White Paper

NEXT GENERATION STORAGE TIERING WITH EMC ISILON SMARTPOOLS

Abstract

Most file systems are a thin layer of organization on top of a block device and cannot efficiently address data at large scale. This paper focuses on OneFS, a modern file system that meets the unique needs of Big Data. OneFS includes SmartPools, a native tiering capability, which enables enterprises to reduce storage costs without sacrificing performance or data protection.

April 2013

EMC²

Part Number H8321.1

Table of Contents

Introduction **4**

Isilon Scale-Out NAS Architecture **4**

Single File System 5
Data Layout and Protection 5
Job Engine 6
Data Rebalancing 6
Caching 7

Smart Pools **7**

Overview 7
Storage Pools 8
Disk Pools 8
Node Pools 9
Tiers 11
Node Equivalence 12
Data Spill Over 14
Automatic Provisioning 14
Global Namespace Acceleration 15
Virtual Hot Spare 16
SmartPools Licensing 17
File Pools 17
File Pool Policies 19
Custom File Attributes 23
File Pool Policy Engine 23
Data Location 24
Node Pool Affinity 25
Performance with SmartPools 25
Using SmartPools to Improve Performance 26
Data Access Settings 27
Leveraging SSDs for Metadata & Data Performance 27
Minimizing the Performance Impact of Tiered Data 28
SmartPools Best Practices 29
SmartPools Use Cases 29

SmartPools Workflow Examples **30**

Example A: Storage Cost Efficiency in Media Post Production 30
Example B: Data Availability & Protection in Semiconductor Design 31
Example C: Investment Protection for Financial Market Data 32
Example D: Metadata Performance for Seismic Interpretation 33

Conclusion **34**

About EMC Isilon **34**

Introduction

EMC Isilon SmartPools enables multiple levels of performance, protection, and storage density to co-exist within the same file system, and unlocks the ability to aggregate and consolidate a wide range of applications within a single extensible, ubiquitous storage resource pool. This helps provide granular performance optimization, workflow isolation, higher utilization, and independent scalability – all with a single point of management.

SmartPools allows you to define the value of the data within your workflows based on policies, and automatically aligns data to the appropriate price/performance tier over time. Data movement is seamless, and with file-level granularity and control via automated policies, manual control, or API interface, you can tune performance and layout, storage tier alignment, and protection settings – all with minimal impact to your end-users.

Storage tiering has a very convincing value proposition, namely segregating data according to its business value, and aligning it with the appropriate class of storage and levels of performance and protection. Information Lifecycle Management techniques have been around for a number of years, but have typically suffered from the following inefficiencies: complex to install and manage, involves changes to the file system, requires the use of stub files, etc.

EMC Isilon SmartPools is a next generation approach to tiering that facilitates the management of heterogeneous clusters. The SmartPools capability is native to the Isilon OneFS scale-out file system, which allows for unprecedented flexibility, granularity, and ease of management. In order to achieve this, SmartPools leverages many of the components and attributes of OneFS, including data layout and mobility, protection, performance, scheduling, and impact management.

Isilon Scale-Out NAS Architecture

In Isilon scale-out Network Attached Storage (NAS), 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 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 stores and manages data in the Isilon storage cluster.

Single File System

OneFS provides a single file system that operates outside the constraints of traditional scale up constructs like RAID groups, allowing data to be placed anywhere on the cluster, thereby accommodating a variety of levels of performance and protection.

Each cluster operates within a single volume and namespace with the file system distributed across all nodes. As such, there is no partitioning, and clients are provided a coherent view of their data from any node in the cluster.

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.

From an application or user perspective, all capacity is available - the storage has been completely virtualized for the users and administrator. The file system can grow organically without requiring too much planning or oversight. No special thought has to be applied by the administrator about tiering files to the appropriate disk, because SmartPools will handle that automatically. Additionally, no special consideration needs to be given to how one might replicate such a large file system, because the Isilon SyncIQ service automatically parallelizes the transfer of the data to one or more alternate clusters.

Data Layout and Protection

OneFS is designed to withstand multiple simultaneous component failures (currently four per node pool) while still affording unfettered access to the entire file system and dataset. Data protection is implemented at the file level using Reed Solomon erasure coding and, as such, is not dependent on any hardware RAID controllers.

Inodes, directories, and other metadata are protected at the same or higher level as the data blocks they reference. Since all data, metadata, and forward error correction (FEC) blocks are striped across multiple nodes, there is no requirement for dedicated parity drives. This guards against single points of failure and bottlenecks, allows file reconstruction to be highly parallelized, and ensures that all hardware components in a cluster are always in play doing useful work.

OneFS also provides a variety of mirroring options ranging from 2x to 8x, allowing from two to eight mirrors of the specified content. This is the method used for protecting OneFS metadata.

Additionally, a logical separation of data and metadata structures within the single OneFS filesystem allows SmartPools to manage data and metadata objects separately, as we will see.

OneFS stores file and directory metadata in inodes and B-trees, allowing the file system to scale to billions of objects and still provide very fast lookups of data or metadata. OneFS is a completely symmetric and fully distributed file system with data and metadata spread across multiple hardware devices. Data are generally protected using erasure coding for space efficiency reasons, enabling utilization levels of up to 80% and above on clusters of five nodes or more. Metadata (which generally makes up around 2% of the system) is mirrored for performance and availability. Protection

levels are dynamically configurable at a per-file or per-file system granularity, or anything in between. Data and metadata access and locking are coherent across the cluster, and this symmetry is fundamental to the simplicity and resiliency of OneFS' shared nothing architecture.

When a client connects to a OneFS-managed node and performs a write operation, files are broken into smaller logical chunks, or stripes units, before being written to disk. These chunks are then striped across the cluster's nodes and protected either via erasure-coding or mirroring. OneFS primarily uses the Reed-Solomon erasure coding system for data protection, and mirroring for metadata. Isilon's file level protection typically provides industry leading levels of utilization. And, for nine node and larger clusters, OneFS is able to sustain up to four full node failures while still providing full access to data.

OneFS uses multiple data layout methods to optimize for maximum efficiency and performance according to the data's access pattern – for example, streaming, concurrency, random, etc. And, like protection, these performance attributes can also be applied per file or per file system.

Job Engine

The Job Engine is OneFS' parallel task scheduling framework, and is responsible for the distribution, execution, and impact management of critical jobs and operations across the entire cluster.

The OneFS Job Engine schedules and manages all the data protection and background cluster tasks: creating jobs for each task, prioritizing them and ensuring that inter-node communication and cluster wide capacity utilization and performance are balanced and optimized. Job Engine ensures that core cluster functions have priority over less important work and gives applications integrated with OneFS – Isilon add-on software or applications integrating to OneFS via the OneFS API – the ability to control the priority of their various functions to ensure the best resource utilization.

Each job, for example the SmartPools job, has an "Impact Profile" comprising a configurable policy and a schedule which characterizes how much of the system's resources the job will take, plus an Impact Policy and an Impact Schedule. The amount of work a job has to do is fixed, but the resources dedicated to that work can be tuned to minimize the impact to other cluster functions, like serving client data.

Data Rebalancing

Another key Job Engine task is AutoBalance. This enables OneFS to reallocate and rebalance data across the nodes in a cluster, making storage space utilization more uniform and efficient.

OneFS manages protection of file data directly, and when a drive or entire node failure occurs, it rebuilds data in a parallel fashion. OneFS avoids the requirement for dedicated hot spare drives and serial drive rebuilds, and simply borrows from the available free space in the file system in order to recover from failures; this technique is called virtual hot spare. This approach allows the cluster to be self-healing, without human intervention, and with the advantages of fast, parallel data reconstruction. The administrator can create a virtual hot spare reserve, which prevents users from consuming capacity that is reserved for the virtual hot spare.

The process of choosing a new layout is called restriping, and this mechanism is identical for repair, rebalance, and tiering. Data is moved in the background with the file available at all times, a process that's completely transparent to end users and applications.

Caching

SmartCache is a globally coherent read and write caching infrastructure that provides low latency access to content. Like other resources in the cluster, as more nodes are added, the total cluster cache grows in size, enabling Isilon to deliver predictable, scalable performance within a single file system.

An Isilon cluster provides a high cache to disk ratio (multiple GB per node), which is dynamically allocated for read operations as needed. This cache is unified and coherent across all nodes in the cluster, allowing a user on one node to benefit from I/O already transacted on another node. OneFS stores only *distinct* data on each node. The node's RAM is used as a "level 2" (L2) cache of such data. These distinct, cached blocks can be accessed across the Infiniband backplane very quickly and, as the cluster grows, 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 sequentially accessed data, OneFS SmartRead aggressively pre-fetches data, greatly improving read performance across all protocols.

Write caching uses write buffering to aggregate, or coalesce, multiple write operations to the NVRAM file systems journals so that they can be written to disk safely and more efficiently. This form of buffering reduces the disk write penalty which could require multiple reads and writes for each write operation.

Smart Pools

Overview

SmartPools is a next-generation data tiering product that builds directly on the core OneFS attributes described above. These include the single file system, extensible data layout and protection framework, parallel job execution, and intelligent caching architecture.

SmartPools was originally envisioned as a combination of two fundamental notions:

- The ability to define sub-sets of hardware within a single, heterogeneous cluster.
- A method to associate logical groupings of files with these hardware sub-sets via simple, logical definitions or rules.

