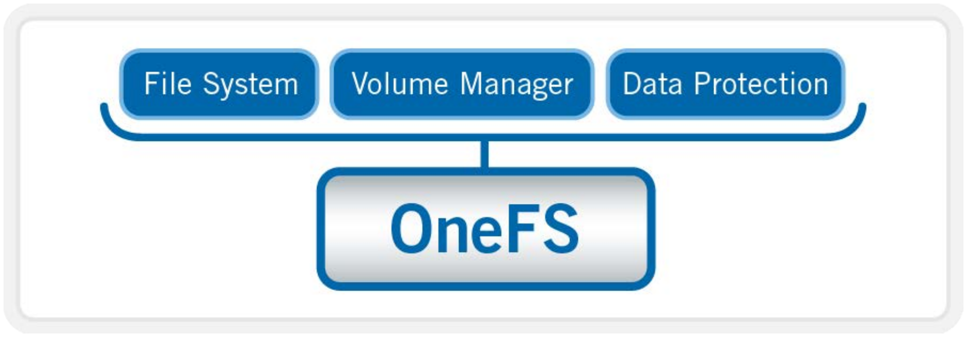

Figure 1: OneFS Combines File System, Volume Manager and Data Protection into One Single Intelligent, Distributed System.

This is the core innovation that directly enables enterprises to successfully utilize the scale-out NAS in their environments today. It adheres to the key principles of scale-out; intelligent software, commodity hardware and distributed architecture. OneFS is not only the operating system but also the underlying file system that drives and stores data in the Isilon scale-out NAS cluster.

Isilon nodes

OneFS works exclusively with the Isilon scale-out NAS nodes, referred to as a “cluster”. A single Isilon cluster consists of multiple “nodes”, which are constructed as rack-mountable enterprise appliances containing: memory, CPU, networking, Non-Volatile Random Access Memory (NVRAM), low-latency InfiniBand interconnects, disk controllers and storage media. Each node in the distributed cluster thus has compute or processing capabilities as well as storage or capacity capabilities.

An Isilon cluster starts with as few as three-nodes, and currently scales to 144-nodes (governed by the largest, 144-port InfiniBand switch that Isilon has qualified). There are many different types of nodes, all of which can be incorporated into a single cluster where different nodes provide different ratios of capacity to throughput or Input/Output operations per second (IOPS).

OneFS has no built-in limitation in terms of the number of nodes that can be included in a single system. Each node added to a cluster increases aggregate disk, cache, CPU, and network capacity. OneFS leverages each of the hardware building blocks, so that the whole becomes greater than the sum of the parts. The RAM is grouped together into a single coherent cache, allowing I/O on any part of the cluster to benefit from data cached anywhere. NVRAM is grouped together to allow for high-throughput writes that are safe across power failures. Spindles and CPU are combined to increase throughput, capacity and IOPS as the cluster grows, for access to one file or for multiple files. A cluster’s storage capacity can range from a minimum of 18 terabytes (TB) to a maximum of 20 petabytes (PB). The maximum capacity will continue to increase as disk drives continue to get denser.

The available Isilon nodes today are broken into several classes, according to their functionality:

- S-Series: IOPS-intensive applications
- X-Series: High-concurrency and throughput-driven workflows
- NL-Series: Near-primary accessibility, with near-tape value
- Performance Accelerator: Independent scaling for ultimate performance
- Backup Accelerator: High-speed and scalable backup and restore solution

Network

There are two types of networks associated with a cluster: internal and external.

Back-end network

All intra-node communication in a cluster is performed using a proprietary, unicast (node to node) protocol. Communication uses an extremely fast low-latency, InfiniBand (IB) network. This back-end network, which is configured with redundant switches for high availability, acts as the backplane for the cluster, enabling each node to act as a contributor in the cluster and isolating node-to-node communication to a private, high-speed, low-latency network. This back-end network utilizes Internet Protocol (IP) over IB for node-to-node communication

Front-end network

Clients connect to the cluster using Ethernet connections (1GbE or 10GbE) that are available on all nodes. Because each node provides its own Ethernet ports, the amount of network bandwidth available to the cluster scales linearly with performance and capacity. The Isilon cluster supports standard network communication protocols to a customer network, including NFS, CIFS, HTTP, FTP and HDFS.

Complete cluster view

The complete cluster is combined with hardware, software, networks in the following view:

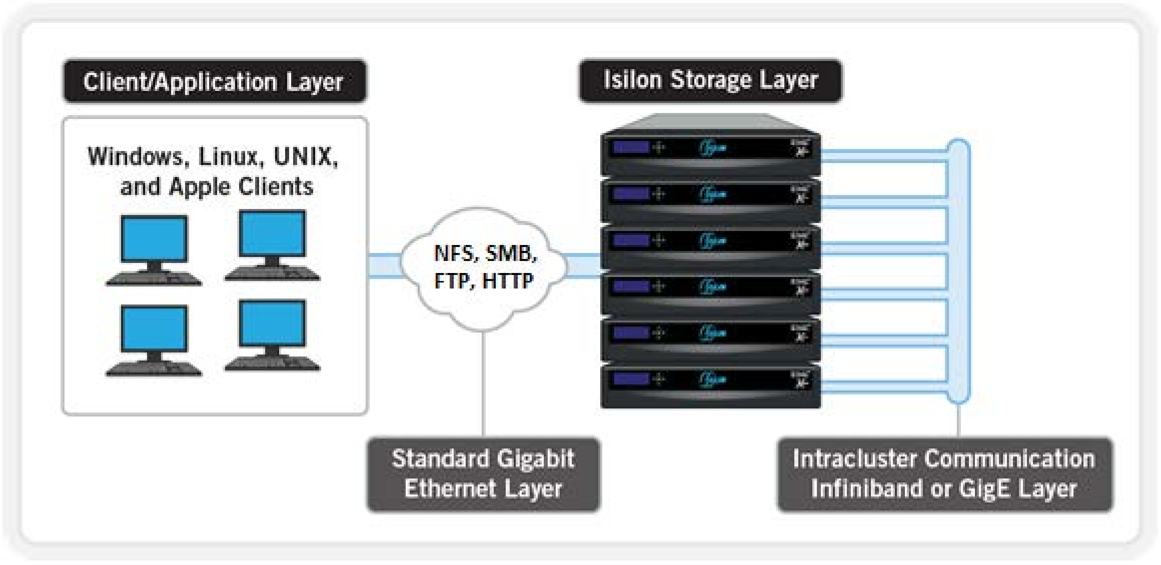
Client/Application Layer
Isilon Storage Layer
Windows, Linux, UNIX, and Apple Clients
NFS, SMB, FTP, HTTP
Standard Gigabit Ethernet Layer
Intracluster Communication Infiniband or GigE Layer

Figure 2: All Components of OneFS at Work

Figure 2 depicts the complete architecture; software, hardware and network all working together in your environment with servers to provide a completely distributed single file system that can scale dynamically as workloads and capacity needs or throughput needs change in a scale-out environment.

OneFS software overview

Operating system

OneFS is built on a BSD-based UNIX Operating System (OS) foundation. It supports both Linux/UNIX and Windows semantics natively, including hard links, delete-on-close, atomic rename, ACLs, and extended attributes. It uses BSD as its base OS because it is a mature and proven Operating System and the open source community can be leveraged for innovation.

Client services

The front-end protocols that the clients can use to interact with OneFS are referred to as client services. Please refer to the Supported Protocols section for a detailed list of supported protocols. In order to understand, how OneFS communicates with clients, we split the I/O subsystem into two halves: the top half or the Initiator and the bottom half or the Participant. Every node in the cluster is a Participant for a particular I/O operation. The node that the client connects to is the Initiator and that node acts as the "captain" for the entire I/O operation. The read and write operation are detailed in later sections

Cluster operations

In a clustered architecture, there are cluster jobs that are responsible for taking care of the health and maintenance of the cluster itself—these jobs are all managed by the OneFS job engine. The Job Engine runs across the entire cluster and is responsible for dividing and conquering large storage management and protection tasks. To achieve this, it reduces a task into smaller work items and then allocates, or maps, these portions of the overall job to multiple worker threads on each node. Progress is tracked and reported on throughout job execution and a detailed report and status is presented upon completion or termination.

Job Engine includes a comprehensive check-pointing system which allows jobs to be paused and resumed, in addition to stopped and started. The Job Engine framework also includes an adaptive impact management system.

The Job Engine typically executes jobs as background tasks across the cluster, using spare or especially reserved capacity and resources. The jobs themselves can be categorized into three primary classes:

- **File System Maintenance Jobs**

  These jobs perform background file system maintenance, and typically require access to all nodes. These jobs are required to run in default configurations, and often in degraded cluster conditions. Examples include file system protection and drive rebuilds.

- **Feature Support Jobs**

  The feature support jobs perform work that facilitates some extended storage management function, and typically only run when the feature has been configured. Examples include deduplication and anti-virus scanning.

- **User Action Jobs**

  These jobs are run directly by the storage administrator to accomplish some data management goal. Examples include parallel tree deletes and permissions maintenance.

