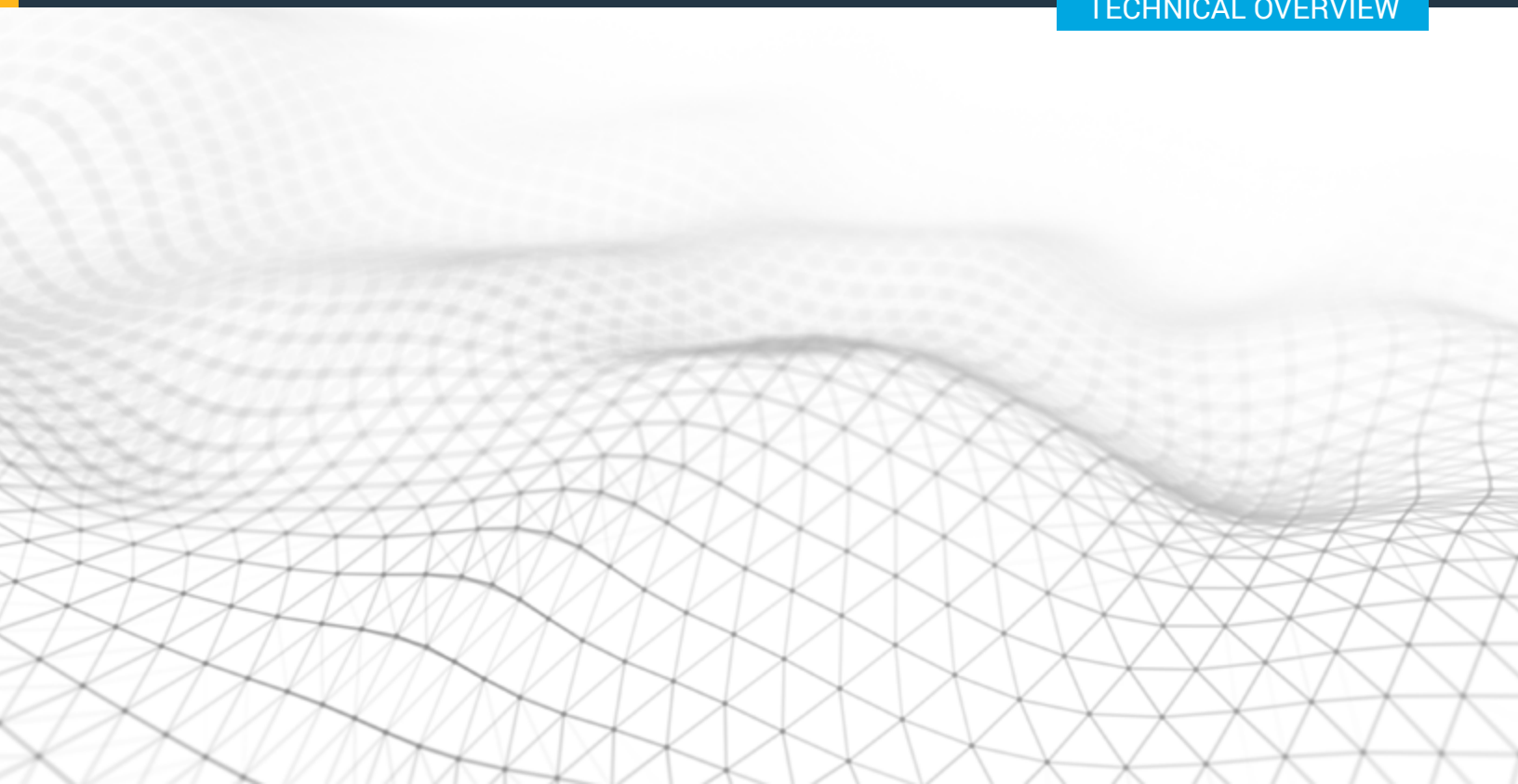
PANASAS®

PanFS 9: Architectural Overview

TECHNICAL OVERVIEW



PANASAS

PanFS Architecture

Executive Summary **3**

The PanFS Architecture: The Preeminent HPC Storage Architecture **4**

Separation of Control and Data Planes 4

Linear Scale-Out of Director and Storage Nodes 5

Parallel and Direct Transfers from the Client 5

File Maps, Parallelism, and Erasure Coding 5

Full POSIX Semantics with Cache Coherency 6

Data Management: Volumes, Snapshots, and Quotas 6

Data Security: ACLs, SELinux, and Encryption at Rest 6

Remarkable Mixed Workload Performance 7

NFS and SMB/CIFS Gateway on Director Nodes 7

Let's Dive Deeper **8**

BladeSets and Three Generations of Storage Nodes 8

Director Nodes, the Repset, and the President 8

Director Software and Deeply Automated Failure Recovery 8

Storage Nodes are Object Storage Devices 9

Each File Is Individually Erasure Coded for Maximum Reliability 9

Reliability That Increases with Scale 10

An Architecture of Even More Reliability 10

Preventing Hot Spots 10

Consistent Mixed Workload Performance 13

The Most Performance Efficient HPC File System 13

Single-Tier Solution Versus Complex/Costly External Tiering 14

Conclusion – PanFS is The Most Comprehensive HPC Storage Solution 15

PANASAS

Executive Summary

HPC environments, by their very nature, tend to be large and are usually quite complex. Whether it's pushing the boundaries in life and physical sciences or supporting reliable engineering, it takes many computers operating together to analyze or simulate the problems at hand. The quantity of data required, and the access performance to keep all those computers busy, can only be met by a true parallel file system, one that maximizes the efficiency of all storage media in a seamless, total-performance storage system.