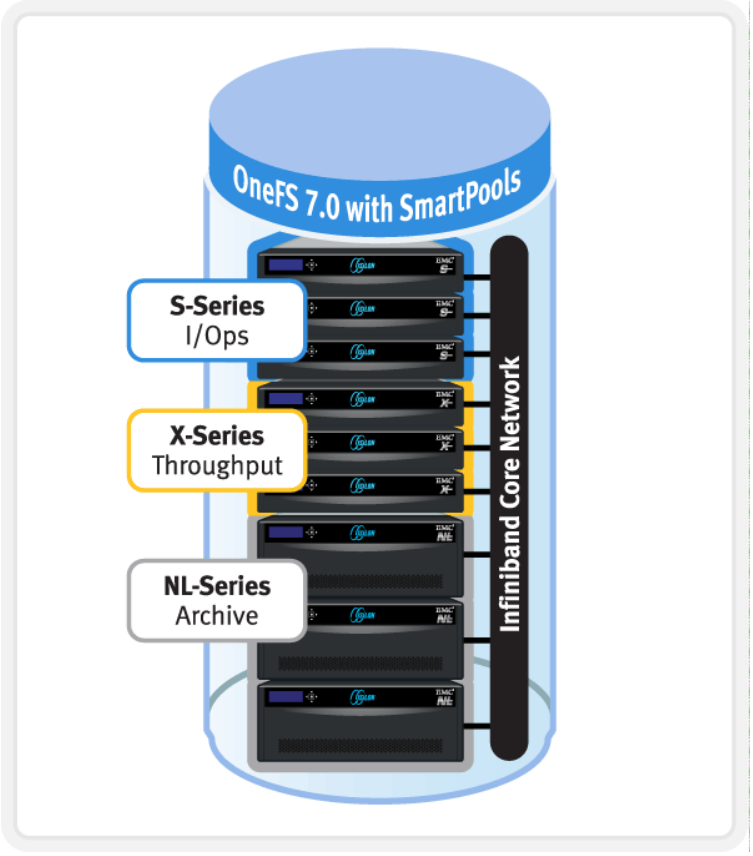


Figure 2. SmartPools Tiering Model

The current SmartPools implementation expands on these two core concepts in several additional ways, while maintaining simplicity and ease of use as primary goals.

Storage Pools

Storage Pools provide the ability to define subsets of hardware within a single cluster, allowing file layout to be aligned with specific sets of nodes through the configuration of storage pool policies. The notion of Storage Pools is an abstraction that encompasses disk pools, node pools, and tiers, all described below.

Disk Pools

Disk Pools are the smallest unit within the Storage Pools hierarchy, as illustrated in figure 3 below. OneFS provisioning works on the premise of dividing similar nodes' drives into sets, or disk pools, with each pool representing a separate failure domain.

These disk pools are protected by default at N+2:1 (or the ability to withstand two disk or one node failure) and typically comprise six drives per node and span from three to forty nodes. Each drive may only belong to one disk pool and data protection stripes or mirrors don't extend across disk pools (the exception being a Global Namespace Acceleration extra mirror, described below). Disk pools are managed by OneFS and are not user configurable.

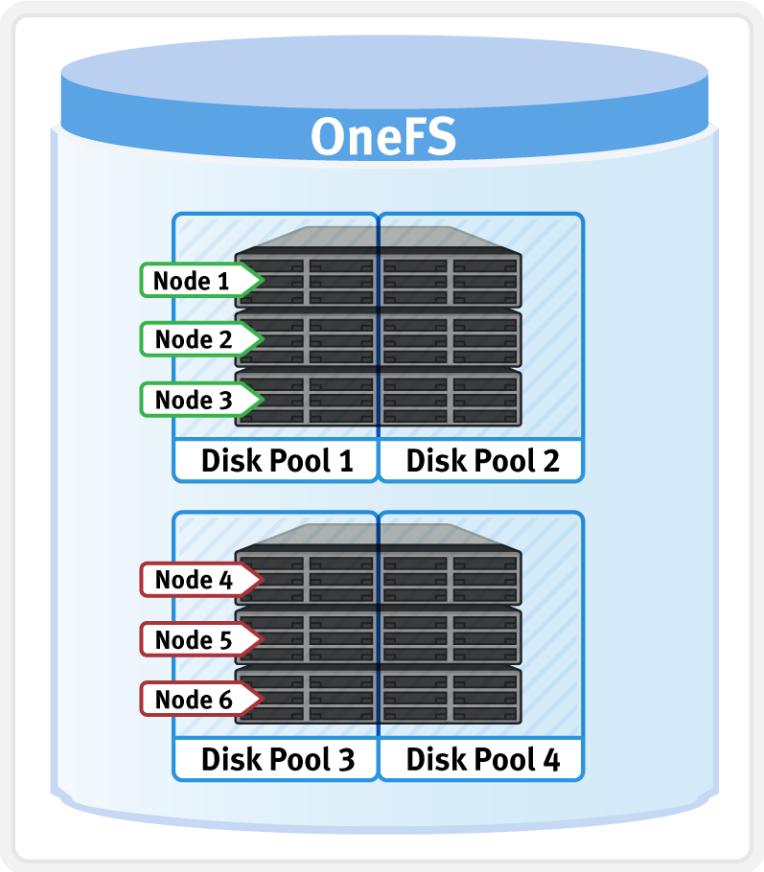


Figure 3. Cluster Disk Pool Allocation

Node Pools

Node Pools are groups of Disk Pools, spread across similar Isilon storage nodes (equivalence classes). This is illustrated in figure 4, below. Multiple groups of different node types can work together in a single, heterogeneous cluster. For example: one Node Pool of S Series nodes typically used for IOPS-intensive applications, one Node Pool of X Series nodes, primarily used for high-concurrent and sequential workloads, and one Node Pool of NL Series nodes, primarily used for archive purposes.

This allows OneFS to present a single storage resource pool comprising multiple drive media types – SSD, high speed SAS, large capacity SATA, etc - providing a range of different performance, protection and capacity characteristics. This heterogeneous storage pool in turn can support a diverse range of applications and workload requirements with a single, unified point of management. It also facilitates the mixing of older and newer hardware, allowing for simple investment protection even across product generations, and seamless hardware refreshes.

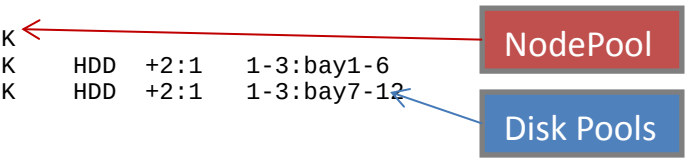
Each Node Pool only contains disk pools from the same type of storage nodes and a disk pool may belong to exactly one node pool. For example, S200 Series nodes with 300 GB SAS drives and one 400 GB SSD per node would be in one node pool, whereas NL400 Series with 3 TB SATA Drives would be in another. Today, a minimum of 3 nodes are required per Node Pool. Nodes are not provisioned (not associated with each other and not writable) until at least three nodes from the same equivalence

class are assigned in a node pool. If nodes are removed from a Node Pool, that pool becomes under-provisioned. In this situation, if two like nodes remain, they are still writable. If only one remains, it is automatically set to read-only.

Once node pools are created, they can be easily modified to adapt to changing requirements. Individual nodes can be reassigned from one node pool to another. Node Pool associations can also be discarded, releasing member nodes so they can be added to new or existing pools. Node Pools can also be renamed at any time without changing any other settings in the Node Pool configuration.

Any new node added to a cluster is automatically allocated to a Node Pool and then subdivided into Disk Pools without any additional configuration steps, inheriting the SmartPools configuration properties of that Node Pool. This means the configuration of Disk Pool data protection, layout and cache settings only needs to be completed once per node pool and can be done at the time the node pool is first created. Automatic allocation is determined by the shared attributes of the new nodes with the closest matching Node Pool (an S node with 600 GB SAS Drives joins a Disk Pool of S Nodes with 600 GB drives). If the new node is not a close match to the nodes of any existing Node Pool, it remains un-provisioned until the minimum Node Pool node membership for like nodes is met (three nodes of same or similar storage and memory configuration).

```
# isi smartpools health -v
SmartPools Health
Name                  Health  Type Prot   Members          Down
Smartfailed
--------------------- ------- ---- ------ ---------------- --------- --------
---
x200_36tb_6gb           OK
 x200_36tb_6gb:1        OK    HDD  +2:1   1-3:bay1-6
 x200_36tb_6gb:3        OK    HDD  +2:1   1-3:bay7-12
```



When a new Node Pool is created and nodes are added SmartPools associates those nodes with an ID. That ID is also used in File Pool policies and file attributes to dictate file placement within a specific Disk Pool.

By default, a file which is not covered by a specific File Pool policy will go to the default Node Pool or pools identified during set up. If no default is specified, SmartPools will write that data to the pool with the most available capacity.