The table below provides a comprehensive list of the exposed Job Engine jobs, the operations they perform, and their respective file system access methods:

Job Name	Job Description	Access Method
AutoBalance	Balances free space in the cluster.	Drive + LIN Scans
AutoBalanceLin	Balances free space in the cluster.	LIN Scan
AVScan	Virus scanning job that ICAP server(s) run.	Treewalk
Collect	Reclaims disk space that could not be freed due to a node or drive being unavailable while they suffer from various failure conditions.	Drive + LIN Scans
Dedupe	Deduplicates identical blocks in the file system.	Treewalk
DedupeAssessment	Dry run assessment of the benefits of deduplication.	Treewalk
DomainMark	Associates a path and its contents with a domain.	Treewalk

FlexProtect	Rebuilds and re-protects the file system to recover from a failure scenario.	Drive + LIN Scans
FlexProtectLin	Re-protects the file system.	LIN Scan
FSAnalyze	Gathers file system analytics data that is used in conjunction with Isilon InsightIQ™ software.	LIN Scan
IntegrityScan	Performs online verification and correction of any file system inconsistencies.	LIN Scan
MediaScan	Scans drives for media-level errors.	Drive + LIN Scans
MultiScan	Runs Collect and AutoBalance jobs concurrently.	LIN Scan
PermissionRepair	Correct permissions of files and directories.	Treewalk
QuotaScan	Updates quota accounting for domains created on an existing directory path.	Treewalk
SetProtectPlus	Applies the default file policy. This job is disabled if SmartPools is activated on the cluster.	LIN Scan
ShadowStoreDelete	Frees space associated with a shadow store.	LIN Scan
SmartPools	Job that runs and moves data between the tiers of nodes within the same cluster.	LIN Scan
SnapRevert	Reverts an entire snapshot back to head.	
SnapshotDelete	Frees disk space that is associated with deleted snapshots.	LIN Scan
TreeDelete	Deletes a path in the file system directly from the cluster itself.	Treewalk

Figure 3: OneFS Job Engine Job Descriptions

Although the file system maintenance jobs are run by default, either on a schedule or in reaction to a particular file system event, any Job Engine job can be managed by configuring both its priority-level (in relation to other jobs) and its impact policy.

An impact policy can consist of one or many impact intervals, which are blocks of time within a given week. Each impact interval can be configured to use a single pre-defined impact-level which specifies the amount of cluster resources to use for a particular cluster operation. Available job engine impact-levels are:

- **Paused**
- **Low**
- **Medium**
- **High**

This degree of granularity allows impact intervals and levels to be configured per job, in order to ensure smooth cluster operation. And the resulting impact policies dictate when a job runs and the resources that a job can consume.

Additionally, Job Engine jobs are prioritized on a scale of one to ten, with a lower value signifying a higher priority. This is similar in concept to the UNIX scheduling utility, 'nice'.

Beginning with OneFS 7.1, the Job Engine allows up to three jobs to be run simultaneously. This concurrent job execution is governed by the following criteria:

- Job Priority
- Exclusion Sets - jobs which cannot run together (ie, FlexProtect and AutoBalance)
- Cluster health - most jobs cannot run when the cluster is in a degraded state.

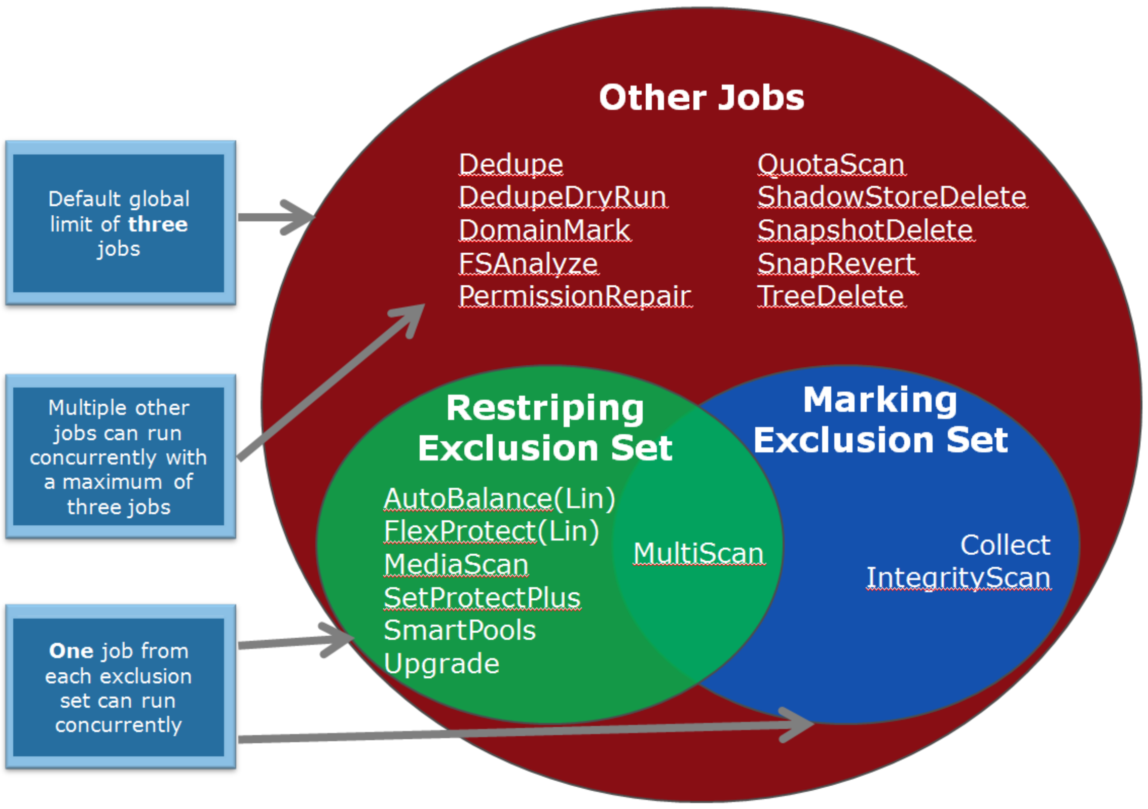

Figure 4: OneFS Job Engine Exclusion Sets

File system structure

The OneFS file system is based on the UNIX file system (UFS) and, hence, is a very fast distributed file system. Each cluster creates a single namespace and file system. This means that the file system is distributed across all nodes in the cluster and is accessible by clients connecting to any node in the cluster. There is no partitioning, and no need for volume creation. Instead of limiting access to free space and to non-authorized files at the physical volume-level, OneFS provides for the same functionality in software via share and file permissions, and via the Isilon SmartQuotas™ service, which provides directory-level quota management.

Because all information is shared among nodes across the internal network, data can be written to or read from any node, thus optimizing performance when multiple users are concurrently reading and writing to the same set of data.

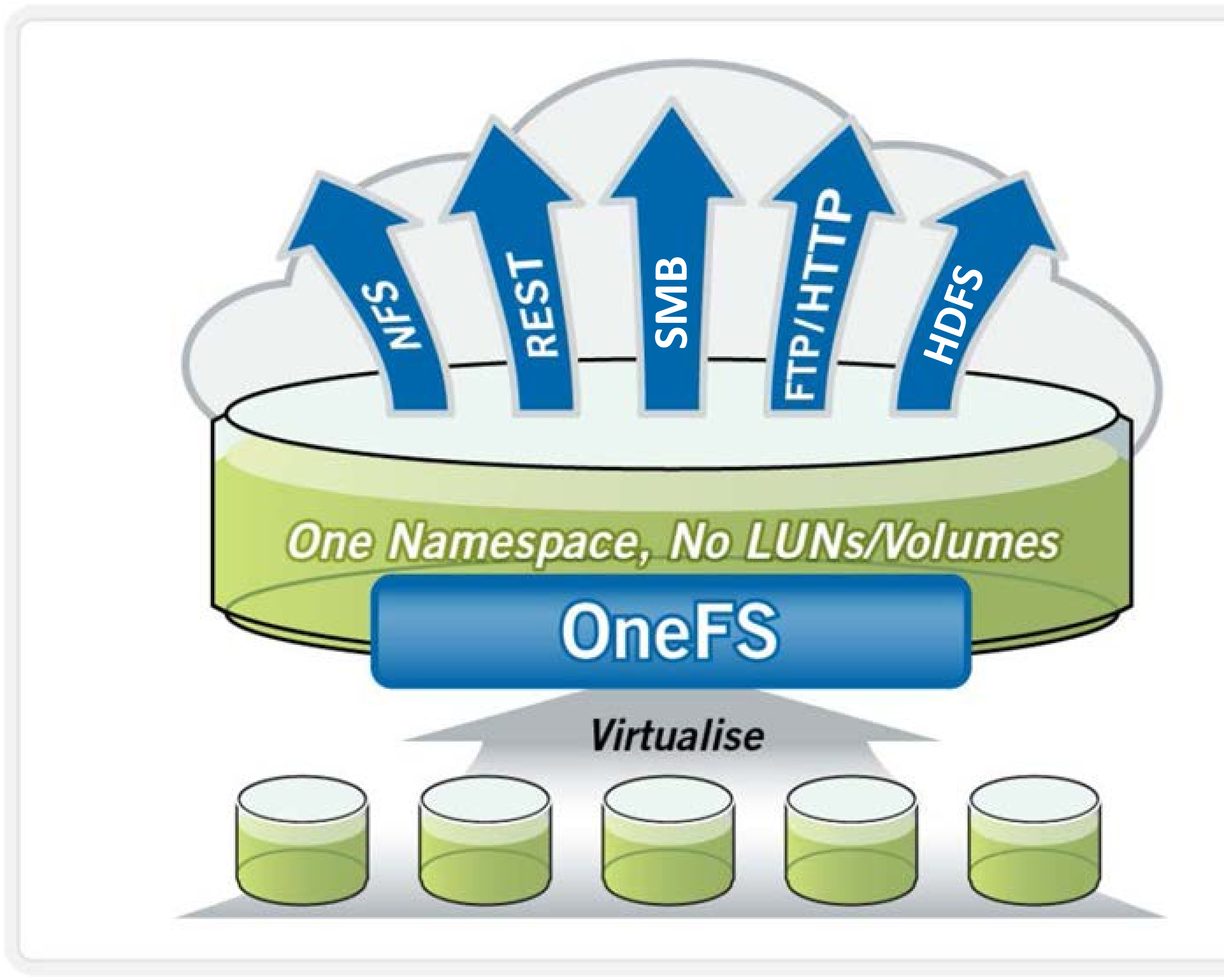

Figure 5: Single File System with Multiple Access Protocols

OneFS is truly a single file system with one namespace. Data and metadata are striped across the nodes for redundancy and availability. The storage has been completely virtualized for the users and administrator. The file tree can grow organically without requiring planning or oversight about how the tree grows or how users use it. No special thought has to be applied by the administrator about tiering files to the appropriate disk, because Isilon SmartPools will handle that automatically without disrupting the single tree. No special consideration needs to be given to how one might replicate such a large tree, because the Isilon SyncIQ™ service automatically parallelizes the transfer of the file tree to one or more alternate clusters, without regard to the shape or depth of the file tree.

