STORAGE DEVELOPER CONFERENCE

SDC 23

*BY Developers FOR Developers*

# Standards-based Parallel Global File System

## Automated Data Orchestration with NFS

David Flynn, Hammerspace Founder & CEO

# Overview

1. Why Parallel NFS now
2. How NFSv4.2 makes Parallel NFS enterprise NAS capable
3. Building a standards-based parallel global file system on NFSv4.2
4. Performance
5. Customers and use-cases
6. Q&A

# Why Parallel NFS is Relevant Now More Than Ever

## The Current Reality:

- Data orchestration is an absolute requirement across silos, sites, & clouds.
- High-performance requirements have gone mainstream.
- The world is moving to software-defined on commodity infrastructure.
- Linux is ubiquitous → enables a sophisticated, standards-based, open-source client to come built-in (not third-party).

## Therefore:

- NFS 4.2 solves these problems.
  - File access that bridges storage silos, sites & clouds.
  - Parallel file system with no need to install third-party client & management tools.
  - Avoids need to rewrite apps to use object storage.

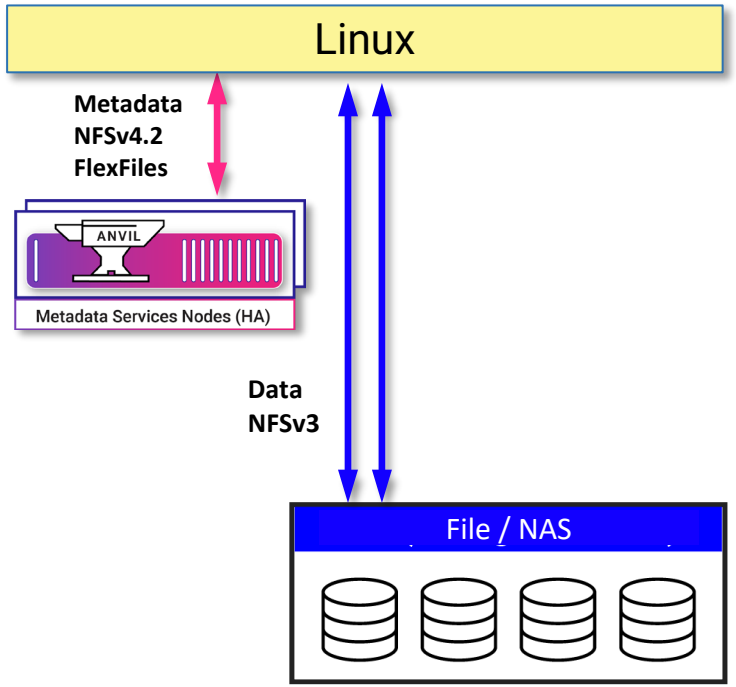
# NFS4.2 – NFS Enhancements and Fixes

- Elimination of excess protocol chatter using
  - Compound operations (versus serialized)
  - Caching and delegations (including client-side timestamp generation, eliminating need to go to the server)
  - This eliminates 80% of NFSv3's GETATTR traffic
  - File open / create is one single round trip to the metadata service (vs three serial round trips for NFSv3)
  - Subsequent open and read of a file just written is ZERO round trips (vs two serial round trips on NFSv3)
- Multiple parallel network connections between client and server and optional RDMA
  - Avoids TCP stack performance limitations
- Ability to write to multiple storage nodes synchronously (striping, mirroring)
  - To build highly reliable, highly available systems from unreliable storage nodes
  - To distribute even a single file access across multiple back-end NFSv3 storage nodes
- Ability to move data while it is live being accessed w/o interruption
- File-granular access / performance telemetry gathering and reporting
- Ability to serve SMB over NFS
  - Mapping of Active Directory principals and ACLs over the NFS protocol
  - SMB extended attributes carried over the NFS protocol (future)
  - Converged file range locking (future)