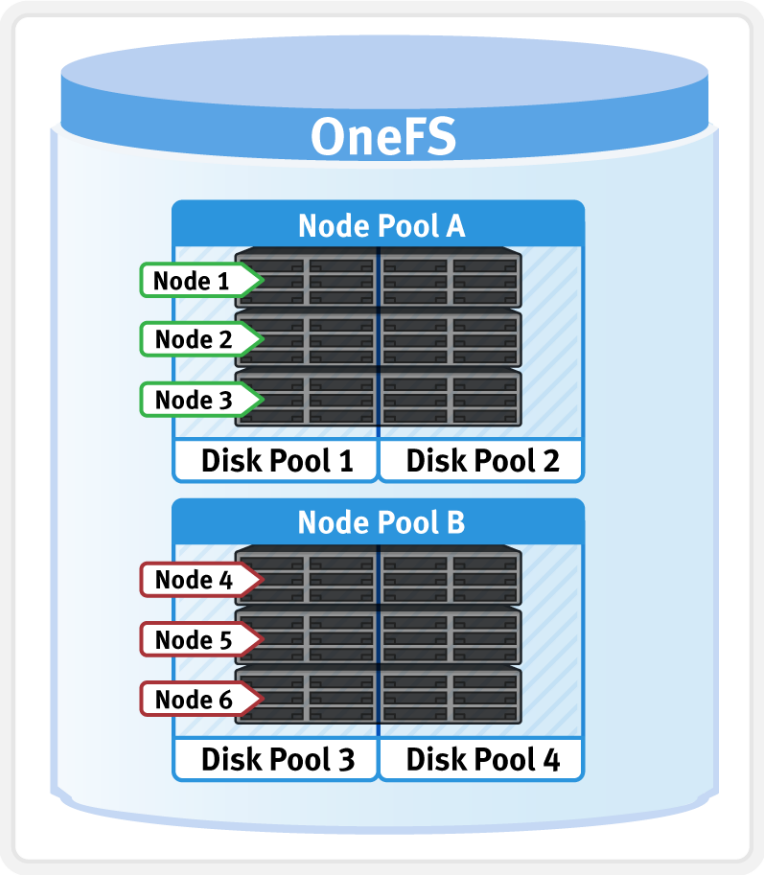


Figure 4. Automatic Provisioning into Node Pools

Tiers

Tiers are groups of node pools combined into a logical superset to optimize data storage, according to OneFS platform type. This is illustrated in figure 6, below. For example, X Series node pools are often combined into a single tier. This tier could incorporate different styles of NL Series node pools (i.e. NL400 with 1TB SATA drives and NL400 with 3TB SATA drives) into a single, logical container. This is a significant benefit because it allows customers who consistently purchase the highest capacity nodes available to consolidate a variety of node styles within a single group, or tier, and manage them as one logical group.

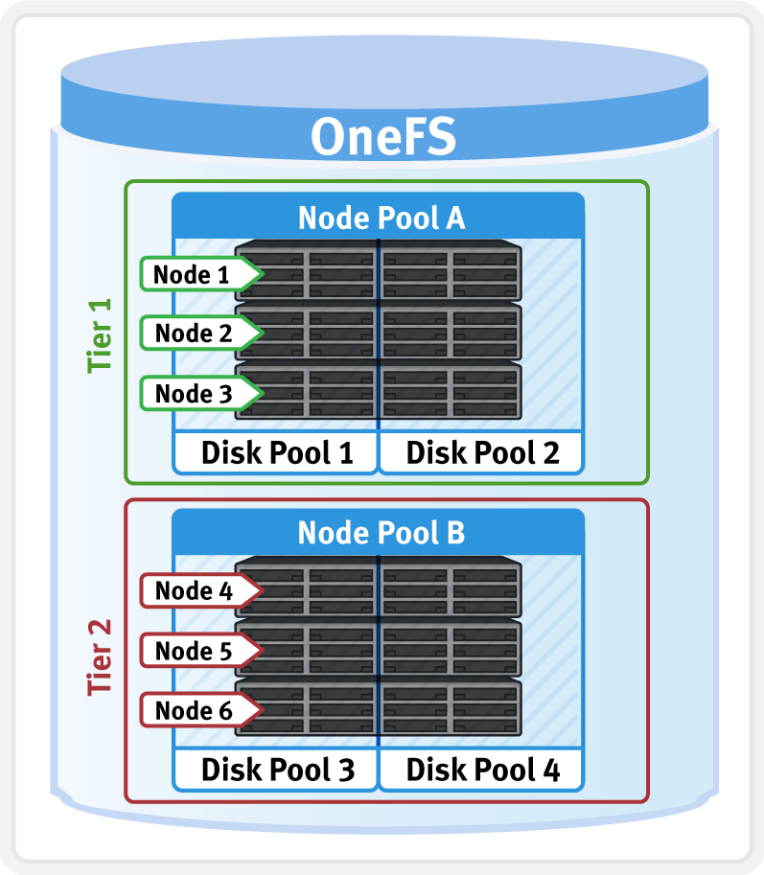


Figure 5. SmartPools Tier Configuration

SmartPools users typically deploy 2 to 4 tiers, with the fastest tier usually containing SSD nodes for the most performance demanding portions of a workflow, and the lowest, capacity-biased tier comprising multi-TB SATA drive nodes.

Tiers & Node Pools	Status	Nodes	Protection	HDD Used	HDD Size	HDD % Used	SSD Used	SSD Size	SSD % Used	
Cluster: wopr										
Tier1										Edit
iq_10000x-ssd		1-5	+2:1	4.89 GB	43.1 TB	0.01%	3.45 GB	913 GB	0.38%	Edit
iq_5000s-ssd		6-11	+2:1	4.13 GB	25.8 TB	0.02%	2.43 GB	548 GB	0.44%	Edit
s200_7.2tb_24gb		80-89, 97	+2:1	15.1 GB	70.4 TB	0.02%	--	--	--	Edit
Tier2										Edit

Figure 6. SmartPools WebUI View – Tiers & Node Pools

Node Equivalence

SmartPools allows nodes of any type supported by the particular OneFS version to be combined within the same cluster. The like-nodes will be provisioned into different node pools according to their physical attributes and these node equivalence classes are fairly stringent. This is in order to avoid disproportionate amounts of work being

directed towards a subset of cluster resources. This is often referred to as “node bullying” and can manifest as severe over-utilization of resources like CPU, network or disks.

However, administrators can safely target specific data to broader classes of storage by creating tiers. For example, if a cluster includes two different varieties of X nodes, these will automatically be provisioned into two different node pools. These two node pools can be logically combined into a tier, and file placement targeted to it, resulting in automatic balancing across the node pools.

Note:

1. A minimum of three nodes of a particular type must be added to the cluster before a node pool is provisioned.
2. If three or more nodes have been added, but fewer than three nodes are accessible, the cluster will be in a degraded state. The cluster may or may not be usable, depending on whether there is still cluster quorum (greater than half the total nodes available).
3. All nodes in the cluster must have a current, valid support contract.

SmartPools segregates hardware by node type and creates a separate node pool for each distinct hardware variant. In order to reside in the same node pool, nodes must have a set of core attributes in common:

1. Family (i.e. X-series)
2. Chassis size (i.e. 4 rack units)
3. Generation (200-series)
4. RAM capacity (i.e. 24GB)
5. Drive capacity (HDD and SSD capacity and quantity)

However, in order to provide investment protection and support incremental node addition between older and newer node styles, equivalence classes are maintained. These include:

Isilon Hundred-series Node	Equivalent IQ-series Node
X200 with 6GB RAM and 12 500GB HDDs	IQ 6000x
X200 with 6GB RAM and 12 1TB HDDs	IQ 12000x
X400 with 24GB RAM and 36 1TB HDDs	IQ 36000x
X400 with 24GB RAM and 36 2TB HDDs	IQ 72000x
X400 with 24GB RAM , 32 1TB HDDs & 4 100GB SSDs	IQ 32000x-SSD
NL400 with 12GB RAM and 36 1TB HDDs	IQ 36NL
NL400 with 12GB RAM and 36 2TB HDDs	IQ 72NL
NL400 with 12GB RAM and 36 3TB HDDs	IQ 108NL

Table 1. Isilon Node Equivalence

Data Spill Over

If a Node Pool fills up, writes to that pool will automatically spill over to the next pool. This default behavior ensures that work can continue even if one type of capacity is full. There are some circumstances in which spillover is undesirable, for example when different business units within an organization purchase separate pools, or data location has security or protection implications. In these circumstances, spillover can simply be disabled. Disabling spillover ensures a file exists in one pool and will not move to another. Keep in mind that reservations for virtual hot sparing will affect spillover – if, for example, VHS is configured to reserve 10% of a pool's capacity, spillover will occur at 90% full.

Protection settings can be configured outside SmartPools and managed at the cluster level, or within SmartPools at either the Node Pool or File Pool level. Wherever protection levels exist, they are fully configurable and the default protection setting for a Node Pool is N+2:1.

Automatic Provisioning

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 efficiently protected disk pools. For example, a forty node nearline cluster with 3TB SATA disks would typically run at a +4 protection level. However, partitioning it into two, twenty node disk pools would allow each pool to run at +2 protection, 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 divide the cluster into pools of disks, which are optimized for both Mean Time to Data Loss (MTTDL) and efficient space utilization. This means that protection

level decisions, such as the forty node cluster example above, are not left to the customer.

With Automatic Provisioning, every set of equivalent node hardware is automatically divided into disk pools comprising up to forty nodes and six drives per node. These disk pools are protected against up to two drive failures per disk pool. Multiple similar disk pools are automatically combined into node pools, which can then be further aggregated into logical tiers and managed with SmartPools file pool policies. By subdividing a node's disks into multiple, separately protected disk pools, nodes are significantly more resilient to multiple disk failures than previously possible.