This design should be compared with namespace aggregation, which is a commonly-used technology to make traditional NAS “appear” to have a single namespace. With namespace aggregation, files still have to be managed in separate volumes, but a simple “veneer” layer allows for individual directories in volumes to be “glued” to a “top-level” tree via symbolic links. In that model, LUNs and volumes, as well as volume limits, are still present. Files have to be manually moved from volume-to-volume in order to load-balance. The administrator has to be careful about how the tree is laid out. Tiering is far from seamless and requires significant and continual intervention. Failover requires mirroring files between volumes, driving down efficiency and ramping up purchase cost, power and cooling. Overall the administrator burden when using namespace aggregation is higher than it is for a simple traditional NAS device. This prevents such infrastructures from growing very large.

Data layout

OneFS uses physical pointers and extents for metadata and stores file and directory metadata in inodes. B-trees are used extensively in the file system, allowing scalability to billions of objects and near-instant lookups of data or metadata. OneFS is a completely symmetric and highly distributed file system. Data and metadata are always redundant across multiple hardware devices. Data is protected using erasure coding across the nodes in the cluster, this creates a cluster that has high-efficiency, allowing 80% or better raw-to-usable on clusters of five nodes or more. Metadata (which makes up generally less than 1% of the system) is mirrored in the cluster for performance and availability. As OneFS is not reliant on RAID, the amount of redundancy is selectable by the administrator, at the file- or directory-level beyond the defaults of the cluster. Metadata access and locking tasks are managed by all nodes collectively and equally in a peer-to-peer architecture. This symmetry is key to the simplicity and resiliency of the architecture. There is no single metadata server, lock manager or gateway node.

Because OneFS must access blocks from several devices simultaneously, the addressing scheme used for data and metadata is indexed at the physical-level by a tuple of {node, drive, offset}. For example if 12345 was a block address for a block that lived on disk 2 of node 3, then it would read, {3,2,12345}. All metadata within the cluster is multiply mirrored for data protection, at least to the level of redundancy of the associated file. For example, if a file were at an erasure-code protection of “N+2”, implying the file could withstand two simultaneous failures, then all metadata needed to access that file would be 3x mirrored, so it too could withstand two failures. The file system inherently allows for any structure to use any and all blocks on any nodes in the cluster.

Other storage systems send data through RAID and volume management layers, introducing inefficiencies in data layout and providing non-optimized block access. Isilon OneFS controls the placement of files directly, down to the sector-level on any drive anywhere in the cluster. This allows for optimized data placement and I/O patterns and avoids unnecessary read-modify-write operations. By laying data on disks in a file-by-file manner, OneFS is able to flexibly control the type of striping as well as the redundancy level of the storage system at the system, directory, and even file-levels. Traditional storage systems would require that an entire RAID volume be dedicated to a particular performance type and protection setting. For example, a set of disks might be arranged in a RAID 1+0 protection for a database. This makes it difficult to optimize spindle use over the entire storage estate (since idle spindles cannot be borrowed) and also leads to inflexible designs that do not adapt with the business requirement. OneFS allows for individual tuning and flexible changes at any time, fully online.

File writes

The OneFS software runs on all nodes equally - creating a single file system that runs across every node. No one node controls or "masters" the cluster; all nodes are true peers.

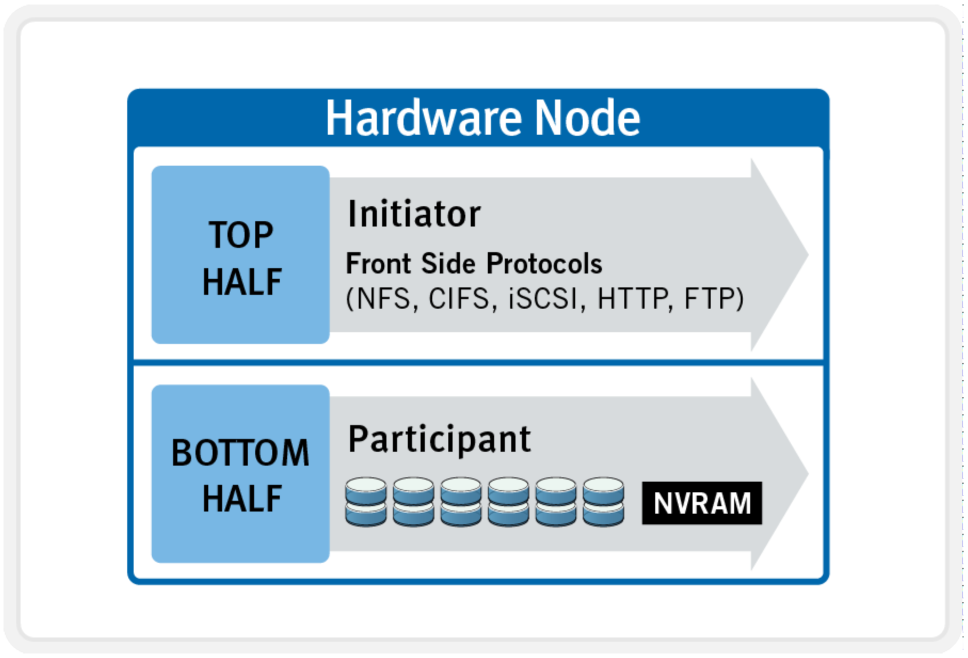

Figure 6: Model of Node Components Involved in I/O

If we were to look at all the components within every node of a cluster that are involved in I/O from a high-level, it would look like Figure 6 above. We have split the stack into a "top" layer, called the Initiator, and a "bottom" layer, called the Participant. This division is used as a "logical model" for the analysis of any one given read or write. At a physical-level, CPUs and RAM cache in the nodes are simultaneously handling Initiator and Participant tasks for I/O taking place throughout the cluster. There are caches and a distributed lock manager that are excluded from the diagram above to keep it simple. They will be covered in later sections of the paper.

When a client connects to a node to write a file, it is connecting to the top half or Initiator of that node. Files are broken into smaller logical chunks called stripes before being written to the bottom half or Participant of a node (disk). Failure-safe buffering using a write coalescer is used to ensure that writes are efficient and read-modify-write operations are avoided. The size of each file chunk is referred to as the stripe unit size.

OneFS stripes data across all nodes—and not simply across disks—and protects the files, directories and associated metadata via software erasure-code or mirroring technology. For data, OneFS can use (at the administrator's discretion) either the Reed-Solomon erasure coding system for data protection, or (less commonly) mirroring. Mirroring, when applied to user data, tends to be used more for high-transaction performance cases. The bulk of user data will generally use erasure coding, as it provides extremely high performance without sacrificing on-disk efficiency. Erasure coding can provide beyond 80% efficiency on raw disk with five nodes or more, and on large clusters can even do so while providing quadruple-level redundancy. The stripe width for any given file is the number of nodes (not disks) that a file is written across. It is determined by the number of nodes in the cluster, the size of the file, and the protection setting (for example, N+2).

OneFS uses advanced algorithms to determine data layout for maximum efficiency and performance. When a client connects to a node, that node's initiator acts as the "captain" for the write data layout of that file. In an Isilon cluster, data, parity, metadata and inodes are all distributed on multiple nodes, and even across multiple drives within nodes. Figure 7 below shows a file write happening across all nodes in a 3 node cluster.

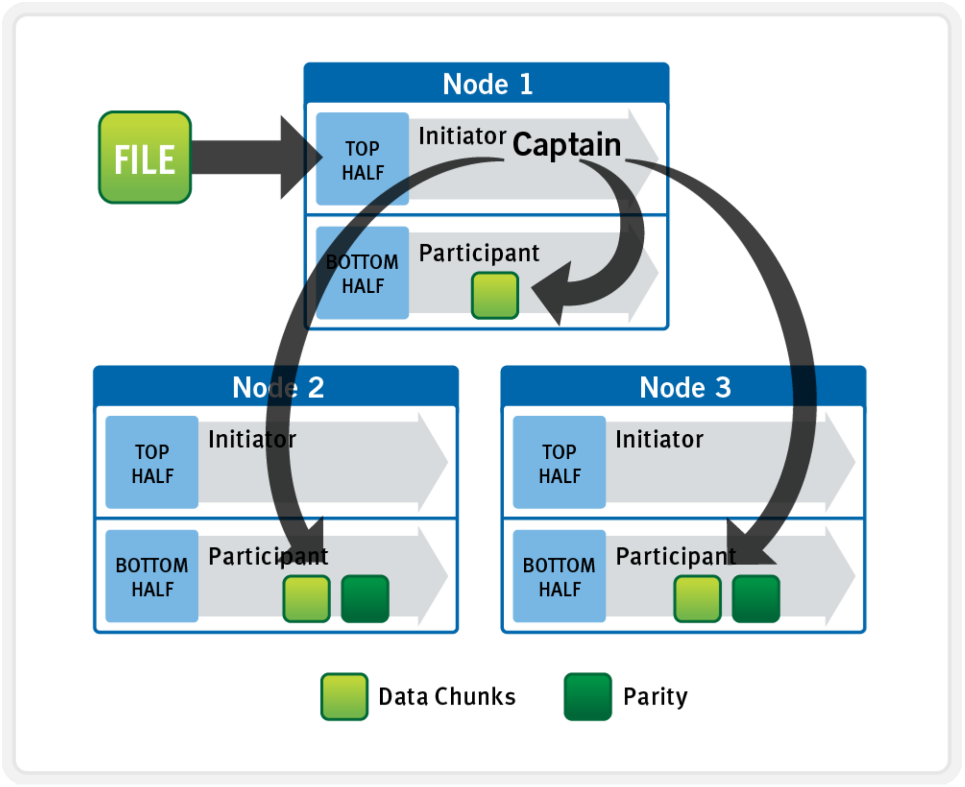

Figure 7: A File Write Operation on a 3-node Isilon Cluster

OneFS uses the InfiniBand back-end network to allocate and stripe data across all nodes in the cluster automatically, so no additional processing is required. As data is being written, it is being protected at the specified level.