The PanFS® parallel file system delivers the highest performance among competitive HPC storage systems at any capacity, and takes the complexity and unreliability of typical high-performance computing (HPC) storage systems off your hands, and it does so using commodity hardware at competitive price points.

PanFS orchestrates multiple computers into a single entity that serves your data to your compute cluster. Through sophisticated software, multiple computers that each have HDDs and/or SSDs attached to them will work together to provide hundreds of Gigabytes per second (GB/s) of data being read and written by your HPC applications. PanFS manages this orchestration without manual intervention, automatically recovering from any failures, continuously balancing the load across those computers, scrubbing the stored data for the highest levels of data protection, and encrypting the stored data to protect it from unwanted exposure.

PanFS was the first storage system designed with the parallel file system architecture that is the de facto dominant storage architecture in HPC systems to this day. While the foundation for PanFS was laid over 20 years ago, the file system continues to adopt the latest technology advancements to provide the exceptionally high performance, reliability and low-touch administration our customers have come to expect and rely upon.

In this document, we're going to take a "breadth-first" tour of the architecture of PanFS, looking at its key components then diving deep into the main benefits.

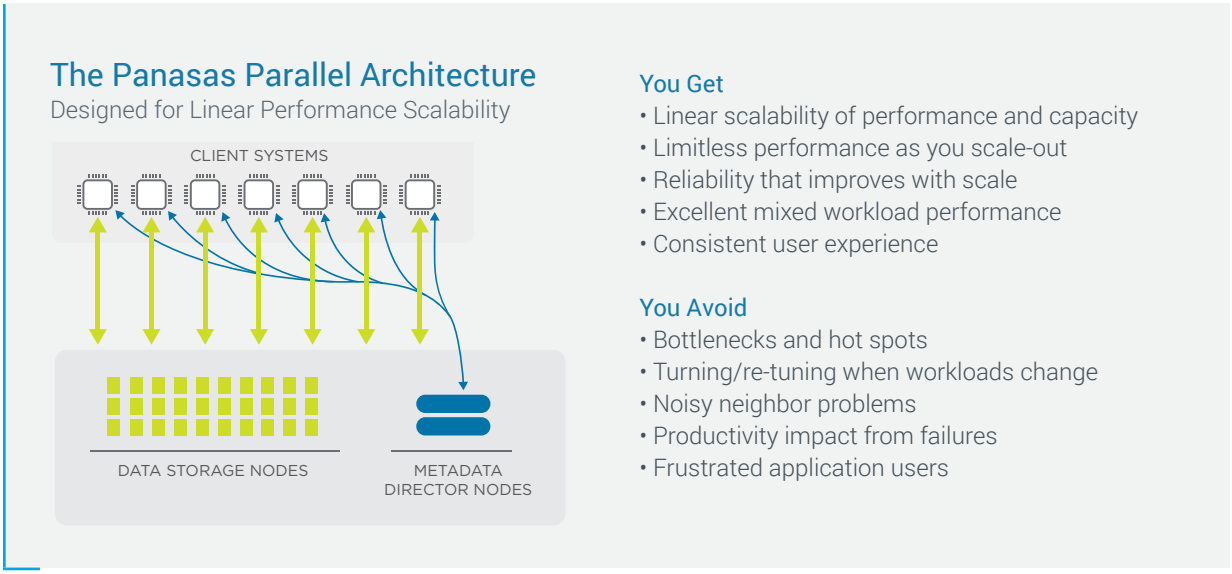


Figure 1: The PanFS Parallel File System

The PanFS Architecture: The Preeminent HPC Storage Architecture

There are three components working together to power the PanFS file system: Director Nodes, Storage Nodes, and the DirectFlow Client driver. The Director Nodes and Storage Nodes are computer systems dedicated to running PanFS software, and together they comprise the Panasas ActiveStor® appliance. The DirectFlow Client driver is a loadable software module that runs on Linux compute servers ("Clients") and interacts with the Director Nodes and Storage Nodes to read and write the files stored by PanFS. Any required administration happens via the GUI or CLI running on a Director Node. There's no need to interact with Storage Nodes or the DirectFlow Client driver - the Director Nodes take care of that.

All the Director Nodes and Storage Nodes work together to provide a single file system namespace that we call a "realm". All the Linux compute servers running DirectFlow Clients that access a realm are not considered part of that realm, instead they are considered Clients.

Separation of Control and Data Planes

PanFS explicitly separates the "control plane" from the "data plane":

- **Director Nodes in PanFS are the core of the control plane.** They process file system metadata (e.g.: directories, file attributes, etc.), coordinate the actions of the Storage Nodes and the DirectFlow Client drivers for file accesses, manage membership and status within the PanFS storage cluster, and control all failure recovery and data reliability operations. Director Nodes are simple, commodity compute servers with a high-speed networking connection, significant DRAM capacity, and an NVDIMM memory for transaction logs.
- **Storage Nodes in PanFS are the core of the data plane.** They are the only part of the overall architecture that stores data or metadata. While Director Nodes serve and modify file system metadata, they use Storage Nodes to store it.