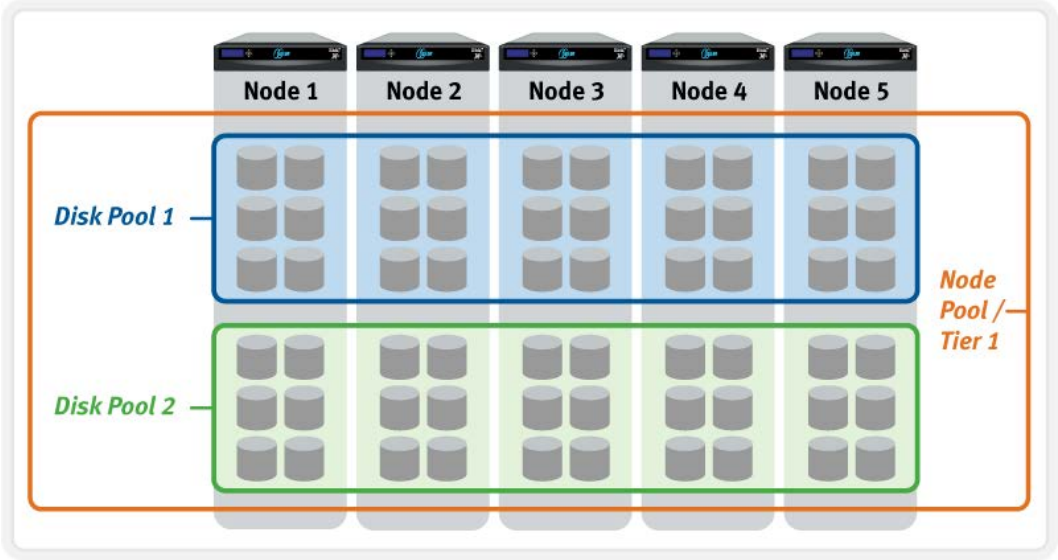


Figure 7. Automatic Provisioning with SmartPools

When initially configuring a cluster, OneFS automatically assigns nodes to node pools. Nodes are not pooled—not associated with each other—until at least three nodes are assigned to the node pool. A node pool with fewer than three nodes is considered an under-provisioned pool.

Global Namespace Acceleration

Global Name Space Acceleration, or GNA, is an unlicensed, configurable component of SmartPools. GNA's principal goal is to help accelerate metadata read operations (eg. filename lookups) by keeping a copy of the cluster's metadata on high performance, low latency SSD media. This allows customers to increase the performance of certain workloads across the whole file system without having to upgrade/purchase SSDs for every node in the cluster. This is illustrated in figure 8, below. To achieve this, an extra mirror of metadata from storage pools that do not contain SSDs is stored on SSDs available anywhere else in the cluster, regardless of node pool boundaries. As such, metadata read operations are accelerated even for data on node pools that have no SSDs.

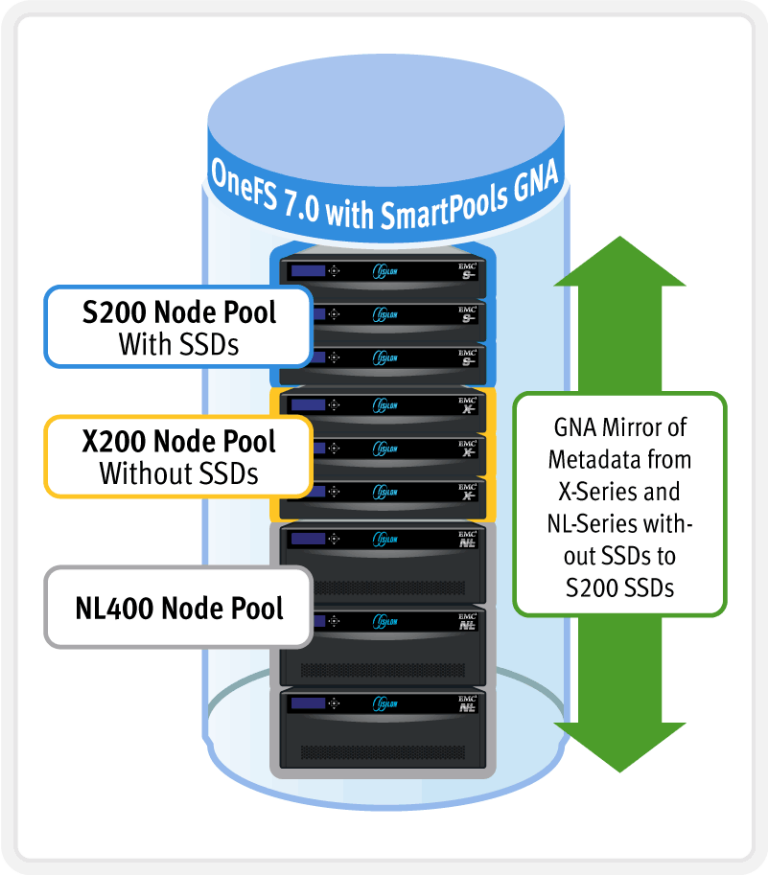


Figure 8. Global Name Space Acceleration

The purpose of GNA is to accelerate the performance of metadata-intensive applications and workloads such as home directories, workflows with heavy enumeration and activities requiring a large number of comparisons. Examples of metadata-read-heavy workflows exist across the majority of Isilon's established and emerging markets. In some, like EDA for example, such workloads are dominant and the use of SSDs to provide the performance they require is ubiquitous.

Virtual Hot Spare

SmartPools Virtual Hot Spare (VHS) helps ensure that node pools maintain enough free space to successfully re-protect data in the event of drive failure. Though configured globally, VHS actually operates at the node pool level so that nodes with different size drives reserve the appropriate VHS space. This helps ensures that, while data may move from one disk pool to another during repair, it remains on the same class of storage. VHS reservations are cluster wide and configurable as either a percentage of total storage (0-20%) or as a number of virtual drives (1-4). The mechanism of this reservation is to allocate a fraction of the node pool's VHS space in each of the node pool's constituent disk pools.

No space is reserved for VHS on SSDs unless the entire node pool consists of SSDs. This means that a failed SSD may have data moved to HDDs during repair, but without adding additional configuration settings, the alternative is reserving an unreasonable percentage of the SSD space in a node pool.

The default for new clusters is for Virtual Hot Spare to have both "reduce amount of available space" and "deny new data writes" enabled with one virtual drive. On upgrade, existing settings are maintained. All customers are encouraged to enable Virtual Hot Spare.

SmartPools Licensing

The base SmartPools license enables three things:

1. The ability to manually set individual files to a specific storage pools.
2. All aspects of file pool policy configuration.
3. Running the SmartPools job.

The SmartPools job runs on a schedule and applies changes that have occurred through file pool policy configurations, file system activity or just the passage of time. The schedule is configurable via standard Job Engine means and defaults to daily at 10pm.

The storage pool configuration of SmartPools requires no license. Drives will automatically be provisioned into disk pools and node pools. Tiers can be created, but have little utility since files will be evenly allocated among the disk pools. The "global smartpools settings" are all still available, with the caveat that spillover is forcibly enabled to "any" in the unlicensed case. Spillover has little meaning when any file can be stored anywhere.

The default file pool policy will apply to all files. This can be used to set the protection policy and I/O optimization settings, but the disk pool policy cannot be changed. The SetProtectPlus job will run to enforce these settings when changes are made.

In the case that a SmartPools license lapses, the disk pool policy set on files' inodes will be ignored and it will be treated as the "ANY" disk pool policy. However, since the disk pool policy is only evaluated when selecting new disk pool targets, writes to files with valid targets will continue to be directed to those targets based on the old disk pool policy until a reprotect or higher level restripe causes the disk pool targets to be reevaluated. New files will inherit the disk pool policy of their parents (possibly through the New File Attributes mechanism), but that disk pool policy will be ignored and their disk pool targets will be selected according to the "ANY" policy.

When SmartPools is not licensed, any disk pool policy is ignored. Instead, the policy is considered to include all disk pools and, as such, file data is directed to and balanced across all pools.

File Pools

This is the SmartPools logic layer, where user configurable File Pool policies govern where data is placed, protected, accessed, and how it moves among the Node Pools and Tiers. This is conceptually similar to storage ILM (information lifecycle management), but does not involve file stubbing or other file system modifications. File Pools allow data to be automatically moved from one type of storage to another within a single cluster to meet performance, space, cost or other requirements, while retaining its data protection settings. For example a File Pool policy may dictate anything written to path /ifs/foo goes to the S Series nodes in Node Pool 1, then moves to the NL Series nodes in Node Pool 3 when older than 30 days.

To simplify management, there are defaults in place for Node Pool and File Pool settings which handle basic data placement, movement, protection and performance. All of these can also be configured via the simple and intuitive UI, delivering deep granularity of control. Also provided are customizable template policies which are optimized for archiving, extra protection, performance and VMware files.

When a SmartPools job runs, the data may be moved, undergo a protection or layout change, etc. There are no stubs. The file system itself is doing the work so no transparency or data access risks apply.

Data movement is parallelized with the resources of multiple nodes being leveraged for speedy job completion. While a job is in progress all data is completely available to users and applications.

The performance of different nodes can also be augmented with the addition of system cache or Solid State Drives (SSDs). An Isilon scale-out NAS cluster can leverage over 14 TB of globally coherent cache. Most nodes can be augmented with up to twenty-four SSD drives.

Within a File Pool, SSD 'Strategies' can be configured to place a copy of that pool's metadata, or even some of its data, on SSDs in that pool. When Global Namespace Acceleration (GNA) is enabled, a copy of all metadata for the entire cluster is kept on SSD, so access to all data on the cluster – even data on nodes which have no SSDs in them – is accelerated. For data that never needs acceleration, the correct SSD strategy is to avoid SSDs.