When writes take place, OneFS divides data out into atomic units called protection groups. Redundancy is built into protection groups, such that if every protection group is safe, then the entire file is safe. For files protected by erasure codes, a protection group consists of a series of data blocks as well as a set of erasure codes for those data blocks; for mirrored files, a protection group consists of all of the mirrors of a set of blocks. OneFS is capable of switching the type of protection group used in a file dynamically, as it is writing. This can allow many additional functionalities including, for example, allowing the system to continue without blocking in situations when temporary node failures in the cluster would prevent the desired number of erasure codes from being used. Mirroring can be used temporarily in these cases to allow writes to continue. When nodes are restored to the cluster, these mirrored protection groups are converted back seamlessly and automatically to erasure-code-protected, without administrator intervention.

The OneFS file system block size is 8KB. A file smaller than 8KB will use a full 8KB block. Depending on the data protection level, this 8KB file could end up using more than 8KB of data space. However, data protection settings are discussed in detail in a later section of this paper. OneFS can support file systems with billions of small files at very high performance, because all of the on-disk structures are designed to scale to such sizes, and provide near-instantaneous access to any one object regardless of the total number of objects. For larger files, OneFS can take advantage of using multiple, contiguous 8KB blocks. In these cases, up to sixteen contiguous blocks can be striped onto a single node’s disk. If a file is 32KB in size, then four contiguous 8KB blocks will be used.

For even larger files, OneFS can maximize sequential performance by taking advantage of a stripe unit consisting of 16 contiguous blocks, for a total of 128KB per stripe unit. During a write, data is broken into stripe units and these are spread across multiple nodes as a protection group. As data is being laid out across the cluster, erasure codes or mirrors, as required, are distributed within each protection group to ensure that files are protected at all times.

One of the key functions of the AutoBalance functionality of OneFS is to reallocate and rebalance data and make storage space more usable and efficient, when possible. In most cases, the stripe width of larger files can be increased to take advantage of new free space (as nodes are added) and to make the on-disk striping more efficient. AutoBalance maintains high on-disk efficiency and eliminates on-disk “hot spots” automatically.

The initiator top half of the “captain” node uses a patented, modified two-phase commit transaction to safely distribute writes to multiple NVRAMs across the cluster, as shown in Figure 8 below.

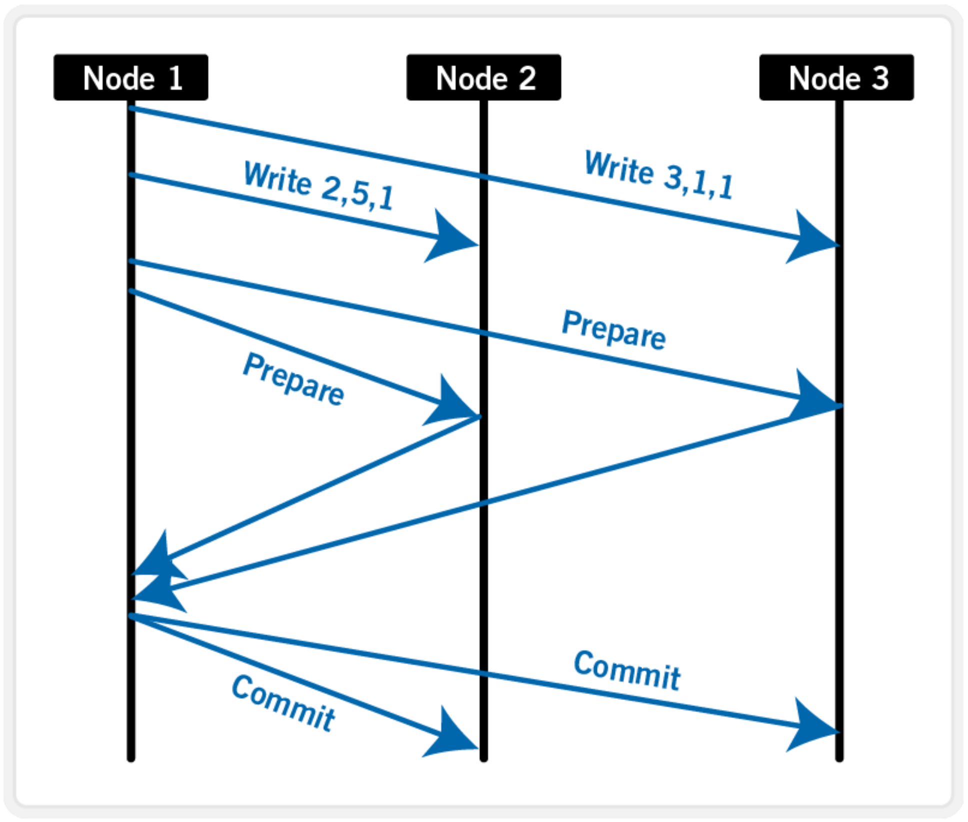

Figure 8: Distributed Transactions and Two-Phase Commit

Every node that owns blocks in a particular write is involved in a two-phase commit. The mechanism relies on NVRAM for journaling all the transactions that are occurring across every node in the storage cluster. Using multiple NVRAMs in parallel allows for high-throughput writes while maintaining data safety against all manner of failures, including power failures. In the event that a node should fail mid-transaction, the transaction is restarted instantly without that node involved. When the node returns, the only required actions are for the node to replay its journal from NVRAM—which takes seconds or minutes—and, occasionally, for AutoBalance to rebalance files that were involved in the transaction. No expensive “fsck” or “disk-check” processes are ever required. No drawn-out resynchronization ever needs to take place. Writes are never blocked due to a failure. The patented transaction system is one of the ways that OneFS eliminates single—and even multiple—points of failure.

In a write operation, the initiator “captains” or orchestrates the layout of data and metadata; the creation of erasure codes; and the normal operations of lock management and permissions control. An administrator from the web management or CLI interface at any point can optimize layout decisions made by OneFS to better suit the workflow. The administrator can choose from the access patterns below at a per-file or directory-level:

- **Concurrency:** Optimizes for current load on the cluster, featuring many simultaneous clients. This setting provides the best behavior for mixed workloads.
- **Streaming:** Optimizes for high-speed streaming of a single file, for example to enable very fast reading with a single client.

- **Random:** Optimizes for unpredictable access to the file, by adjusting striping and disabling the use of any prefetch cache.

Write caching

Write caching accelerates the process of writing data to an Isilon cluster. This is achieved by batching up smaller write requests and sending them to disk in bigger chunks, removing a significant amount of disk writing latency. When clients write to the cluster, OneFS temporarily writes the data to an NVRAM-based journal cache on the initiator node, instead of immediately writing to disk. OneFS can then flush these cached writes to disk at a later, more convenient time. Additionally, these writes are also mirrored to participant nodes' NVRAM journals to satisfy the file's protection requirement. Therefore, in the event of a cluster split or unexpected node outage, uncommitted cached writes are fully protected.

The write cache operates as follows:

- An NFS client sends Node 1 a write request for a file with N+2 protection.
- Node 1 accepts the writes into its NVRAM write cache (fast path) and then mirrors the writes to participant nodes' log files for protection.
- Write acknowledgements are returned to the NFS client immediately and as such, write to disk latency is avoided.
- As Node 1's write cache fills, it is periodically flushed and writes are committed to disk via the two phase commit process (described above) with the appropriate parity protection applied (N+2).
- The write cache and participant node log files are cleared and available to accept new writes.

File reads

In an Isilon cluster, data, metadata and inodes are all distributed on multiple nodes, and even across multiple drives within nodes. When reading or writing to the cluster, the node a client attaches to acts as the "captain" for the operation.

In a read operation, the "captain" node gathers all of the data from the various nodes in the cluster and presents it in a cohesive way to the requestor.

Due to the use of cost-optimized industry standard hardware, the Isilon cluster provides a high ratio of cache to disk (multiple GB per node) that is dynamically allocated for read and write operations as needed. This RAM-based cache is unified and coherent across all nodes in the cluster, allowing a client read request on one node to benefit from I/O already transacted on another node. These cached blocks can be quickly accessed from any node across the low-latency InfiniBand backplane, allowing for a large, efficient RAM cache, which greatly accelerates read performance. As the cluster grows larger, the cache benefit increases. For this reason, the amount of I/O to disk on an Isilon cluster is generally substantially lower than it is on traditional platforms, allowing for reduced latencies and a better user experience.

For files marked with an access pattern of concurrent or streaming, OneFS can take advantage of pre-fetching of data based on heuristics used by the Isilon SmartRead component. SmartRead can create a data “pipeline” from L2 cache, prefetching into a local “L1” cache on the “captain” node. This greatly improves sequential-read performance across all protocols, and means that reads come directly from RAM within milliseconds. For high-sequential cases, SmartRead can very aggressively prefetch ahead, allowing reads or writes of individual files at very high data rates.