PANASAS

> One PanFS customer gradually added over 1,500 Storage Nodes and 150 Director Nodes to a single PanFS realm over a period of several years and saw linear growth in performance every step of the way. This was done while supporting a user community of several thousand researchers, all running their own HPC applications at the same time.

Storage Nodes are commodity systems, but they are models we've chosen for their carefully balanced hardware architecture in terms of their HDD, SSD, NVMe, and DRAM capacities, strength of CPU, networking bandwidth, etc.

- **The DirectFlow Client driver is a loadable file system implementation installed on compute servers and used by your application programs like any other file system.** It works with the Director Nodes and Storage Nodes to deliver fully POSIX-compliant and cache-coherent file system behavior, from a single namespace, across all the servers in the compute cluster. All the popular Linux distributions and versions are supported.

Linear Scale-Out of Director and Storage Nodes

PanFS scales out both Director Nodes and Storage Nodes. For more metadata processing performance, more Director Nodes can be added. For more capacity or more storage performance, more Storage Nodes can be added.

In scale-out storage systems like PanFS, there simply is no maximum performance or maximum capacity. To achieve more performance or more capacity, simply add a few more nodes. PanFS has been architected to provide linear scale-out, e.g., adding 50% more Storage Nodes will deliver 50% more performance in addition to 50% more capacity.

Parallel and Direct Transfers from the Client

PanFS is a parallel file system that can consistently deliver orders of magnitude more bandwidth than the standard NFS or SMB/CIFS protocols. Each file stored by PanFS is individually striped across many Storage Nodes, allowing each component piece of a file to be read and written in parallel, increasing the performance of accessing each file.

PanFS is also a direct file system that allows the compute server to talk over the network directly to all the Storage Nodes. Typical Enterprise products will funnel file accesses through "head nodes" running NFS or SMB/CIFS and then across a backend network to other nodes that contain the HDDs or SSDs that hold the file. Obviously, that can create bottlenecks when data traffic piles up on the head nodes, plus it brings additional costs for a separate backend network. In contrast, for each file that the application wants to access, the DirectFlow Client on the compute server will talk over the network directly to all the Storage Nodes that hold that file's data. The Director Nodes are out-of-band, which makes for a much more efficient architecture and one that is much

> All the processes running on all the compute servers will see the same file system namespace, metadata, and user file data contents, all the time.

less prone to hotspots, bottlenecks, and erratic performance common to Scale-Out NAS systems.

File Maps, Parallelism, and Erasure Coding

PanFS makes use of the multiple Storage Nodes by assigning a map to each file that shows where all the striped component parts of that file can be found, which Storage Node holds each part. The DirectFlow Client uses that map to know which Storage Nodes to access, directly as well as in parallel.

PanFS also uses Network Erasure Coding as part of that striping to ensure the highest levels of data integrity and reliability.

Full POSIX Semantics with Cache Coherency

The POSIX Standard defines the semantics of modern file systems. It defines what a file is and what a directory is, what attributes each has, and the open, close, read, write, and lseek operations used to access files. Billions of lines of software have been written that leverage the POSIX standard for access to storage.

The DirectFlow Client provides the same semantics as a locally-mounted, POSIX-compliant file system, with the assurance that if some other process (on another compute server) is writing to a file at the same time this process is reading from it, this process will not read stale data. In file system terminology, PanFS has been engineered to provide cache coherency across all the nodes running the DirectFlow Client.

Data Management: Volumes, Snapshots, and Quotas

At least one Volume must be created at the root of the PanFS file system namespace, but multiple Volumes are recommended. While each Volume is a separate unit of administrative control, they all share the capacity of the realm. For example, each Volume could have a different set of per-user quota values or snapshot schedule. Each Volume is just a normal directory tree in the PanFS namespace, except that hardlinks that cross between one Volume and another are not supported.

Per-Volume snapshots are a convenient way to enable user-directed recovery of prior versions of files, with no system administrator involvement required. They are accessed via the typical hidden and synthetic ".snapshots" subdirectory in each directory in the namespace.

Both "hard" and "soft" quotas are supported, at both the per-user level and Volume level. Each user can have their own soft and hard capacity quota for each Volume, and each Volume can be configured with its own soft and hard quota for total capacity consumed by all users in that Volume.

Data Security: ACLs, SELinux, and Encryption at Rest

PanFS supports two features that prevent unauthorized data access while the realm is online, ACLs and SELinux, and one that prevents unauthorized data access while the realm is

> Data security is becoming an ever more important characteristic of storage systems and Panasas will continue developing solutions that protect your data from unauthorized access.

> PanFS has a wide performance profile that supports a range of small files as well as large files in very randomized workloads such as might be found in a shared resource center or in hosting home directories, in addition to the core HPC large file workloads.

offline, Encryption at Rest. The Encryption at Rest and SELinux data security features are new with PanFS release 9.

PanFS supports Access Control Lists (ACLs) on each file and directory in addition to the traditional Linux user id, group id, and mode bits such as "joe dev -rwxr-xr-x". PanFS ACLs are fully compatible with Windows ACLs, with ActiveDirectory-based and LDAP-based account management, and provide fine grained control over which user accounts can execute which operations on each file or directory.