Overall system performance impact can be configured to suit the peaks and lulls of an environment's workload. Change the time or frequency of any SmartPools job and the amount of resources allocated to SmartPools. For extremely high-utilization environments, a sample File Pool policy can be used to match SmartPools run times to non-peak computing hours. While resources required to execute SmartPools jobs are low and the defaults work for the vast majority of environments, that extra control can be beneficial when system resources are heavily utilized.

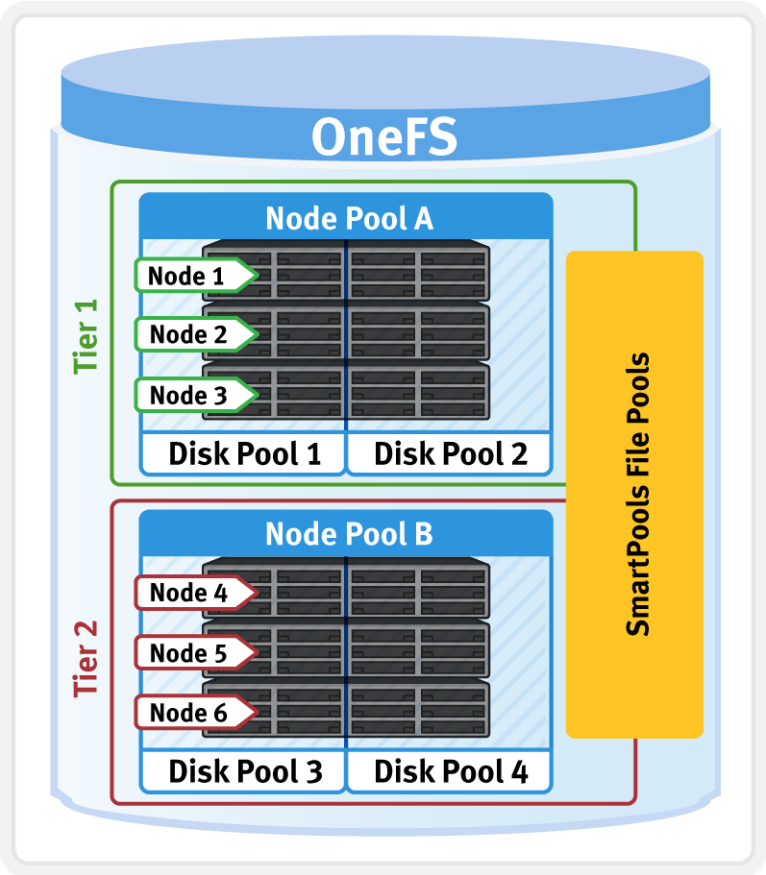


Figure 9. SmartPools File Pool Policy Engine

File Pool Policies

SmartPools file pool policies can be used to broadly control the three principal attributes of a file. Namely:

1. Where a file resides.
    1. Tier
    2. Node Pool
2. The file performance profile (I/O optimization setting).
    1. Sequential
    2. Concurrent
    3. Random
    4. SmartCache write caching
3. The protection level of a file.
    1. Parity protected (N+1-4, N+2:1, etc)
    2. Mirrored (2x – 8x)

A file pool policy is built on a file attribute the policy can match on. The attributes a file Pool policy can use are any of: File Name, Path, File Type, File Size, Modified Time, Create Time, Metadata Change Time, Access Time or User Attributes.

Once the file attribute is set to select the appropriate files, the action to be taken on those files can be added – for example: if the attribute is File Size, additional settings are available to dictate thresholds (all files bigger than… smaller than…). Next, actions are applied: move to Node Pool x, set to y protection level and lay out for z access setting.

File Attribute	Description
File Name	Specifies file criteria based on the file name
Path	Specifies file criteria based on where the file is stored
File Type	Specifies file criteria based on the file-system object type
File Size	Specifies file criteria based on the file size
Modified Time	Specifies file criteria based on when the file was last modified
Create Time	Specifies file criteria based on when the file was created
Metadata Change Time	Specifies file criteria based on when the file metadata was last modified
Access Time	Specifies file criteria based on when the file was last accessed
User Attributes	Specifies file criteria based on custom attributes – see below

Table 2. OneFS File Pool File Attributes

'And' and 'Or' operators allow for the combination of criteria within a single policy for extremely granular data manipulation.

File Pool Policies that dictate placement of data based on its path force data to the correct disk on write directly to that Node Pool without a SmartPools job running. File Pool Policies that dictate placement of data on other attributes besides path name get written to Disk Pool with the highest available capacity and then moved, if necessary to match a File Pool policy, when the next SmartPools job runs. This ensures that write performance is not sacrificed for initial data placement.

As mentioned above in the Node Pools section, any data not covered by a File Pool policy is moved to a tier that can be selected as a default for exactly this purpose. If no Disk Pool has been selected for this purpose, SmartPools will default to the Node Pool with the most available capacity.

In practice, default File Pool policies are almost always used because they can be very powerful. Most administrators do not want to set rules to govern all their data. They are generally concerned about some or most of their data in terms of where it sits and how accessible it is, but there is always data for which location in the cluster is going to be less important. For this data, there is the default policy, which is used for files for which none of the other policies in the list have applied. Typically, it is set to optimize cost and to avoid using storage pools that are specifically needed for other data. For example, most default policies are at a lower protection level, and use only the least expensive tier of storage.

When a File Pool policy is created, SmartPools stores it in the Isilon OneFS configuration database with any other SmartPools policies. When a SmartPools job runs, it runs all the policies in order. If a file matches multiple policies, SmartPools will apply only the first rule it fits. So, for example if there is a rule that moves all jpg files to a nearline Node Pool, and another that moves all files under 2 MB to a performance tier, if the jpg rule appears first in the list, then jpg files under 2 MB will go to nearline, NOT the performance tier. As mentioned above, criteria can be combined within a single policy using 'And' or 'Or' so that data can be classified very granularly. Using this example, if the desired behavior is to have all jpg files over 2 MB to be moved to nearline, the File Pool policy can be simply constructed with an 'And' operator to cover precisely that condition.

Policy order, and policies themselves, can be easily changed at any time. Specifically, policies can be added, deleted, edited, copied and re-ordered.

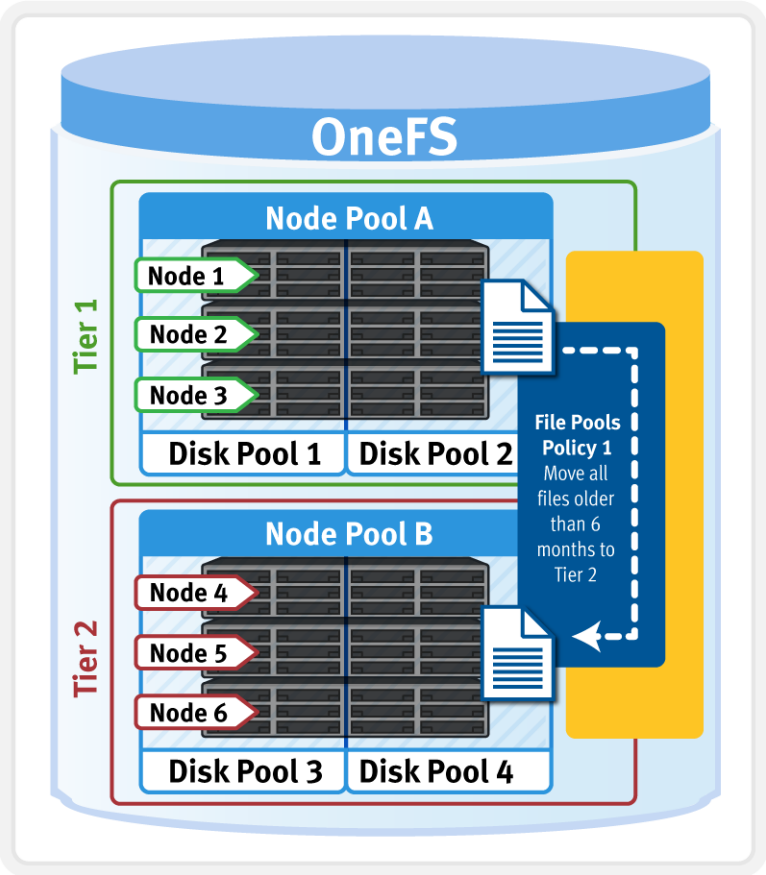


Figure 10. SmartPools File Pool Policy governing data movement

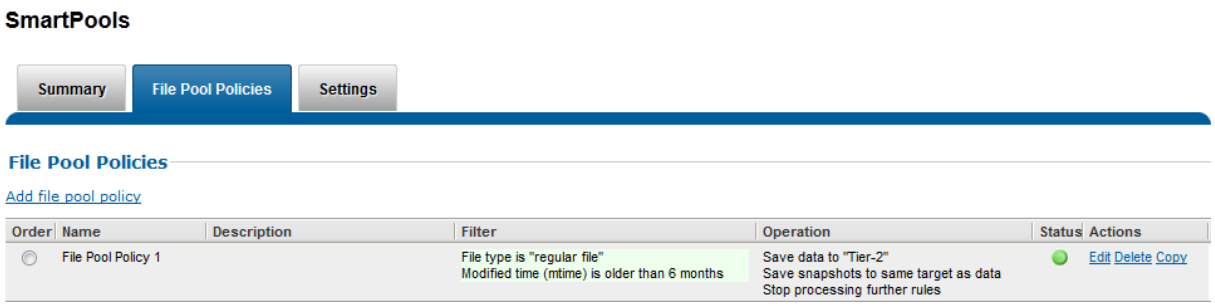


Figure 11. File Pool Policy Example

Figures 10 and 11 illustrate a common File Pools use case: An organization and wants the active data on their performance nodes in Tier 1 (SAS + SSD) and to move any data not accessed for 6 months to the cost optimized archive Tier 2.