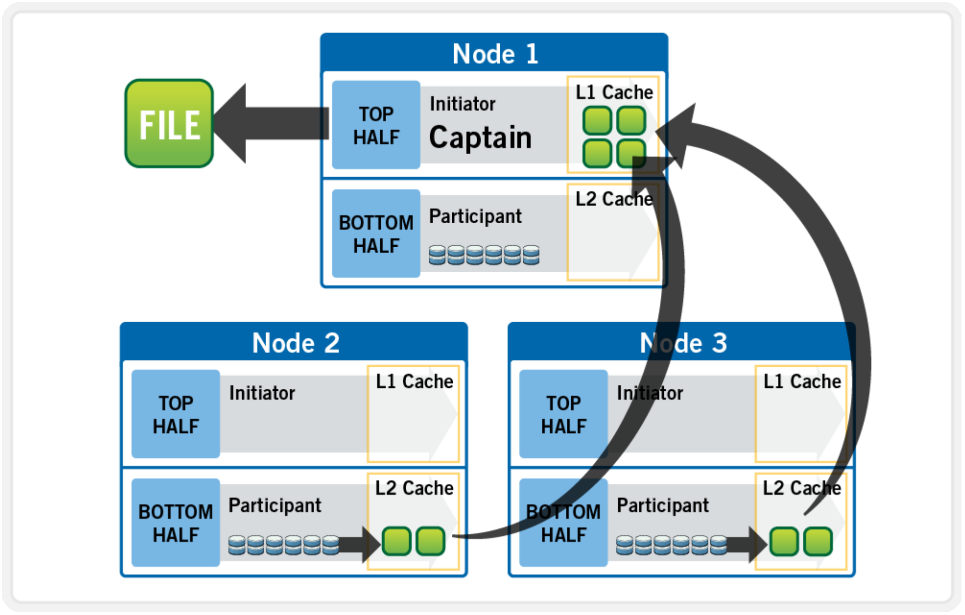

Figure 9: A File Read Operation on a 3-node Isilon Cluster

Figure 9 illustrates how SmartRead reads a sequentially-accessed, non-cached file that is requested by a client attached to Node1 in a 3-node cluster.

1. Node1 reads metadata to identify where all the blocks of file data exist.
2. Node1 also checks its L1 cache to see if it has the file data being requested.
3. Node1 builds a read pipeline, sending concurrent requests to all nodes that have a piece of file data to retrieve that file data from disk.
4. Each node pulls the blocks of file data from disk into their L2 cache, and transmits the file data to Node1.
5. Node1 records the incoming data to L1 cache, simultaneously serving the file to the client. Meanwhile, the pre-fetching process continues.
6. For highly sequential cases, data in L1 cache may be optionally “dropped behind” to free RAM for other L1 or L2 cache demands.

SmartRead’s intelligent caching allows for very high read performance with high levels of concurrent access. Importantly, it is faster for Node1 to get file data from the cache of Node2 (over the low-latency cluster interconnect) than to access its own local disk. SmartRead’s algorithms control how aggressive the pre-fetching is (disabling pre-fetch for random-access cases) and how long data stays in the cache, and optimizes where data is cached.

Locks and concurrency

OneFS has a fully distributed lock manager that marshals locks on data across all nodes in a storage cluster. The locking manager is highly extensible, and allows for multiple lock "personalities" to support both file system locks as well as cluster-coherent protocol-level locks such as SMB share mode locks or NFS advisory-mode locks. OneFS also has support for delegated locks such as CIFS oplocks and NFSv4 delegations.

Every node in a cluster is a coordinator for locking resources and a coordinator is assigned to lockable resources based upon an advanced hashing algorithm. The way the algorithm is designed is that the coordinator almost always ends up on a different node than the initiator of the request. When a lock is requested for a file, it could be a shared lock (allowing multiple users to share the lock simultaneously, usually for reads) or an exclusive lock (allowing one user at any given moment, typically for writes).

Figure 10 below illustrates an example of how threads from different nodes could request a lock from the coordinator.

1. Node 2 is designated to be the coordinator of these resources.
2. Thread 1 from Node 4 and thread 2 from Node 3 request a shared lock on a file from Node 2 at the same time.
3. Node 2 checks if an exclusive lock exists for the requested file.
4. If no exclusive locks exist, Node 2 grants thread 1 from Node 4 and thread 2 from Node 3 shared locks on the requested file.
5. Node 3 and Node 4 are now performing a read on the requested file.
6. Thread 3 from Node 1 requests an exclusive lock for the same file as being read by Node 3 and Node 4.
7. Node 2 checks with Node 3 and Node 4 if the shared locks can be reclaimed.
8. Node 3 and Node 4 are still reading so Node 2 asks thread 3 from Node 1 to wait for a brief instant.
9. Thread 3 from Node 1 blocks until the exclusive lock is granted by Node 2 and then completes the write operation.

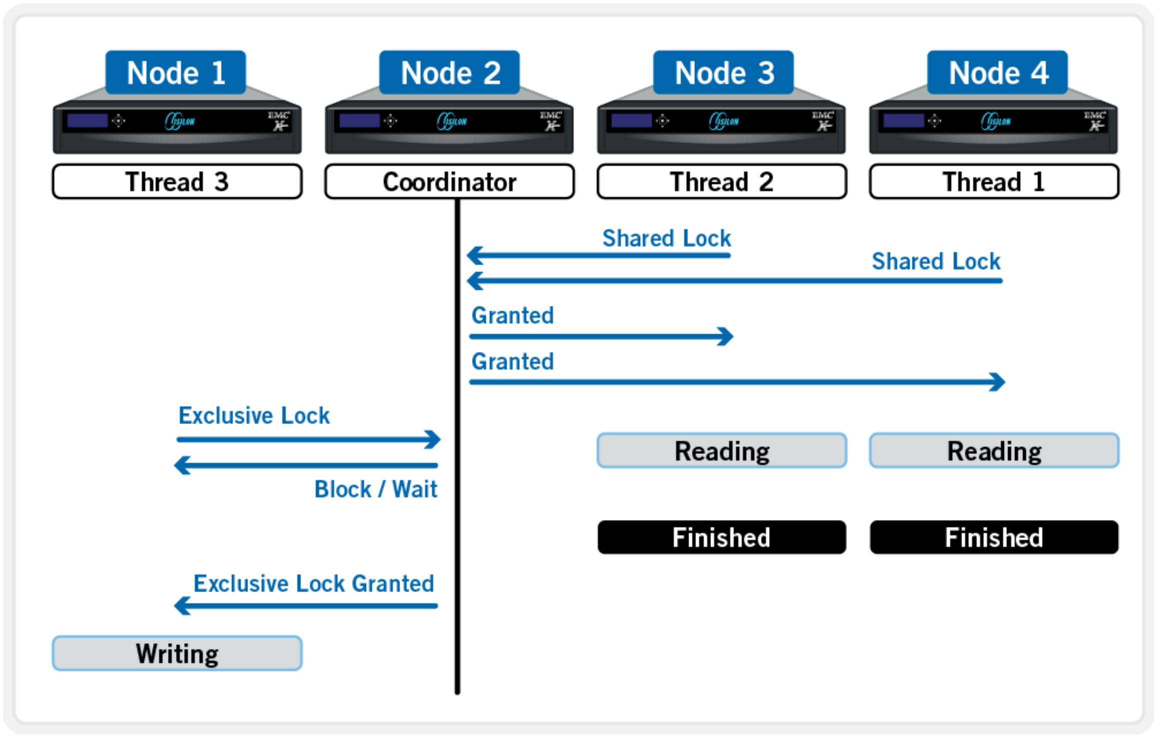

Figure 10: Distributed Lock Manager

Multi-threaded IO

With the growing use of large NFS datastores for server virtualization and enterprise application support comes the need for high throughput and low latency to large files. To accommodate this, OneFS Multi-writer supports multiple threads concurrently writing to individual files.

In the above example, concurrent write access to a large file can become limited by the exclusive locking mechanism, applied at the whole file level. In order to avoid this potential bottleneck, OneFS Multi-writer provides more granular write locking by sub-diving the file into separate regions and granting exclusive write locks to individual regions, as opposed to the entire file. As such, multiple clients can simultaneously write to different portions of the same file.

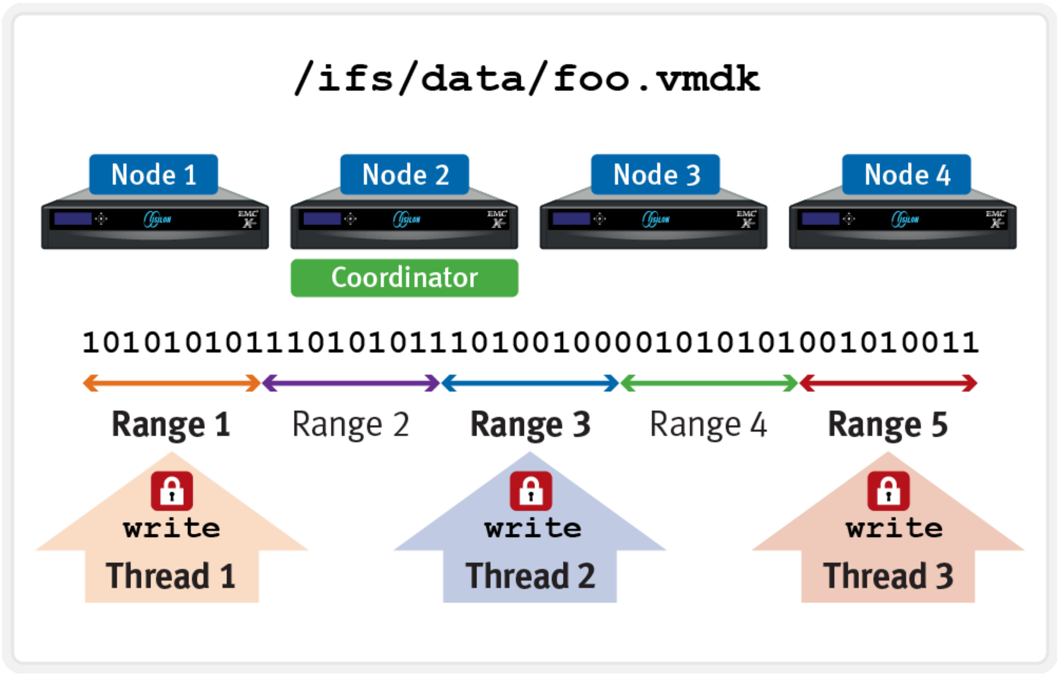

Figure 11: Multi-threaded IO Writer

Data protection

Power loss

A file system journal, which stores information about changes to the file system, is designed to enable fast, consistent recoveries after system failures or crashes, such as power loss. The file system replays the journal entries after a node or cluster recovers from a power loss or other outage. Without a journal, a file system would need to examine and review every potential change individually after a failure (an “fsck” or “chkdsk” operation); in a large file system, this operation can take a long time.