Running SELinux on the client systems that access PanFS via DirectFlow provides a kernel-implemented set of mandatary access controls that confine user programs and system services to the minimum level of file and data access that they require. PanFS integrates the security. selinux security label that enables this into the PanFS inode structure, resulting in near-zero added access latency when SELinux is used. This is the foundation for true Multi-Level Security policy implementations that allow users and data in different security levels and compartments to share the same compute and storage resources.

All Panasas Storage nodes use industry-standard self-encrypting drives (SEDs) which implement NIST-approved AES-256 hardware-based encryption algorithms built into each drive. These drives are specifically designed so that encryption does not reduce performance. PanFS allows encryption-at-rest to be non-disruptively enabled, disabled, and drive keys to be rotated upon command, as well as cryptographic erasure for securely repurposing of the storage without risking exposure of the prior data.

Key management is outsourced via the Key Management Interoperability Protocol (KMIP) to well-established and proven cyber security key management solutions that provide centralized and simplified key management. During normal runtime PanFS periodically verifies that the KMIP server is alive and well and contains all the right keys, and will raise an alert if there is any type of problem.

Remarkable Mixed Workload Performance

The PanFS architecture not only delivers exceptionally high but also very consistent performance for workloads that include a wide range of file sizes and access patterns and workloads that change significantly over time.

The effect is a dramatic broadening of the use cases that PanFS can support in an HPC environment compared to other parallel file systems. All other parallel file systems require time consuming and laborious tuning and retuning as workloads change.

NFS and SMB/CIFS Gateway on Director Nodes

One of the roles of the Director Nodes in PanFS is to act as gateways that translate NFS and SMB/CIFS operations into DirectFlow operations, allowing clients such as laptops and workstations to access the same namespace and data as the HPC compute cluster does. PanFS provides high performance NFSv3 and SMB/CIFS v3.1 access (via a recent release of Samba), but as a result of its parallel and direct nature, the DirectFlow Protocol will always be the highest performance path to PanFS storage.

Let's Dive Deeper

BladeSets and Three Generations of Storage Nodes

PanFS can be configured to create what we call a "BladeSet" for different classes of Storage Nodes. For example, Storage Nodes with a capacity of 28TB each should not be combined into the same BladeSet as Storage Nodes with a capacity of 116TB each. This helps to evenly spread the workload across the pool of Storage Nodes and avoid hotspots. PanFS can support multiple BladeSets in a realm and in the same namespace at the same time. Any given Volume can only draw capacity from a single BladeSet.

A typical reason to have different classes of Storage Nodes in a realm is as a result of expanding a realm over time by adding the latest generation Storage Nodes each time. The latest Storage Nodes might need to be in a different BladeSet if they are different enough from the other Storage Nodes in the realm. Adding Storage Nodes, whether in the same BladeSet or a new one, is a non-disruptive operation performed while the file system is online.

Director Nodes, the Repset, and the President

Any realm needs a minimum of three Director Nodes, but there's no maximum. The administrator should designate three or five Director Nodes, the odd number makes it easier to break ties in voting, out of the total set of Director Nodes in the realm to be the rulers of the realm. Each of those rulers will have an up-to-date, fully replicated copy of the configuration database for the realm, which is why the group of rulers is called the repset. Those rulers will elect one of themselves to be the realm president who will coordinate all the configuration, status, and failure recovery operations for the realm. If the Director Node currently designated the president were to fail, another member of the repset would be immediately and automatically elected to be the new president. The configuration database is kept up to date on all members of the repset via a distributed transaction mechanism.

Director Nodes make themselves available to do any of a large set of operational and maintenance tasks the president may need them to do. Those include managing a number of Volumes, being a gateway into PanFS for the NFS and/or SMB/CIFS protocols, helping perform background scrubbing of user data, helping to recover from a failure of a Storage Node, and helping to perform Automatic