As the list of File Pool policies grows (SmartPools currently supports up to 128 policies), it becomes less practical to manually walk through all of them to see how a file will behave when policies are applied. SmartPools also has some advanced options which are available on the command line for this kind of scenario testing and trouble shooting.

File Pool policies can be created, copied, modified, prioritized or removed at any time. Sample policies are also provided that can be used as is or as templates for customization.

Custom File Attributes

Custom File Attributes, or user attributes, can be used when more granular control is needed than can be achieved using the standard file attributes options (File Name, Path, File Type, File Size, Modified Time, Create Time, Metadata Change Time, Access Time). User Attributes use key value pairs to tag files with additional identifying criteria which SmartPools can then use to apply File Pool policies. While SmartPools has no utility to set file attributes this can be done easily by using the "`setextattr`" command.

Custom File Attributes are generally used to designate ownership or create project affinities. For example, a biosciences user expecting to access a large number of genomic sequencing files as soon as personnel arrive at the lab in the morning might use Custom File Attributes to ensure these are migrated to the fastest available storage.

Once set, Custom File Attributes are leveraged by SmartPools just as File Name, File Type or any other file attribute to specify location, protection and performance access for a matching group of files. Unlike other SmartPools File Pool policies, Custom File Attributes can be set from the command line or platform API.

Anatomy of a SmartPools JobWhen a SmartPools job runs, SmartPools examines all file attributes and checks them against the list of SmartPools policies. To maximize efficiency, wherever possible, the SmartPools' job utilizes an efficient, parallel metadata scan (LIN tree scan) instead of a more costly directory tree-walk. This is even more efficient when the SmartPools metadata on SSD strategy is deployed.

A SmartPools LIN tree scan breaks up the metadata into ranges for each node to work on in parallel. Each node can then dedicate a single or multiple threads to execute the scan on their assigned range. A LIN tree walk also ensures each file is opened only once, which is much more efficient when compared to a directory walk where hard links and other constructs can result in single threading, multiple opens, etc.

When the SmartPools file pool policy engine finds a match between a file and a policy, it stops processing policies for that file, since the first policy match determines what will happen to that file. Next, SmartPools checks the file's current settings against those the policy would assign to identify those which do not match. Once SmartPools has the complete list of settings that need to apply to that file, it sets them all simultaneously, and moves to restripe that file to reflect any and all changes to Node Pool, protection, SmartCache use, layout, etc.

File Pool Policy Engine

The SmartPools File Pool policy engine falls under the control and management of the Job Engine. The default schedule for this process is every day at 10pm, and with a low impact policy. The schedule, priority and impact policy can be manually configured and tailored to a particular environment and workload.

The engine can also be run on-demand using a separate invocation to apply the appropriate file-pool membership settings to an individual file or subdirectory without having to wait for the background scan to do it.

To test how a new policy will affect file dispositions, a SmartPools job can be run on a subset of the data. This can be either a single file or directory or group of files or directories. The job can either be run live, to actually make the policy changes, or in a 'dry-run' mode to estimate the scope and effect of a policy. This means the end state can be simulated, showing how each file would be affected by the set of File Pool Policies in place.

Running a SmartPools job against a directory or group of directories is available as a command line option. The following CLI syntax will run the engine on-demand for specific files or subtrees:

```
isi smartpools apply [-r] <filename>
```

For a dry-run assessment that calculates and reports without making changes use:

```
isi smartpools apply –nv [PATH]
```

For a particular file pool policy, the following information will be returned:

1. Policy Number
2. Files matched
3. Directories matched
4. ADS containers matched
5. ADS streams matched
6. Access changes skipped
7. Protection changes skipped
8. File creation templates matched
9. File data placed on HDDs
10. File data placed on SSDs

Note: When using the CLI utility above, SmartPools will traverse the specified directory tree, as opposed to performing a LIN scan, since the operation has been specified at the directory level. Also, the command is synchronous and, as such, will wait for completion before returning the command prompt.

Data Location

The file system explorer provides a detailed view of where SmartPools-managed data is at any time by both the actual Node Pool location and the File Pool policy-dictated location (i.e. where that file will move after the next successful completion of the SmartPools job).

When data is written to the cluster, SmartPools writes it to a single Node Pool only. This means that, in almost all cases, a file exists in its entirety within a Node Pool, and not across Node Pools. SmartPools determines which pool to write to based upon one of two situations: If a file matches a file pool policy based on directory path, that file will be written into the Node Pool dictated by the File Pool policy immediately. If

that file matches a file pool policy which is based on any other criteria besides path name, SmartPools will write that file to the Node Pool with the most available capacity. If the file matches a file pool policy that places it on a different Node Pool than the highest capacity Node Pool, it will be moved when the next scheduled SmartPools job runs.

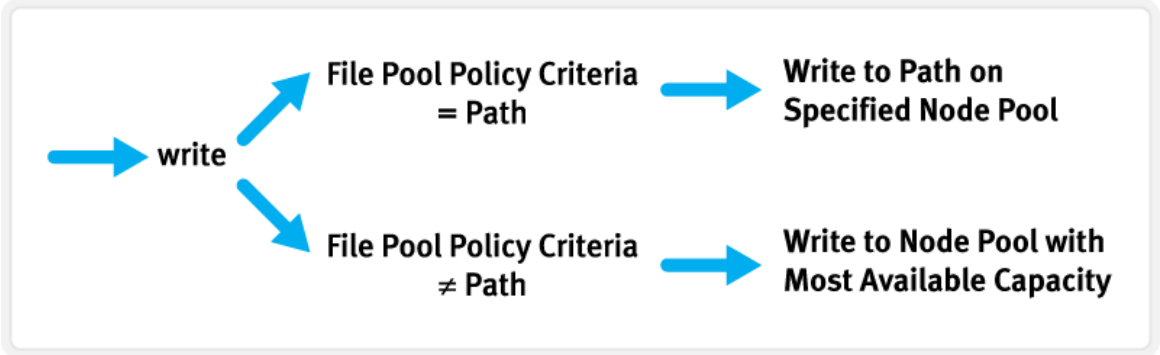


Figure 12. File Pools and Node Pools

For performance, charge back, ownership or security purposes it is sometimes important to know exactly where a specific file or group of files is on disk at any given time. While any file in a SmartPools environment typically exists entirely in one Storage Pool, there are exceptions when a single file may be split (usually only on a temporary basis) across two or more Node Pools at one time.

Node Pool Affinity

SmartPools generally only allows a file to reside in one Node Pool. A file may temporarily span several Node Pools in some situations. When a file Pool policy dictates a file move from one Node Pool to another, that file will exist partially on the source Node Pool and partially on the Destination Node Pool until the move is complete. If the Node Pool configuration is changed (for example, when splitting a Node Pool into two Node Pools) a file may be split across those two new pools until the next scheduled SmartPools job runs. If a Node Pool fills up and data spills over to another Node Pool so the cluster can continue accepting writes, a file may be split over the intended Node Pool and the default Spillover Node Pool. The last circumstance under which a file may span more than One Node Pool is for typical restriping activities like cross-Node Pool rebalances or rebuilds.

Performance with SmartPools

One of the principle goals of storage tiering is to reduce data storage costs without compromising data protection or access. There are several areas in which tiering can have a positive or negative performance impact on performance. It is important to consider each, and how SmartPools behaves in each situation.

Using SmartPools to Improve Performance

SmartPools can be used to improve performance in several ways:

- Location-based Performance
- Performance Settings
- SSD Strategies
- Performance Isolation

Location-based performance leverages SmartPools file pool policies to classify and direct data to the most appropriate media (SSD, SAS, or SATA) for its performance requirements.

In addition, SmartPools file pool rules also allow data to be optimized for both performance and protection.

As we have seen, SSDs can also be employed in a variety of ways, accelerating combinations of data, metadata read and write, and metadata performance on other tiers.

Another application of location-based performance is for performance isolation goals. Using SmartPools, a specific node pool can be isolated from all but the highest performance data, and File Pool policies can be used to direct all but the most critical data away from this node pool. This approach is sometimes used to isolate just a few nodes of a certain type out of the cluster for intense work. Because Node Pools are easily configurable, a larger node pool can be split, and one of the resulting node pools isolated and used to meet a temporary requirement. The split pools can then reconfigured back into a larger node pool.

For example, the administrators of this cluster are meeting a temporary need to provide higher performance support for a specific application for a limited period of time. They have split their highest performance tier and set policies to migrate all data not related to their project to other Node Pools. The servers running their critical application have direct access to the isolated Node Pool. A default policy has been set to ensure all other data in the environment is not placed on the isolated Node Pool. In this way, the newly-created Node Pool is completely available to the critical application.