OneFS is a journaled file system in which each node contains a battery-backed NVRAM card used for protecting uncommitted writes to the file system. The NVRAM card battery charge lasts many days without requiring a recharge. When a node boots up, it checks its journal and selectively replays transactions to disk where the journaling system deems it necessary.

OneFS will mount only if it can guarantee that all transactions not already in the system have been recorded. For example, if proper shutdown procedures were not followed, and the NVRAM battery discharged, transactions might have been lost; to prevent any potential problems, the node will not mount the file system.

Hardware failures and quorum

In order for the cluster to properly function and accept data writes, a quorum of nodes must be active and responding. A quorum is defined as a simple majority: a cluster with x nodes must have $\lfloor x/2 \rfloor +1$ nodes online in order to allow writes. For example, in a seven-node cluster, four nodes would be required for a quorum. If a node or group of nodes is up and responsive, but is not a member of a quorum, it runs in a read-only state.

The Isilon system uses a quorum to prevent “split-brain” conditions that can be introduced if the cluster should temporarily split into two clusters. By following the quorum rule, the architecture guarantees that regardless of how many nodes fail or come back online, if a write takes place, it can be made consistent with any previous writes that have ever taken place. The quorum also dictates the number of nodes required in order to move to a given data protection level. For an erasure-code-based protection-level of $N+M$, the cluster must contain at least $2M+1$ nodes. For example, a minimum of seven nodes is required for an N+3 configuration; this allows for a simultaneous loss of three nodes while still maintaining a quorum of four nodes for the cluster to remain fully operational. If a cluster does drop below quorum, the file system will automatically be placed into a protected, read-only state, denying writes, but still allowing read access to the available data.

Hardware failures—add / remove nodes

A system called the group management protocol enables global knowledge of the cluster state at all times, and guarantees a consistent view across the entire cluster of the state of all other nodes. If one or more nodes become unreachable over the cluster interconnect, the group is “split” or removed from the cluster. All nodes resolve to a new consistent view of their cluster. (Think of this as if the cluster were splitting into two separate groups of nodes, though note that only one group can have quorum.) While in this split state, all data in the file system is reachable and, for the side maintaining quorum, modifiable. Any data stored on the “down” device is rebuilt using the redundancy stored in the cluster.

If the node becomes reachable again, a “merge” or add occurs, bringing node(s) back into the cluster. (The two groups merge back into one.) The node can rejoin the cluster without being rebuilt and reconfigured. This is unlike hardware RAID arrays, which require drives to be rebuilt. AutoBalance may restripe some files to increase efficiency, if some of their protection groups were overwritten and transformed to narrower stripes during the split.

The OneFS Job Engine also includes a process called Collect, which acts as an orphan collector. When a cluster splits during a write operation, some blocks that were allocated for the file may need to be re-allocated on the quorum side. This will “orphan” allocated blocks on the non-quorum side. When the cluster re-merges, the Collect job will locate these orphaned blocks through a parallelized mark-and-sweep scan, and reclaim them as free space for the cluster.

Scalable rebuild

OneFS does not rely on hardware RAID either for data allocation, or for reconstruction of data after failures. Instead OneFS manages protection of file data directly, and when a failure occurs, it rebuilds data in a parallelized fashion. OneFS is able to determine which files are affected by a failure in constant time, by reading inode data in a linear manor, directly off disk. The set of affected files are assigned to a set of worker threads that are distributed among the cluster nodes by the job engine. The worker nodes repair the files in parallel. This implies that as cluster size increases, the time to rebuild from failures decreases. This has an enormous efficiency advantage in maintaining the resiliency of clusters as their size increases.

Virtual hot spare

Most traditional storage systems based on RAID require the provisioning of one or more “hot spare” drives to allow independent recovery of failed drives. The hot spare drive replaces the failed drive in a RAID set. If these hot spares are not themselves replaced before more failures appear, the system risks a catastrophic data loss. OneFS avoids the use of hot spare drives, and simply borrows from the available free space in the system in order to recover from failures; this technique is called virtual hot spare. In doing so, it allows the cluster to be fully self-healing, without human intervention. The administrator can create a virtual hot spare reserve, allowing the system to self-heal despite ongoing writes by users.

N + M data protection

An Isilon cluster is designed to tolerate one or more simultaneous component failures, without preventing the cluster from serving data. To achieve this, OneFS will protect files with either parity based protection, via Reed-Solomon error correction (N+M protection), or a mirroring system. Data protection is applied in software at the file-level, enabling the system to focus on recovering only those files that are compromised by a failure, rather than having to check and repair an entire file-set or volume. OneFS metadata and inodes are always protected by mirroring, rather than Reed-Solomon coding, and with at least the level of protection as the data they reference.

Because all data, metadata, and parity information is distributed across the nodes of the cluster, the Isilon cluster does not require a dedicated parity node or drive, or a dedicated device or set of devices to manage metadata. This ensures that no one node can become a single point of failure. All nodes share equally in the tasks to be performed, providing perfect symmetry and load-balancing in a peer-to-peer architecture.

The Isilon system provides several levels of configurable data protection settings, which you can modify at any time without needing to take the cluster or file system offline.

For a file protected with erasure codes, we say that each of its protection groups is protected at a level of $N+M/b$, where $N>M$ and $M>=b$. The values N and M represent, respectively, the number of drives used for data and for erasure codes within the protection group. The value of b relates to the number of data stripes used to lay out that protection group, and is covered below. A common and easily-understood case is where $b=1$, implying that a protection group incorporates: N drives worth of data; M drives worth of redundancy, stored in erasure codes; and that the protection group should be laid out over exactly one stripe across a set of nodes. This allows for M members of the protection group to fail simultaneously and still provide 100% data availability. The M erasure code members are computed from the N data members. Figure 12 below shows the case for a regular 4+2 protection group *($N=4$, $M=2$, $b=1$).*

Because OneFS stripes files across nodes, this implies that files striped at $N+M$ can withstand M simultaneous node failures without loss of availability. OneFS therefore provides resiliency across any type of failure, whether it be to a drive, a node, or a component within a node (say, a card). Furthermore, a node counts as a single failure, regardless of the number or type of components that fail within it. Therefore, if five drives fail in a node, it only counts as a single failure for the purposes of *N+M* protection.

OneFS can uniquely provide a variable level of M, up to four, providing for quadruple-failure protection. This goes far beyond the maximum level of RAID commonly in use today, which is the double-failure protection of RAID-6. Because the reliability of the storage increases geometrically with this amount of redundancy, N+4 protection can be orders of magnitude more reliable than traditional hardware RAID. This added protection means that large capacity SATA drives, such as 3 TB and 4 TB drives, can be added with confidence.

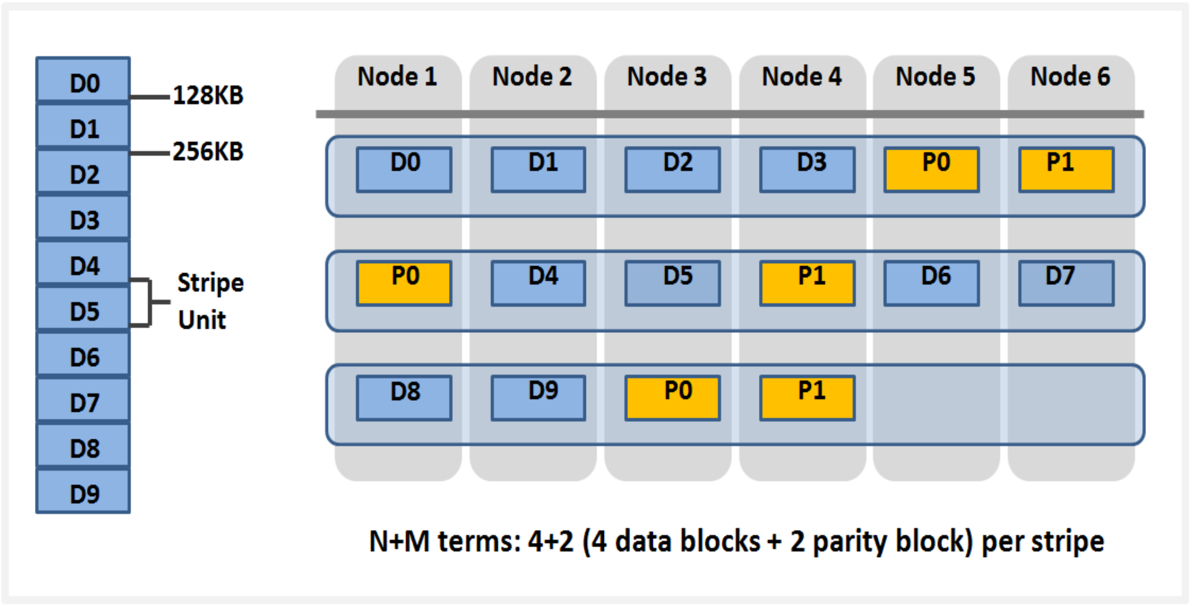

Figure 12: OneFS Redundancy – N+M Parity Protection

Smaller clusters can be protected with N+1 protection, but this implies that while a single drive or node could be recovered, two drives in two different nodes could not. Drive failures are orders of magnitude more likely than node failures. For clusters with large drives, it is desirable to provide protection for multiple drive failures, though single-node recoverability is acceptable.

To provide for a situation where we wish to have double-disk redundancy and single-node redundancy, we can build up “double-length” protection groups of size. These double-length protection groups will “wrap” once over the same set of nodes, as they are laid out. Since each protection group contains exactly two disks worth of redundancy, this mechanism will allow a cluster to sustain either a double-drive failure or a full node failure, without any data unavailability.