# Hammerspace Architecture Overview

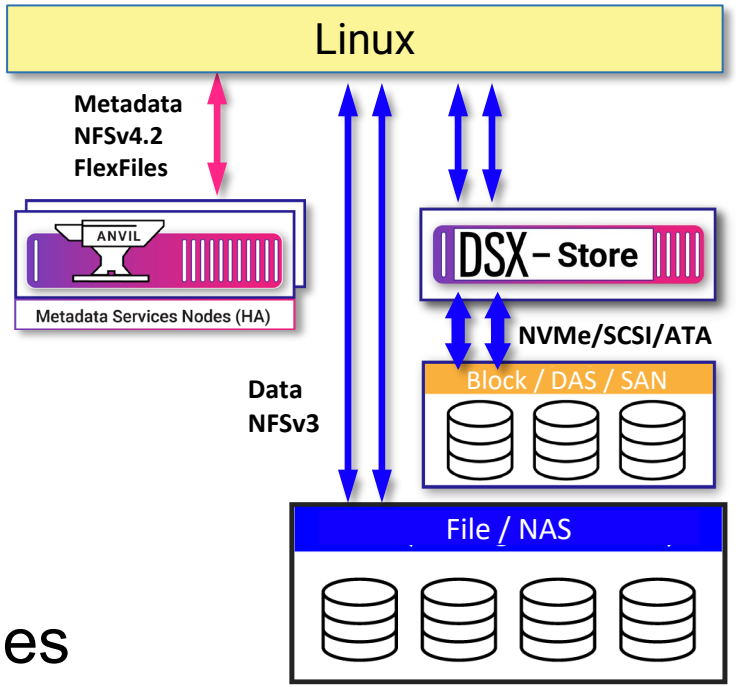
- Metadata
  - Hammerspace “Anvil”
  - Bare-metal, virtual, or container deployment
  - Synchronous replicated cluster for HA
  - Billions of inodes with millions active open
  - Full enterprise NAS data services
  - Instant data-in-place assimilation
- Client
  - NFS v4.2 in-box from RHEL 7.6 onward
- Data
  - Any NFS v3 NAS
  - Leverages NTAP, Isilon file clone APIs
  - Linear scalable data-path performance



# DSX – Store Function

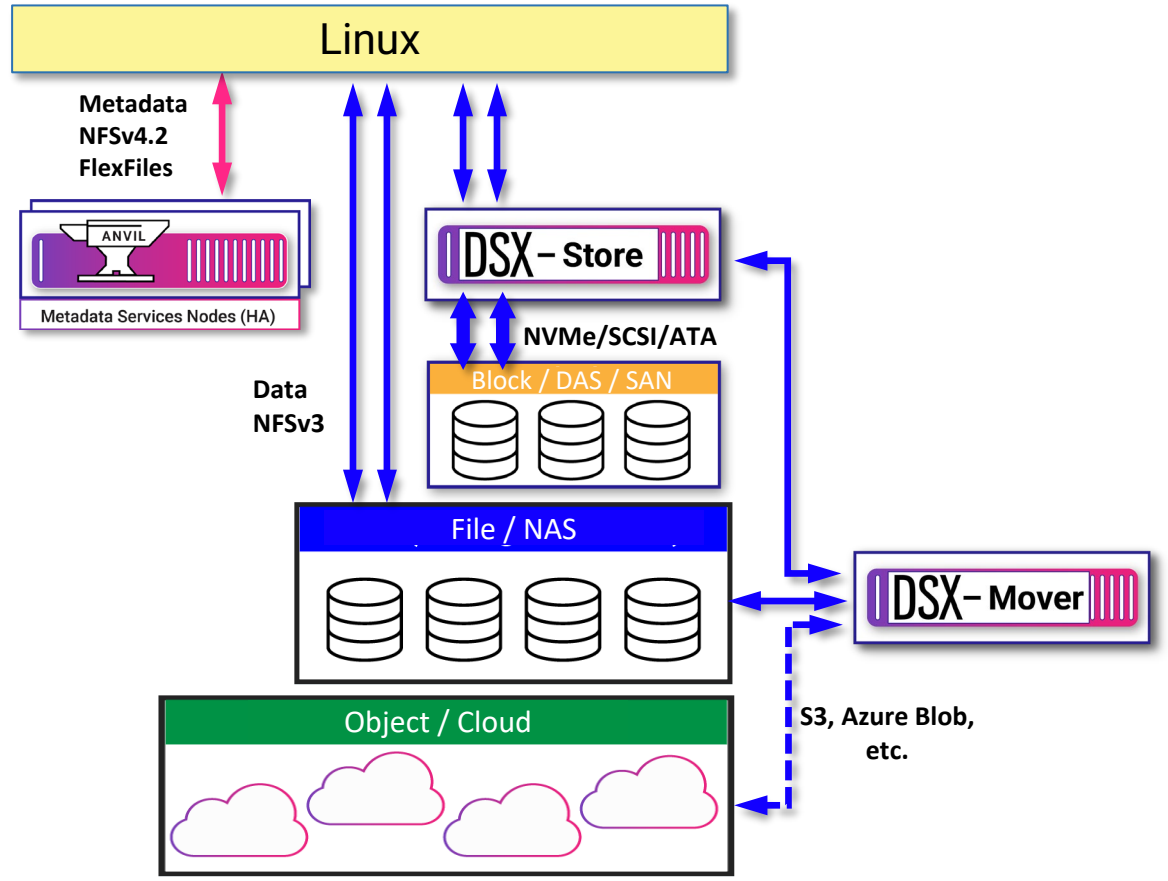
- Bare-metal, virtual or container deployment
- Parallel, linear scalable performance
- Sources any block storage
  - Direct attached
    - SSD , NVMe, HDD
    - Optional local striping and mirroring
  - Network attached
    - SAN, iSCSI, EBS
- Supports share snapshots and file clone
- Client can mirror writes to multiple DSX nodes
- Or use erasure encoded groups of DSX nodes