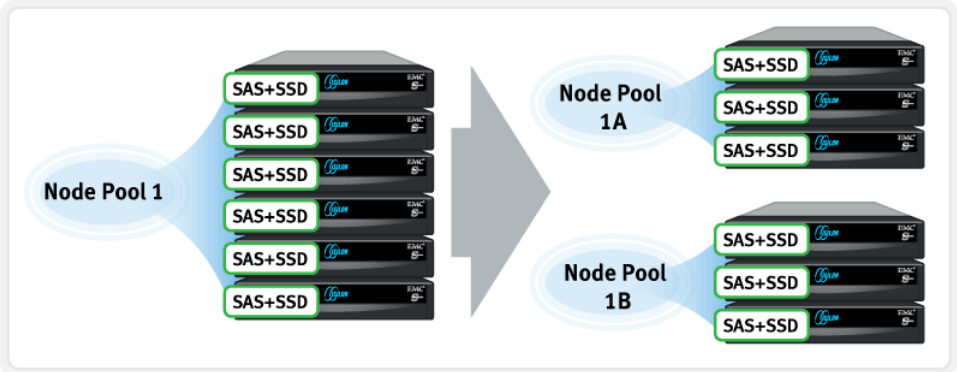


Figure 13. Performance Isolation Example

Data Access Settings

At the File Pool (or even the single file) level, Data Access Settings can be configured to optimize data access for the type of application accessing it. Data can be optimized for Concurrent, Streaming or Random access. Each one of these settings changes how data is laid out on disk and how it is cached.

Data Access Setting	Description	On Disk Layout	Caching
Concurrency	Optimizes for current load on the cluster, featuring many simultaneous clients. This setting provides the best behavior for mixed workloads.	Stripes data across the minimum number of drives required to achieve the data protection setting configured for the file.	Moderate prefetching
Streaming	Optimizes for high-speed streaming of a single file, for example to enable very fast reading with a single client.	Stripes data across a larger number of drives.	Aggressive prefetching
Random	Optimizes for unpredictable access to the file by performing almost no cache prefetching.	Stripes data across the minimum number of drives required to achieve the data protection setting configured for the file.	Little to no prefetching

Table 3. SmartPools Data Access Settings

As the settings indicate, the Random access setting performs little to no read-cache prefetching to avoid wasted disk access. This works best for small files (< 128KB) and large files with random small block accesses. Streaming access works best for sequentially read medium to large files. This access pattern uses aggressive prefetching to improve overall read throughput, and on disk layout spreads the file across a large number of disks to optimize access. Concurrency (the default setting for all file data) access is the middle ground with moderate prefetching. Use this for file sets with a mix of both random and sequential access.

Leveraging SSDs for Metadata & Data Performance

Adding SSDs within a Node Pool can boost performance significantly for many workloads. In the Isilon architecture SSDs can be used to accelerate performance across the entire cluster using SSD Strategies for data or metadata acceleration.

There are several SSD Strategies to choose from: Metadata Acceleration, Avoid SSDs, Data on SSDs, and Global Namespace Acceleration. The default setting, for Pools where SSDs are available, is ‘Metadata Acceleration’.

- Metadata read acceleration: Creates a preferred mirror of file metadata on SSD and writes the rest of the metadata, plus all the actual file data, to HDDs.
- Metadata read & write acceleration: Creates all the mirrors of a file's metadata on SSD.
- Avoid SSDs: Never uses SSDs; writes all associated file data and metadata to HDDs only. This strategy is used when there is insufficient SSD storage and you wish to prioritize its utilization.
- Data on SSDs: All of a node pool's data and metadata resides on SSD.

NOTE: SSD strategies and GNA work in concert. For example, if all of a cluster's data is set to "avoid SSD" strategy, enabling GNA has no effect, and the SSDs won't be used. If the files are all set to "metadata read" with GNA disabled, only the files on node pools with SSDs will get read acceleration. If GNA is then enabled, all files will get read acceleration.

Possibly the most common setting will be the "metadata read" SSD strategy for ALL files and GNA enabled. A SmartPools license will be required just to be able to direct different files to different tiers of storage. Effectively the SSD strategy will be globally set because it will be set to the same value on all files even though the storage pools for different file pools will be set differently.

Minimizing the Performance Impact of Tiered Data

In addition to areas where SmartPools can be used to enhance performance, it is important to note that any tiering approach will have some cases where performance can suffer as a result of data location, tiering data movement activity, etc.

Above we discussed data location and performance isolation as performance enhancement functions of SmartPools. As expected, if data is located on slower drive media it will typically be slower to access.

Another architectural attribute of scale-out NAS is that data may reside on one node, while the network access point for clients accessing that data may well be on, or load balanced to, another class of node. This access node may have different front end IO capabilities, cache, etc, but the performance characteristics will still primarily be governed by the class of storage on which the data resides.

A good rule of thumb is that while SAS is going to be faster than SATA in most cases, spindle counts can have a very important impact as well. The most important impact to performance in a SmartPools environment is the characteristics of the nodes in the Node Pool the application is connecting to, rather than what type of media the data physically resides on. For example, assuming there is no bottleneck on the network side, a streaming application will see very little difference in performance between data on a SATA node with less CPU power versus a SAS node with more CPU power as long as its connection into the cluster is via a Node Pool where the nodes have a great deal of cache. Similarly, for applications with Random access patterns, as long as the Node Pool they connect into has sufficient CPU, in most cases actual data location by spinning media type will not make a significant difference in performance.

Another important performance consideration in a tiered storage environment is the effect of data movement itself on overall system resources. The very action of moving data from one Node Pool to another does use system resources. SmartPools

mitigates the impact of data movement in multiple ways. First, architecturally, the job engine is very efficient and secondly through Impact Policies which are completely configurable by the end user to control how much in the way of system resources are allocated to data movement and when data movement takes place. The Job Engine impact policies are:

- **Paused**: Do nothing and wait.
- **Low**: Use 10% or less of Job Engine allocated resources.
- **Medium**: Use 30% or less of Job Engine allocated resources.
- **High**: Use maximum amount of Job Engine allocated resources.

Impact policy can be set once for SmartPools jobs, or controlled by a File Pool policy which can change the impact settings to match the typical peaks and lulls of the workload in a particular environment.

SmartPools Best Practices

For optimal cluster performance, we recommend observing the following OneFS SmartPools best practices:

- Define a performance and protection profile for each tier and configure accordingly.
- Enable SmartPools Virtual Hot Spares with a minimum of 10% space allocation.
- Avoid creating hard links to files which will cause the file to match different file pool policies
- If node pools are combined into tiers, the file pool rules should target the tiers rather than specific node pools within the tiers.
- Avoid creating tiers that combine node pools both with and without SSDs.
- Ensure that SSDs comprise a minimum of 2% of the total cluster usable capacity, spread across at least 20% of the nodes, before enabling GNA.
- Determine if metadata operations for a particular workload are biased towards reads, writes, or an even mix, and select the optimal SmartPools metadata strategy.
- Ensure that cluster capacity utilization (HDD and SSD) remains below 90%.

SmartPools Use Cases

OneFS SmartPools is typically used in the following ways:

Traditional File Tiering

- Store "current" data on the faster nodes and "old" data on cost-optimized storage.
- More flexibility in choosing which files reside where.
- Faster movement of files.

Metadata Acceleration

- Use SSD nodes to accelerate metadata operations (eg. searches and indexing) on archive data stored on NL nodes.

Application Partitioning

- Store files from different users, groups, or projects on separate hardware.

Equivalent Node Support

- Add or replace a new style of node to a pool of near identical previous generation nodes.

Auto-provisioning & Large Cluster Support

- Large clusters are automatically provisioned into disk pools to enhance reliability.
- Painless to add nodes.
- Appropriate protection is configured automatically.

SmartPools Workflow Examples

Example A: Storage Cost Efficiency in Media Post Production

In media post production, each phase typically deals with massive quantities of very large files. Editing and visual effects work is very storage resource intensive and, when complete, the products are then archived as reference work and retained for long periods. In the last five years alone, with the move to 3D rendering and High Definition formats, the amount of storage needed for post-production has increased as much as ten-fold and finished projects have doubled or tripled in size. Post production facilities are under increased pressure to provide faster turn times, richer effects, and an ever greater number of post-release projects to market tie-ins and licensed products and derivatives.

Post production creates a massive amount of data as their main product. This can grow to 100s of terabytes and often multiple petabytes, and needs to be accessed very quickly by many people and processes while a project is active. Then it continues its rich commercial life within the company, being drawn on intermittently for follow-on projects over long periods of time. This means that Post Production data is critical and timely, then dormant, until it's required again and the process repeats.

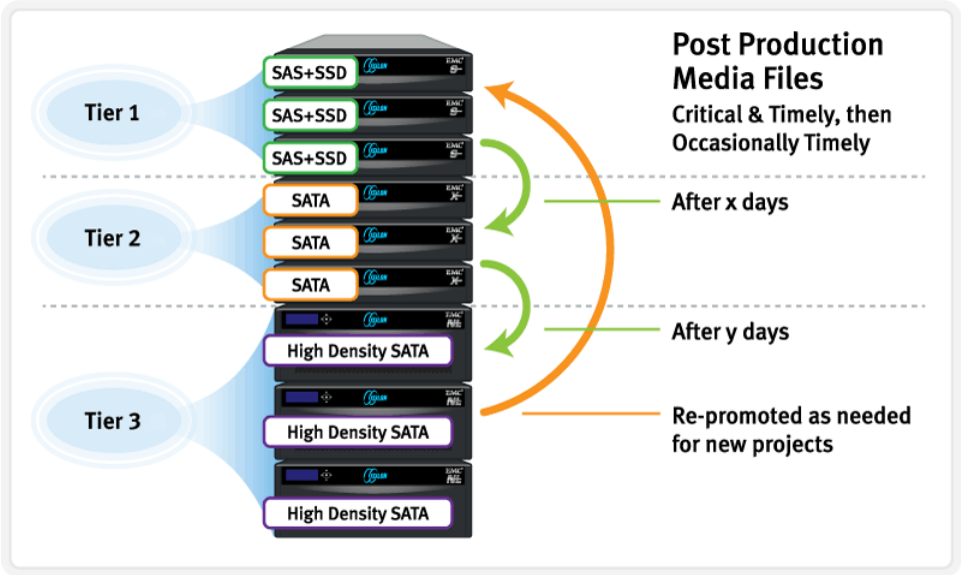


Figure 14. Media Post Production Example

The increasingly popular approach is to have a two or three-tiered solution where working data is on high performance disk and all the rest is on lower performance storage, all of it set for streaming access and protected at a high level. This way, 60 to 95% of all data under management can reside on less expensive disk, but it is all accessible at reasonable performance rates. Furthermore, this archived data can be promoted to the fastest disk at any point and it's completely protected for long periods of time.