Most important for small clusters, this method of striping is highly efficient, with an on-disk efficiency of $M/(N+M)$. For example, on a cluster of five nodes with double-failure protection, were we to use N=3, M=2, we would obtain a 3+2 protection group with an efficiency of $1-2/5$ or 60%. Using the same 5-node cluster but with each protection group laid out over 2 stripes, N would now be 8 and M=2, so we could obtain *1-2/(8+2)* or 80% efficiency on disk, retaining our double-drive failure protection and sacrificing only double-node failure protection.

OneFS supports erasure code protection levels of N+1, N+2, N+3, and N+4 (from one to four drive or complete node failures per cluster), as well as N+2:1 (double-drive and single-node tolerant) and N+3:1 (triple-drive and single-node tolerant) protection levels. Additionally, mirrored protection levels between 2x and 8x (mirrored to eight-times mirrored) are also provided for both file and metadata. OneFS data protection is also extremely flexible, allowing protection to be applied to individual files, directories and their contents, or the entire file system.

OneFS enables an administrator to modify the protection policy in real time, while clients are attached and are reading and writing data. Note that increasing a cluster's protection level may increase the amount of space consumed by the data on the cluster.

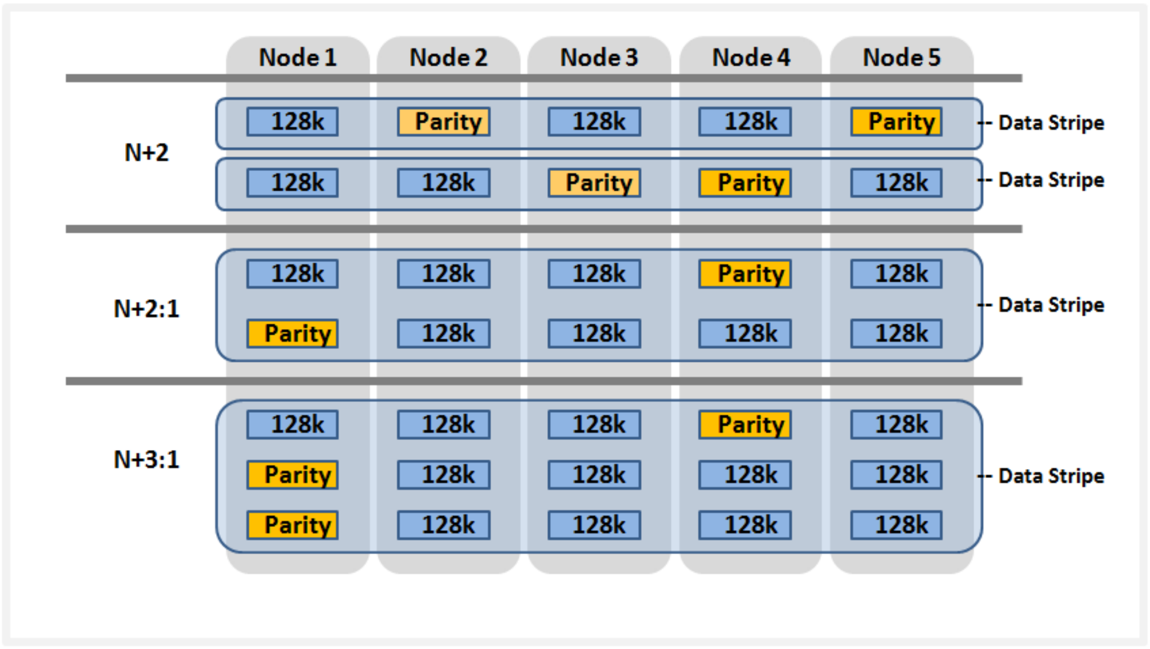

Figure 13: OneFS Hybrid Parity Protection Schemes (N+M:x)

Automatic partitioning

Data tiering and management in OneFS is handled by the SmartPools framework. From a data protection and layout efficiency point of view, SmartPools facilitates the subdivision of large numbers of high-capacity, homogeneous nodes into smaller, more Mean Time to Data Loss (MTTDL)-friendly disk pools. For example, an 80-node nearline (NL) cluster would typically run at a +4 protection level. However, partitioning it into four, twenty node disk pools would allow each pool to run at +2:1, thereby lowering the protection overhead and improving space utilization, without any net increase in management overhead.

In keeping with the goal of storage management simplicity, OneFS will automatically calculate and partition the cluster into pools of disks, or 'node pools', which are optimized for both MTTDL and efficient space utilization. This means that protection level decisions, such as the 80-node cluster example above, are not left to the customer - unless desired.

With Automatic Provisioning, every set of equivalent node hardware is automatically divided into node pools comprising up to forty nodes and six drives per node. These node pools are protected by default at +2:1, and multiple pools can then be combined into logical tiers and managed with SmartPools file pool policies. By subdividing a node's disks into multiple, separately protected pools, nodes are significantly more resilient to multiple disk failures than previously possible.

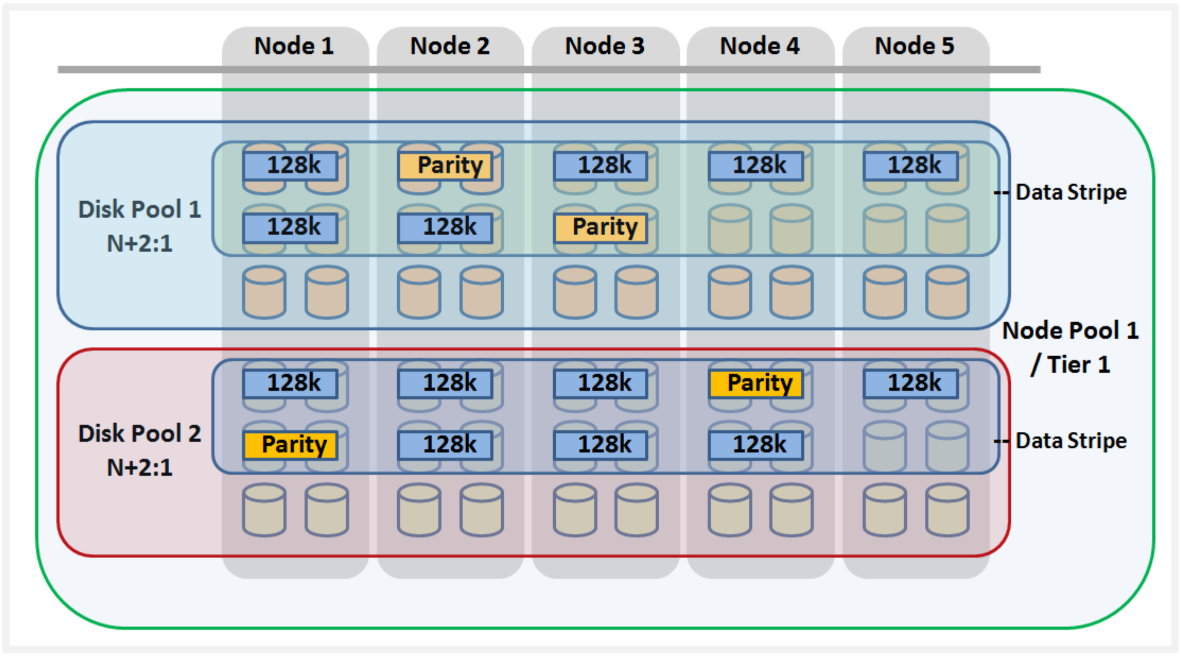

Figure 14: Automatic Partitioning with SmartPools

Supported protocols

Clients with adequate credentials and privileges can create, modify, and read data using one of the standard supported methods for communicating with the cluster:

- NFS (Network File System
- SMB/CIFS (Server Message Block/Common Internet File System)
- FTP (File Transfer Protocol)
- HTTP (Hypertext Transfer Protocol)
- HDFS (Hadoop Distributed File System)
- REST API (Representational State Transfer Application Programming Interface)

By default, only the SMB/CIFS and NFS protocols are enabled in the Isilon cluster. The file system root for all data in the cluster is **/ifs** (the Isilon OneFS file system). This is presented via SMB/CIFS protocol as an 'ifs' share (\\<*cluster_name**ifs*), and via the NFS protocol as an '/ifs' export (*<cluster_name>:/ifs)*.

Note: Data is common between all protocols, so changes made to file content via one access protocol are instantly viewable from all others.

Dynamic scale / scale on demand

Performance and capacity

In contrast to traditional storage systems that must “scale up” when additional performance or capacity is needed, OneFS enables an Isilon storage system to “scale out,” seamlessly increasing the existing file system or volume into petabytes of capacity while increasing performance in tandem in a linear fashion.

Adding capacity and performance capabilities to an Isilon cluster is significantly easier than with other storage systems—requiring only three simple steps for the storage administrator: adding another node into the rack, attaching the node to the InfiniBand network, and instructing the cluster to add the additional node. The new node provides additional capacity and performance since each node includes CPU, memory, cache, network, NVRAM and I/O control pathways.

The AutoBalance feature of OneFS will automatically move data across the InfiniBand network in an automatic, coherent manner so existing data that resides on the cluster moves onto this new storage node. This automatic rebalancing ensures the new node will not become a hot spot for new data and that existing data is able to gain the benefits of a more powerful storage system. The AutoBalance feature of OneFS is also completely transparent to the end user and can be adjusted to minimize impact on high-performance workloads. This capability alone allows OneFS to scale transparently, on-the-fly, from 18 TB up to 20 PB with no added management time for the administrator, or increase in complexity within the storage system.

A large-scale storage system must provide the performance required for a variety of workflows, whether they be sequential, concurrent, or random. Different workflows will exist between applications and within individual applications. OneFS provides for all of these needs simultaneously with intelligent software. More importantly, with OneFS, throughput and IOPS scale linearly with the number of nodes present in a single system. Due to balanced data distribution, automatic rebalancing and distributed processing, OneFS is able to leverage additional CPUs, network ports, and memory as the system scales.