# DSX – Mover / Cloud Mover Function

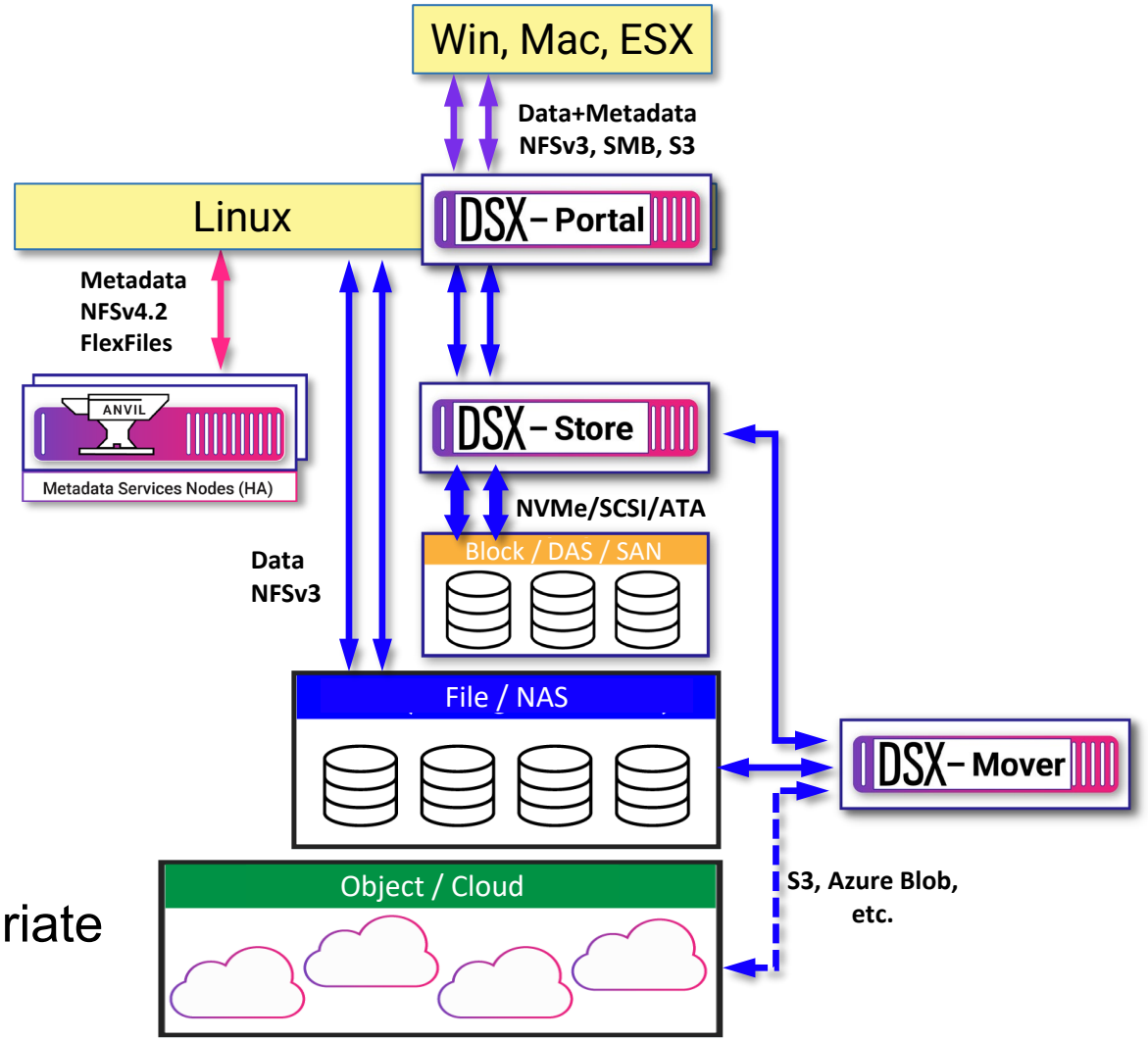
- Bare-metal, virtual or container deployment
- Parallel, linear scalable performance
- Stateless, scale-out
- Fully automatic scheduling
- File to file mobility
  - NFSv3
  - No interruption to ongoing access
- File to object mobility
  - S3, Azure Blob, etc. over HTTPs
  - Global dedupe, compression, encryption
  - Transfer & egress optimized