The Panasas Parallel Architecture

Designed for Linear Performance Scalability

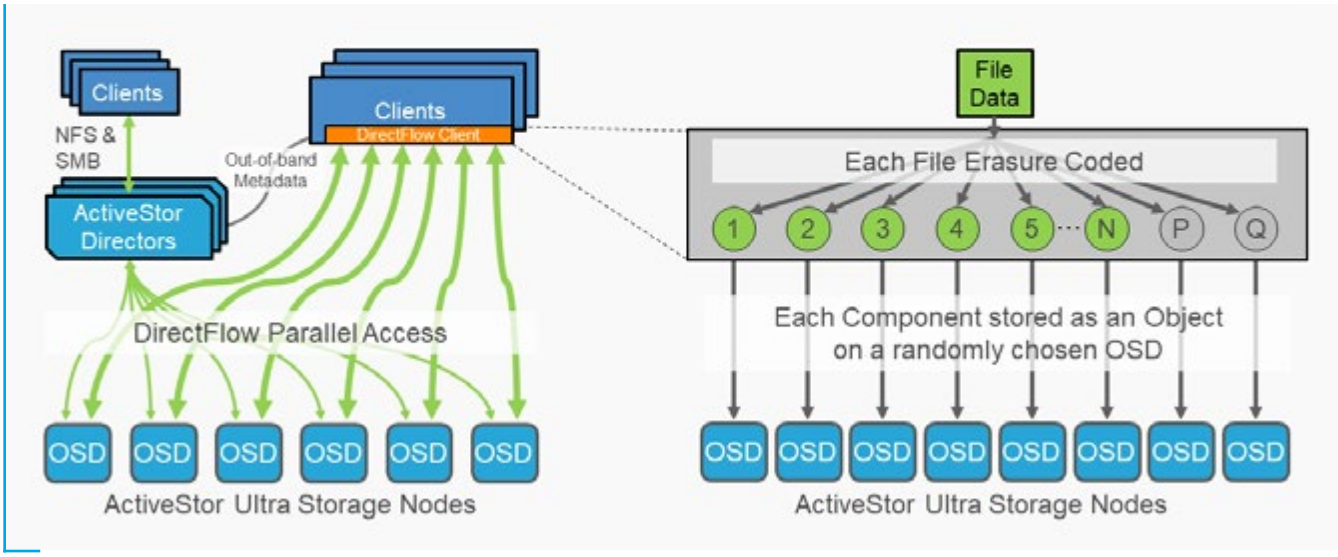


Figure 2: Per-File Erasure Coding Across OSDs

Capacity Balancing across the Storage Nodes in a BladeSet, among others. The president's decisions can change over time as circumstances change; for example: moving a gateway or a Volume to a different Director Node. However, the president will only do that if the operation is fully transparent to client systems.

Director Software and Deeply Automated Failure Recovery

In addition to the other responsibilities Director Nodes have managing the POSIX semantics and cache coherency of the files in a Volume, they also need to manage the status and health of each of the Storage and Director Nodes that are part of the realm. Panasas has analyzed the failure modes of each of the commodity platforms that PanFS has been ported to and we have included recovery logic for each of those cases into the Director Node software stack. That additional engineering work is a significant contributor to the overall reliability of a PanFS realm and is one of the keys to its low-touch administration. PanFS automatically reacts to failures and recovers from them, taking care of itself.

We have customers who have had zero unplanned downtime over the multi-year life of their PanFS realms.

Storage Nodes are Object Storage Devices

Storage Nodes in PanFS are actually highly sophisticated Object Storage Devices (OSDs) and we gain the same scale-out and shared-nothing architectural benefits from our OSDs as any Object Store would. The definition of an Object used in our OSDs comes from the Small Computer System Interface (SCSI) standard definition of Objects rather than the Amazon S3 definition of Objects.

A RAID array reconstructs the contents of drives while PanFS reconstructs the contents of files.

PanFS has linear scale-out reconstruction performance in the event of a Storage Node failure, dramatically reducing recovery times, so PanFS reliability goes up with scale.

Each File Is Individually Erasure Coded for Maximum Reliability

PanFS uses Objects to store POSIX files, but it does so in a differently than how S3 Objects are typically used to store files. Instead of storing each file in an Object, PanFS stripes a large POSIX file across a set of Component Objects and adds additional Component Objects into that stripe that store the P and Q data protection values of N+2 erasure coding. Using multiple Objects per POSIX file enables the striping of a file that is one of the sources of a parallel file system's performance.

While large POSIX files are stored using erasure coding across multiple Component Objects, small POSIX files use triple-replication across three Component Objects. That approach delivers higher performance than what can be achieved by using erasure coding on such small files and makes it more space efficient as well. Unless the first write to a file is a large one, it will start as a small file. If a small file grows into a large file, at the point that the erasure coded format becomes more efficient, the Director Node will transparently transition the file to the erasure coded format.

When a file is created and as it grows into a large file, the Director Node that is managing those operations will randomly assign each of the individual Component Objects that make up that file to different Storage Nodes. No two Component Objects for any file will be in the same failure domain.

The Storage Nodes for any given file are not selected completely randomly, however. The selection process does consider the current capacity utilization of each Storage Node in order to keep the capacity and bandwidth of the pool of Storage Nodes balanced. We call that Passive Capacity Balancing to differentiate it from Active Capacity Balancing.

Reliability That Increases with Scale

Any system can experience failures and as systems grow larger, their increasing complexity typically leads to lower overall reliability. For example, in an old-school RAID subsystem, since the odds of any given HDD failing are roughly the same during the current hour as they were during the prior hour, more time in degraded mode equals higher odds of another HDD failing while the RAID subsystem is still in degraded mode. If enough HDDs were to be in a failed state at the same time there would be data loss, so recovering back to full data protection levels as quickly as possible becomes the key aspect of any resiliency plan.

If a Panasas Storage Node were to fail, PanFS would reconstruct only those Component Objects

that were on the Storage Node that failed, not the entire raw capacity of the Storage Node like a RAID array would. PanFS would read the Component Objects for each affected file from all the other Storage Nodes and use each file's erasure code to reconstruct the Component Objects that were on the failed Node.

When a BladeSet in PanFS is first set up, it sets aside a configurable amount of spare space on all the Storage Nodes in that BladeSet for reconstructions. When PanFS reconstructs the missing Component Objects, it writes them to the spare space on randomly chosen Storage Nodes in the same BladeSet. As a result, during a reconstruction, PanFS uses the combined write bandwidth of all the Storage Nodes in that BladeSet.

PanFS also continuously scrubs the data integrity of the system in the background by slowly reading through all the files in the system, validating that the erasure codes for each file match the data in that file.