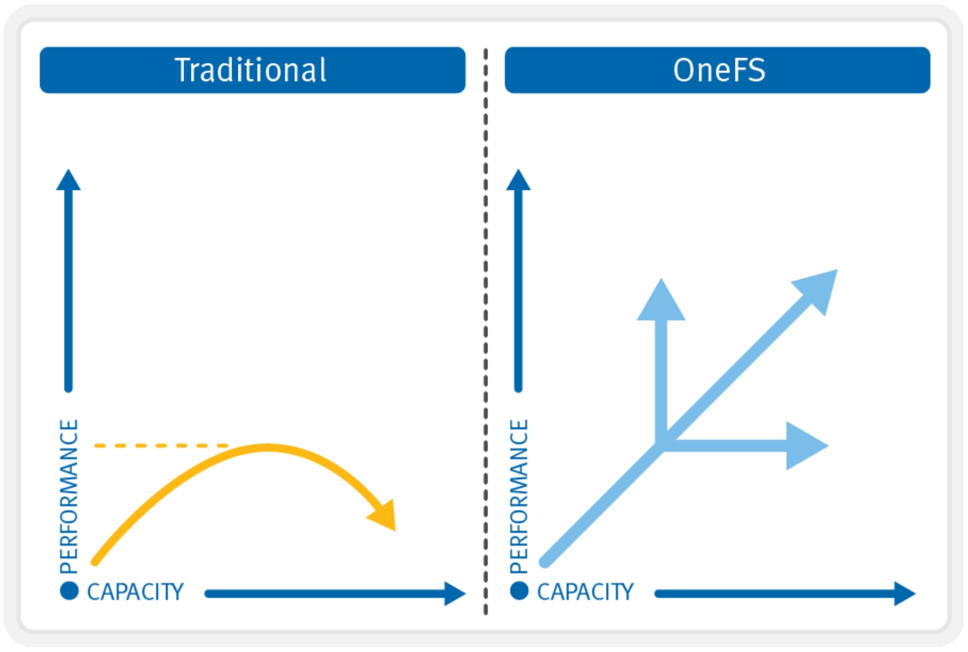

Figure 15: OneFS Linear Scalability

Interfaces

Administrators can use multiple interfaces to administer an Isilon storage cluster in their environments:

- Web Administration User Interface (“WebUI”)
- Command Line Interface via SSH network access or RS232 serial connection
- LCD Panel on the nodes themselves for simple add/remove functions
- RESTful Platform API for programmatic control and automation of cluster configuration and management.

Authentication and access control

Authentication services offer a layer of security by verifying users’ credentials before allowing them to access and modify files. OneFS supports four methods for authenticating users:

- Active Directory (AD)
- LDAP (Lightweight Directory Access Protocol)
- NIS (Network Information Service)
- Local users & Groups

OneFS supports the use of more than one authentication type. However, it is recommended that you fully understand the interactions between authentication types before enabling multiple methods on the cluster. Refer to the product documentation for detailed information about how to properly configure multiple authentication modes.

Active Directory

Active Directory, a Microsoft implementation of LDAP, is a directory service that can store information about the network resources. While Active Directory can serve many functions, the primary reason for joining the cluster to the domain is to perform user and group authentication.

You can configure and manage a cluster's Active Directory settings from the Web Administration interface or the command-line interface; however, it is recommended that you use Web Administration whenever possible.

Each node in the cluster shares the same Active Directory machine account making it very easy to administer and manage.

LDAP

The Lightweight Directory Access Protocol (LDAP) is a networking protocol used for defining, querying, and modifying services and resources. A primary advantage of LDAP is the open nature of the directory services and the ability to use LDAP across many platforms. The Isilon clustered storage system can use LDAP to authenticate users and groups in order to grant them access to the cluster.

NIS

The Network Information Service (NIS), designed by Sun Microsystems, is a directory services protocol that can be used by the Isilon system to authenticate users and groups when accessing the cluster. NIS, sometimes referred to as Yellow Pages (YP), is different from NIS+, which the Isilon cluster does not support.

Local users

The Isilon clustered storage system supports local user and group authentication. You can create local user and group accounts directly on the cluster, using the WebUI interface. Local authentication can be useful when directory services—Active Directory, LDAP, or NIS—are not used, or when a specific user or application needs to access the cluster.

Access zones

Access zones provide a method to logically partition cluster access and allocate resources to self-contained units, thereby providing a shared tenant environment. To facilitate this, Access Zones tie together the three core external access components:

- Cluster network configuration
- File protocol access
- Authentication

As such, Isilon SmartConnect^{TM} zones are associated with a set of SMB/CIFS shares and one or more authentication providers for access control. And overlapping shares are permitted, allowing the centralized management of a single home directory namespace, but provisioned and secured for multiple tenants. This is particularly useful for enterprise environments where multiple separate business units are served by a central IT department. Another example is during a server consolidation initiative, when merging multiple Windows file servers that are joined to separate, un-trusted, Active Directory forests.

With Access Zones, the built-in System access zone includes an instance of each supported authentication provider, all available SMB shares, and all available NFS exports by default.

These authentication providers can include multiple instances of Microsoft Active Directory, LDAP, NIS, and local user or group databases.

Roles Based Administration

Roles Based Administration is a cluster management roles based access control system (RBAC) that divides up the powers of the “root” and “administrator” users into more granular privileges, and allows assignment of these to specific roles. These roles can then be granted to other non-privileged users. For example, data center operations staff can be assigned read-only rights to the entire cluster, allowing full monitoring access but no configuration changes to be made. OneFS provides a collection of built-in roles, including audit, system & security administrator, plus the ability to create custom defined roles. Roles Based Administration is integrated with the OneFS command line interface, WebUI and Platform API.

Software upgrade

Upgrading to the latest version of OneFS allows you to take advantage of any new features, fixes and functionality on the Isilon Cluster. Clusters can be upgraded using two methods: Simultaneous or Rolling Upgrade

Simultaneous upgrade

A simultaneous upgrade installs the new operating system and reboots all nodes in the cluster at the same time. A simultaneous upgrade requires a temporary, sub-2 minute, interruption of service during the upgrade process while the nodes are restarted.

Rolling upgrade

A rolling upgrade individually upgrades and restarts each node in the cluster sequentially. During a rolling upgrade, the cluster remains online and continues serving data to clients with no interruption in service. A rolling upgrade can only be performed within a OneFS code version family and not between OneFS major code version revisions.

Performing the upgrade

Isilon highly recommends that, prior to an upgrade, you run a pre-install verification script. The script verifies that the configuration in your current installation of OneFS is compatible with the version of OneFS that is intended for the upgrade. A patch that contains the script is available from the Isilon Customer Support Center.

During an upgrade, a pre-installation upgrade check is also performed to ensure that only a supported configuration is permitted to upgrade. When an unsupported configuration is found, the upgrade is stopped and instructions on troubleshooting the issue are displayed. Proactively running the pre-installation upgrade check before starting an upgrade helps to avoid any interruption due to incompatible configuration.

Once the administrator has downloaded the upgrade package from the Isilon Customer Support Center, they can either run the upgrade via the CLI or through the Web Administration Interface. Verification on the upgrade can be done via either interface by checking the health status of the entire cluster once the upgrade completes successfully.

EMC Isilon data protection and management software

Isilon offers a comprehensive portfolio of data protection and management software to address your needs:

InsightIQ™	Performance Management	Maximize performance of your Isilon scale-out storage system with innovative performance monitoring and reporting tools
SmartPools™	Resource Management	Implement a highly efficient, automated tiered storage strategy to optimize storage performance and costs
SmartQuotas™	Data Management	Assign and manage quotas that seamlessly partition and thin provision storage into easily managed segments at the cluster, directory, sub-directory, user, and group levels
SmartConnect™	Data Access	Enable client connection load balancing and dynamic NFS failover and failback of client connections across storage nodes to optimize use of cluster resources
SnapshotIQ™	Data Protection	Protect data efficiently and reliably with secure, near instantaneous snapshots while incurring little to no performance overhead. Speed recovery of critical data with near-immediate on-demand snapshot restores.
Isilon for vCenter	Data Management	Manage Isilon functions from vCenter.
SyncIQ™	Data Replication	Replicate and distribute large, mission-critical data sets asynchronously to multiple shared storage systems in multiple sites for reliable disaster recovery capability. Push-button failover and failback simplicity to increase availability of mission-critical data.
SmartLock™	Data Retention	Protect your critical data against accidental, premature or malicious alteration or deletion with our software-based approach to Write Once Read Many (WORM) and meet stringent compliance and governance needs such as SEC 17a-4 requirements.
SmartDedupe™	Data Deduplication	Maximize storage efficiency by scanning the cluster for identical blocks and then eliminating the duplicates, decreasing the amount of physical storage required.

Please refer to product documentation for details on all of the above Isilon software products.

Conclusion

With OneFS, organizations and administrators can scale from 18 TB to 20 PB within a single file system, single volume, with a single point of administration. OneFS delivers high-performance, high-throughput, or both, without adding management complexity.

Next-generation data centers must be built for sustainable scalability. They will harness the power of automation, leverage the commoditization of hardware, ensure the full consumption of the network fabric, and provide maximum flexibility for organizations intent on satisfying an ever-changing set of requirements.

OneFS is the next-generation file system designed to meet these challenges. OneFS provides:

- Fully distributed single file system
- High-performance, fully symmetric cluster
- File striping across all nodes in a cluster
- Automated software to eliminate complexity
- Dynamic content balancing
- Flexible data protection
- High availability
- Web-based and command-line administration

OneFS is ideally suited for file-based and unstructured “Big Data” applications in enterprise environments – including large-scale home directories, file shares, archives, virtualization and business analytics – as well as a wide range of data-intensive, high performance computing environments including energy exploration, financial services, Internet and hosting services, business intelligence, engineering, manufacturing, media & entertainment, bioinformatics, and scientific research.

About EMC

EMC Corporation is a global leader in enabling businesses and service providers to transform their operations and deliver IT as a service. Fundamental to this transformation is cloud computing. Through innovative products and services, EMC accelerates the journey to cloud computing, helping IT departments to store, manage, protect and analyze their most valuable asset - information - in a more agile, trusted and cost-efficient way. Additional information about EMC can be found at www.EMC.com.