Example B: Data Availability & Protection in Semiconductor Design

Logical and physical design and verification are complex early stages in the Electronic design automation (EDA) process for semiconductor production. This design data is critical and time to market is paramount in this industry. The company is counting on the revenue from this product, and the workflow is high priority. Multiple engineers from around the globe will typically be collaborating on the project, and all need timely access to the data.

Previous design data and other intellectual property is also heavily leveraged and reused in subsequent projects, so archive and retention requirements spanning decades are commonplace. Moreover, it can remain critical for many years, even decades. But it is not timely – no one has a need to access older designs at high performance speeds. All of the instances in which historical design data is referenced are not time critical enough to change the type of disk they are stored on, though an argument could be made against deep archive in the case of legal discovery timeframes. Therefore, older designs are critical, but not timely.

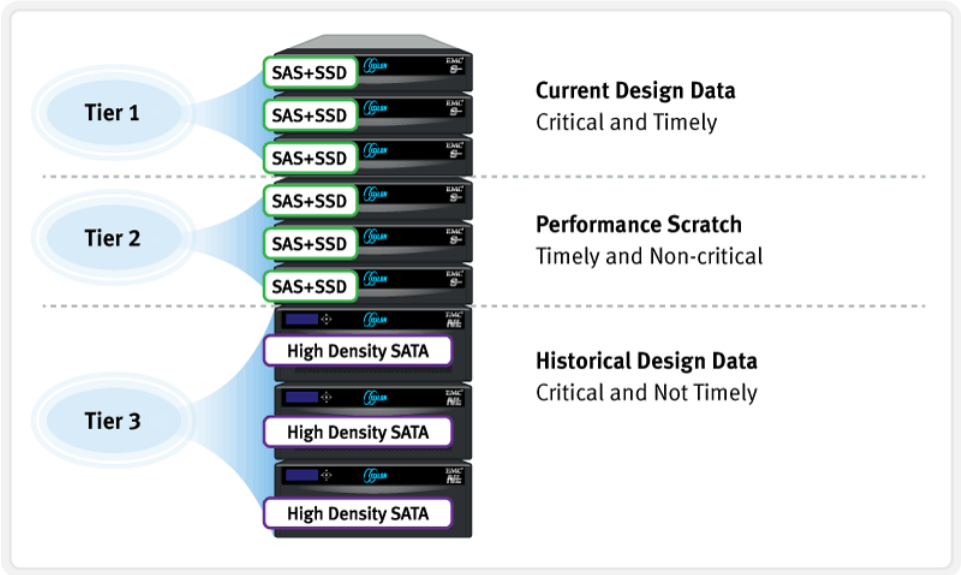


Figure 15. Data Performance & Protection in EDA

These engineering design data and historical design data examples are valuable because they illustrate the need for data protection for two sets of data that exist side by side in the same infrastructure, one of which is timely and critical, the other critical and not timely. The need is clearly to have a system that can serve up data quickly or less expensively, but protect both data types equally.

EDA workflows often employ an additional tier of storage, or 'scratch space', for the transient processing requirements of HPC compute farms. Typically, this has high transactional performance requirements and often employs SSD-based storage node. But, since scratch is temporary data, the protection and retention requirements are very low.

Using SmartPools, this can be achieved with a three tier architecture using high performance SAS and SSD (S Series) nodes for both the performance and scratch tiers and high-capacity SATA (NL Series) for the high-capacity archive tier. One File Pool policy would restrict Historical Design data to the high-capacity tier, protecting it at a high level, while another would restrict current design data to the fastest tier at the same protection level.

Example C: Investment Protection for Financial Market Data

Financial market data is collected by an investment firm from multiple global sources around the clock. This real time data needs to be captured and stored, where it is then analyzed to predict micro and macro market trends in order to justify purchase recommendations.

The data capture, plus subsequent analysis results, generates terabytes of data per day. Real-time capture is free, but if any time period is missed, historical data must be purchased from a service. Hence, performance and reliability are crucial. The majority of data analyzed is less than 48 hours old, but the data is also retained indefinitely. This allows long term analysis and review of model accuracy over time to be performed in any review timeframe.

Because this is a demanding production environment where data life is very long, there is a strong preference for scalable, durable systems that combine performance and archive capabilities with a compelling cost of ownership.

In this case, the preferred architecture would typically be a three tiered SmartPools cluster, where new data is ingested onto mid-performance spinning media, analysis is run on high-performance spinning media – possibly with SSDs – and older data that is used only intermittently would be moved to slower, cost optimized disk.

The original equipment stays online for many years – just moving down the hierarchy as completely capitalized capacity – while new drives and new nodes with better price and/or performance characteristics are added above them. This OneFS investment protection approach helps avoid the pain and disruption of frequent hadware replacement.

Example D: Metadata Performance for Seismic Interpretation

The seismic data required for energy exploration is typically vast, and growing, with many organizations keeping petabytes of data online for faster analysis and more accurate drilling predictions. This data may be arranged in tens or hundreds of thousands of files across thousands of directories. Often less than 30% of those files may be used in any given month, but administrators constantly need to locate files for pre- and post-stack analysis, interpretation, and project planning. The latencies involved with traversing enormous directory structures and file quantities, listing directory contents, and other metadata operations, are often measured in minutes. This is clearly unacceptable for operations that are repeated multiple times per day. Seismic data is the core of exploration and is therefore critical, and while for rarely accessed files the data may not be timely, the metadata is. In summary, seismic data is critical and its metadata is timely; its data may be timely or not timely. In this case, Global Namespace Acceleration is the logical approach.

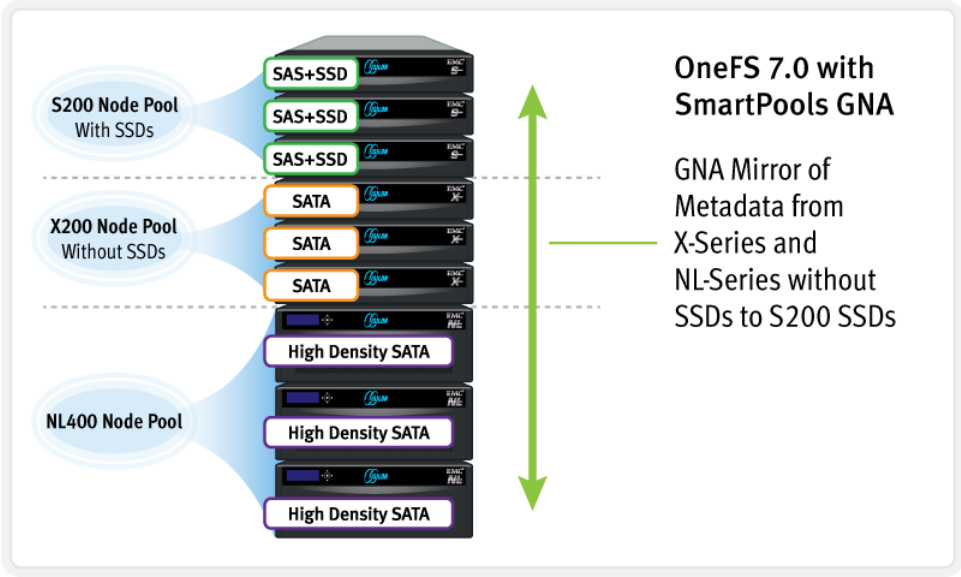


Figure 16. Global Namespace Acceleration

All of the metadata is stored on SSD, but data may be relegated to faster or slower disk based on its relative timeliness. The result is the ability to instantly locate any data, while still being able to cost reduce the majority of the data on the system. Because metadata is accelerated, there are performance benefits to many other activities as well including backups, migration and replication.

Conclusion

To date, traditional storage tiering implementations have typically been expensive, technically risky, and administratively complex. More importantly, they have often failed to achieve their primary goal of aligning data value with accessibility, protection, and performance requirements.

SmartPools' unique integration with Isilon OneFS, the industry's leading Scale-Out NAS architecture, delivers on the promise of simple storage tiering with significant storage cost savings, without sacrificing performance or data protection.

With its simple, powerful interface, flexible options and intelligent default settings, SmartPools is easy to configure and manage, and scales from terabytes to petabytes. Scalability to petabytes and the ability to add new capacity and new technologies while retaining older capacity in the same system means strong investment protection. Integration with OneFS core functions eliminates data migration risks and gives the user control over what system resources are allocated to data movement.

To learn more about SmartPools and other EMC Isilon products please see http://www.emc.com/domains/isilon/index.htm.

About EMC Isilon

As the global leader in scale-out storage, EMC Isilon delivers powerful yet simple solutions for enterprises that want to manage their data, not their storage. Isilon products are simple to install, manage and scale, at any size. And, unlike traditional enterprise storage, Isilon stays simple no matter how much storage is added, how much performance is required or how business needs change in the future. Information about Isilon can be found at http://www.emc.com/isilon.