# DSX – Portal Function – Legacy Client Support

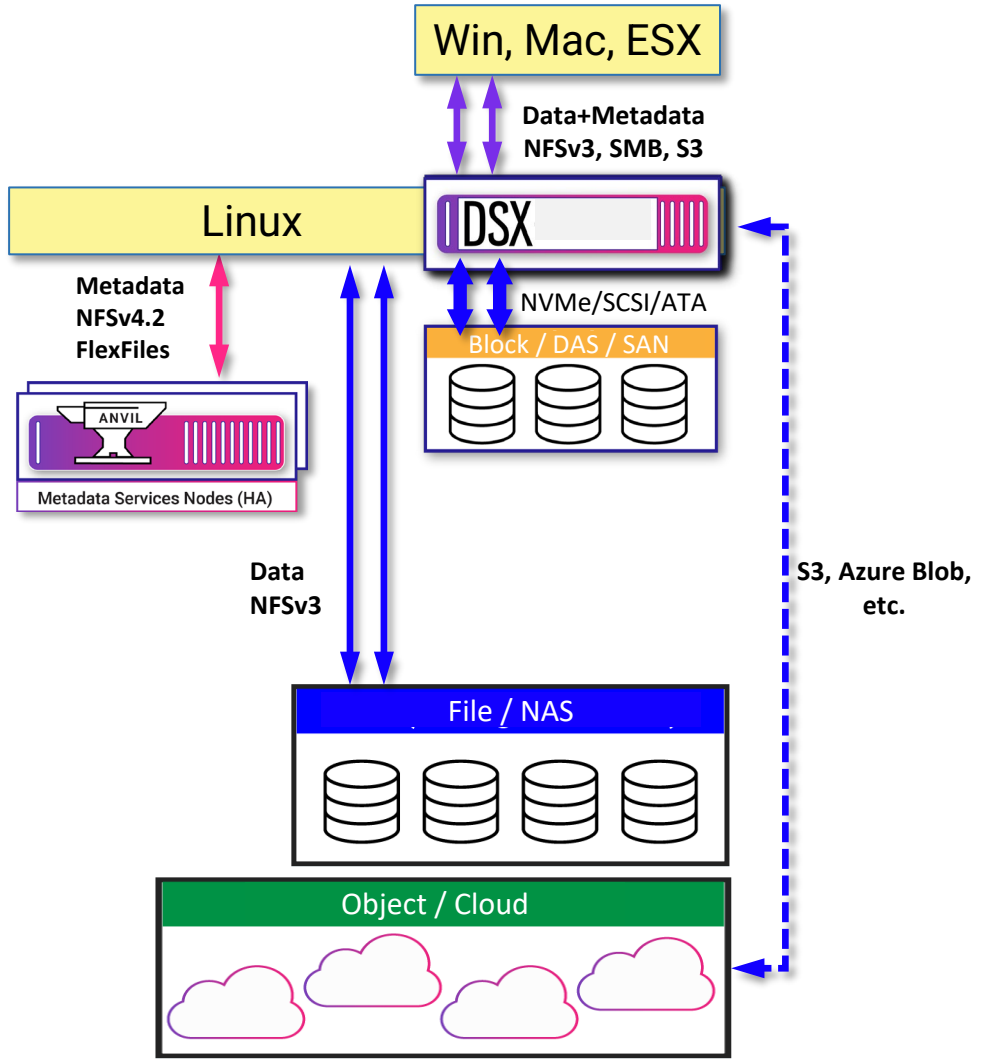
- Bare-metal, virtual or container deployment
- Parallel, linear scalable performance
- Stateless, scale-out
- Virtual IPs with fail-over
- NFS v3, SMB 2.x/3 and S3
- Global file locking
- Extensive Caching
  - Metadata
  - Read data
  - Write-back and write-through caching as appropriate