An Architecture of Even More Reliability

The N+2 erasure coding that PanFS implements protects against two simultaneous failures within any given BladeSet without any data loss. More than two failures in a realm can be automatically and transparently recovered from as long as there are no more than two failed Storage Nodes at any one time in any given BladeSet.

If in extreme circumstances three Storage Nodes in a single BladeSet were to fail at the same time, PanFS has one additional line of defense that would limit the effects of that failure. All directories in PanFS are stored quad-replicated (four complete copies of each directory, no two copies on the same Storage Node) rather than the triple-replicated or erasure coded formats used for regular files.

If a third Storage Node were to fail in a BladeSet while two others were being reconstructed, that BladeSet would immediately transition to read-only state as a result. Only the files in that BladeSet that had Component Objects on all three of the failed Storage Nodes would have lost data, a smaller and smaller percentage as the size of the BladeSet increases. All the other files in the BladeSet would be unaffected or recoverable using their erasure coding.

Since PanFS would have one complete directory tree still available to it, it can identify the full pathnames of precisely which files need to be restored from a backup or reacquired from their original source, and can therefore also recognize

> Data scrubbing is a hallmark of Enterprise class storage systems and is only found in one HPC class storage system, PanFS.

> PanFS has a much more intelligent, graceful failure model that we call "Extended File System Availability" compared to all-or-nothing architectures.

> No "hotspots" means that all the Storage Nodes are evenly loaded, they are all contributing equally to the performance of the realm.

PanFS Performance Innovations

Matching I/O patterns to storage device capabilities

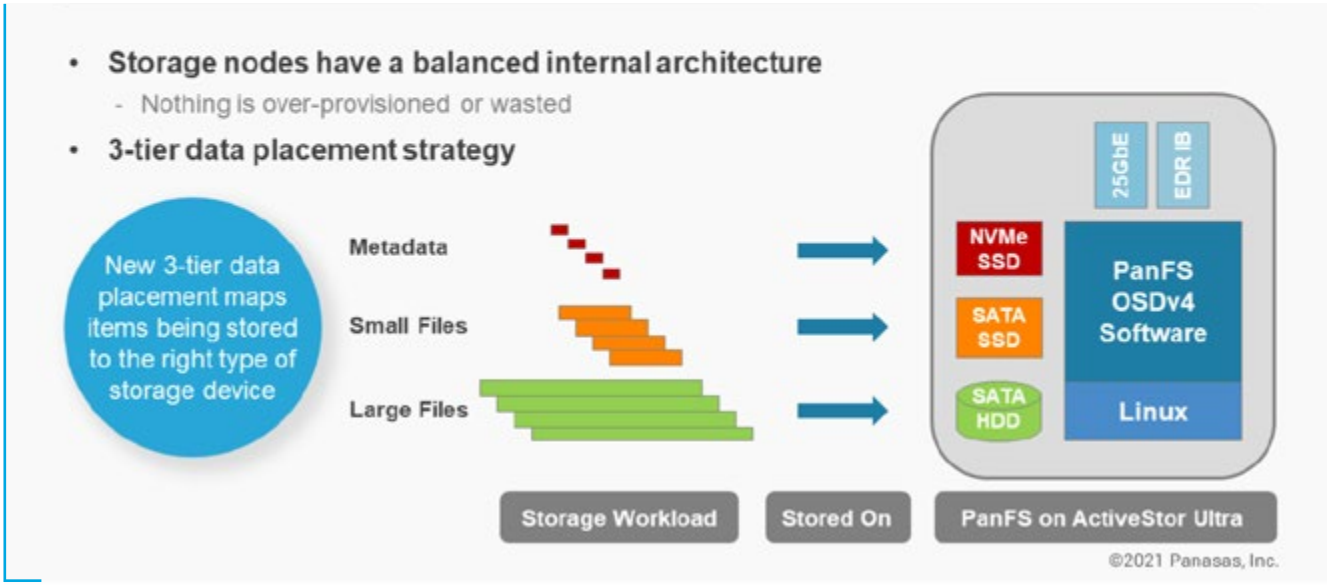


Figure 3: Intelligent Data Placement that Optimizes for Device Performance

which files were either unaffected or recovered using their erasure coding. PanFS is unique in the way it provides clear knowledge of the impact of a given event, as opposed to other architectures which leave you with significant uncertainty about the extent of the data loss.

Preventing Hot Spots

The random assignment of Component Objects to Storage Nodes spreads the load from any hot files across those Nodes. In most PanFS installations the number of Storage Nodes is much larger than the typical stripe width of a file, so each hot file is very likely to only share a few Storage Nodes with any other hot files. That greatly reduces the odds of any one Storage Node becoming overloaded and impacting the performance of the whole realm. The result is much more consistent system performance, no matter what workload is being requested by the compute servers or how it changes over time, and PanFS does so without any tuning or manual intervention.

> Both active and passive capacity balancing happen seamlessly and continuously without any administrator intervention.

Since scalable performance depends upon spreading all the files relatively evenly across the pool of Storage Nodes, PanFS includes both Passive Capacity Balancing and Active Capacity Balancing. In Passive Capacity Balancing, when new files are created, we apply a slight bias when assigning the new Component Objects to Storage Nodes by placing slightly more Component Objects onto Storage Nodes that have somewhat lower utilization. Over time, that tends to keep the Storage Nodes equalized.

> The ActiveStor Ultra node architecture results in three significant advantages: each storage device only performs operations it is good at so we get more from it, small file accesses never wait for large files since they're on separate devices, and latency-sensitive metadata accesses never wait for anything.

If Passive Capacity Balancing is insufficient, for example if many files are deleted at once and the balance is now off by more than a threshold, PanFS also includes Active Capacity Balancing. The realm president will ask the pool of Directors to examine the utilization of all the Storage Nodes and transparently move Component Objects from over-full Storage Nodes to underutilized Storage Nodes.

Both Passive and Active Capacity Balancing are used when new Storage Nodes are incorporated into a realm. If a realm is expanded by 20%, for example, Passive Capacity Balancing will immediately begin using those new Storage Nodes as one part of some newly created files. In the background, since the utilization of those new Storage Nodes is so much lower than the utilization of the existing Storage Nodes, Active Capacity Balancing will begin moving Component Objects to the new Storage Nodes. The new Storage Nodes will immediately begin contributing to the performance of the realm and will gradually pick up more and more of the realm's workload until all the Storage Nodes are contributing equally to the overall performance of the realm again.

Consistent Mixed Workload Performance

The price/performance and mixed-workload performance of a storage subsystem depends heavily upon how effectively its architecture makes use of the performance of the underlying commodity storage devices (e.g., HDDs, SSDs, etc). Panasas is uniquely expert in getting the most performance from all those devices.

In our most recent storage appliance product, ActiveStor Ultra, we utilize the PanFS feature that we call Dynamic Data Acceleration. This allows ActiveStor Ultra to use a carefully balanced set of HDDs, SSDs, NVMe SSD, NVDIMM, and DRAM and provide a combination of excellent performance and low cost per TB.

- We use the DRAM in each OSD as an extremely low latency cache of the most recently read or written data and metadata.
- An NVDIMM (a storage class memory device) is the lowest latency type of persistent storage device available, and we use one to store our transaction logs: user data and metadata being written by the application to the OSD, plus our internal metadata. That allows PanFS to provide very low latency commits back to the application.
- NVMe SSDs are built for very low latency accesses, so we store all our metadata in a database and keep that database on an NVMe SSD. Metadata accesses are very sensitive to latency, whether it is POSIX metadata for the files being stored or metadata for the internal operations of the OSD.
- SSDs provide cost-effective and high-bandwidth storage as a result of not having any seek times, so that's where we keep our small Component Objects. Any POSIX file of less than about 1MB in size will be stored on SSD.

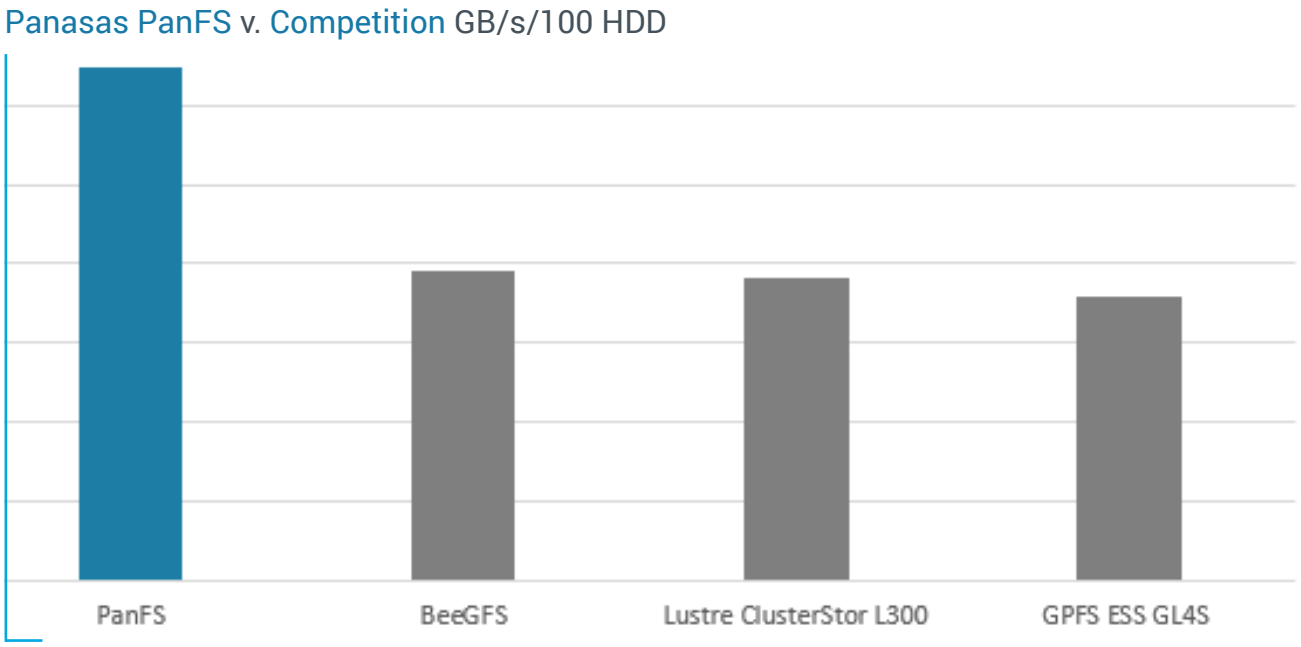


Figure 4: Bandwidth Delivered (in GB/s) per 100 HDDs for Comparable HPC Storage Systems

- HDDs will provide high bandwidth data storage if they are never asked to store anything small and only asked to do large sequential transfers. Therefore, we only store large Component Objects on our low-cost-per-TB HDDs.

To gain the most benefit from the SSD's performance, we try to keep the SSD about 80% full. If it falls below that, we will (transparently and in the background) pick the smallest Component Objects in the HDD pool and move them to the SSD until it is about 80% full. If the SSD is too full, we will move the largest Component Objects on the SSD to the HDD pool. Every ActiveStor Ultra Storage Node performs this optimization independently and continuously. It's easy for an ActiveStor Ultra to pick which Component Objects to move, it just needs to look in its local NVMe-based database.

The Most Performance Efficient HPC File System

Comparing the efficiency of HPC storage systems can be difficult. The range of architectural assumptions and technologies applied by each make apples-to-apples comparisons very difficult, so we need to start by deciding on a metric of comparison. We believe that it's fair to compare hybrid configurations such as ActiveStor Ultra that contain both HDDs and SSDs to other hybrid systems and to compare all-flash deployments to all-flash deployments.

> As a result of focusing on using each type of storage device only for what it is good at, PanFS can deliver twice the performance from a given overall capacity point as other products.

In hybrid systems, the dominant share of the total capacity in the system will be in the cost-effective HDDs, so measuring the peak aggregate performance delivered to the compute cluster divided by the number of HDDs in the storage cluster is an easy-to-understand approach. It shows how efficient the file system architecture is and how much performance it can extract from a given number of HDDs. In the graph above, we've plotted

GB/s delivered per 100 HDDs in the storage system, and PanFS is clearly at the next level.

Single-Tier Solution Versus Complex/Costly External Tiering

Storage Nodes in PanFS are typically built using an all-hot design principle. We first decide how many of each type of storage device (e.g.: NVMe SSDs, SSDs, HDDs, etc) will be in a new Storage Node design. We then ensure that there is enough network bandwidth in and out of that Storage Node to keep every storage device in that Node completely busy all the time. And finally, we include appropriate CPU power and DRAM in each Node to drive both the network and the devices to full performance.

In contrast, some other storage solutions segregate their flash-based devices into a hot tier and the more cost-effective HDDs into a cold tier in an attempt to reduce the overall average cost per TB. Typically, the cold tier takes the form of a heavyweight, separately administered cold archive (e.g.: with an S3 interface), plus an additional separately administered product that will move data between the hot tier and the cold tier. Such tiered storage solutions move files between their hot and cold tiers based upon temperature, how recently a file has been accessed. That's based upon the simplistic theory that if you haven't accessed a file in a while, you won't access it for a while longer. For some important workloads such as AI, that simply isn't true.

PanFS uses Component Object size as the metric to move content between the different device types within each ActiveStor Ultra Node. Each device type has different per-TB costs, but our purpose in moving the data between the device types is to get the most performance out of those storage devices rather than trying to manage costs.

> Our Storage Node's "balanced architecture" gives us the optimal price/performance, nothing is over-provisioned, and everything is working at peak efficiency.

> We gain cost savings as an intended side benefit of our focus on device performance rather than losing performance as a side effect of a focus on cost.

ActiveStor Ultra's multiple storage media types, when managed by PanFS, form an elegant storage platform that can automatically adapt to changing file sizes and workloads, with all those devices directly contributing to the application performance you need. With heavyweight tiering, since an application can typically only directly access the hot tier, the hardware in the cold tier cannot contribute to the performance the application needs. That results in three types of costs:

1. The stranded performance costs of not allowing the HDDs in the cold tier to contribute to application performance, thereby lowering the effective price/performance of the system.
2. The monetary costs of the additional networking and hot tier storage performance required to move data between the hot and cold tiers without impacting application I/O performance.
3. The direct costs of administering two separate tiers plus the policies required to move data back and forth. The cost of skilled employees is a significant piece of the overall TCO of HPC-class storage systems.

PANASAS

> All the storage hardware you buy should be contributing to the performance you need.
>
> We like to say that extraordinary architecture leads to extraordinary products.

Conclusion

PanFS is The Most Comprehensive HPC Storage Solution

PanFS has deep roots in the HPC business. Panasas has developed and contributed many core aspects to the architectures and best practices of HPC storage over the years. PanFS' focus on reliability and low-touch administration is unique in an HPC-class storage system, and bridges the gap between traditional HPC that focused on performance to the detriment of reliability and the new reality of HPC moving into the Enterprise as a core service.

The term "scratch storage" that has traditionally been applied to HPC-class storage systems implies two core things: that the storage system is very fast, and that it is unreliable. With PanFS, you no longer have to choose, PanFS version 9 is both very fast and very reliable. It can store and serve the high-performance intermediate files of HPC compute jobs while at the same time storing and serving all the home directories and ancillary files of your user community, and it can do that with Enterprise-class stability, data reliability features, and ease of administration.

PanFS version 9 is the first HPC-class storage system that can meet all the needs of a high-performance environment.

09/2021

PANASAS®

2680 N. First St., Suite 150
San Jose, CA 95134

PH +1.888.PANASAS
F +1.408.215.6801
www.panasas.com