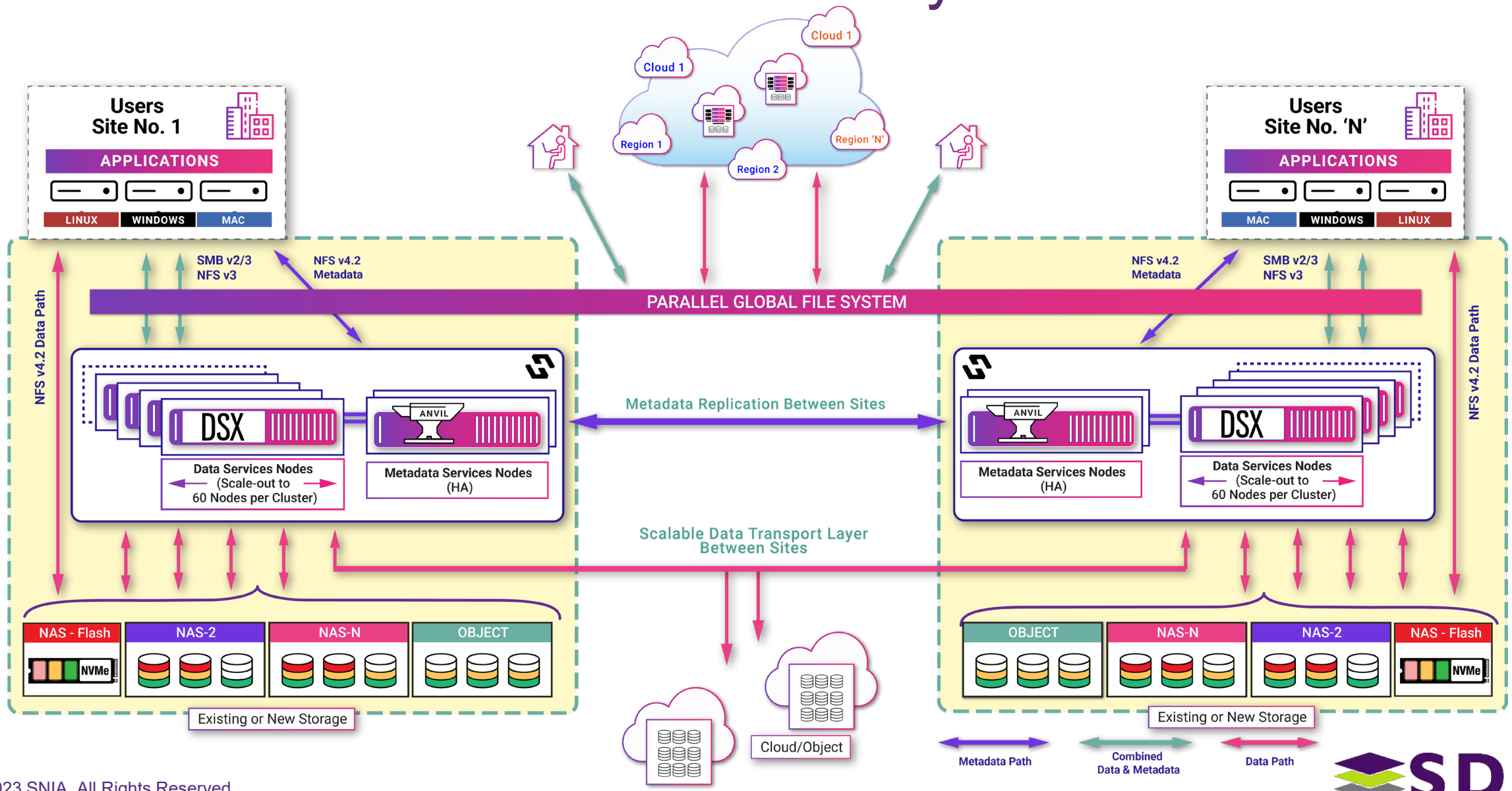
# DSX – Containerized Microservices

- Deployment flexibility
  - Co-resident on client nodes (hyper-converged)
  - Dedicated storage-only nodes
- Eliminates networking hops
  - Port, cost and latency reduction
- Bypasses serialization over NFS
  - IO short-circuits in the kernel
- Achieves full NVMe performance
  - Tens of Gbytes per second
  - Millions of IOPS
  - Microsecond latency



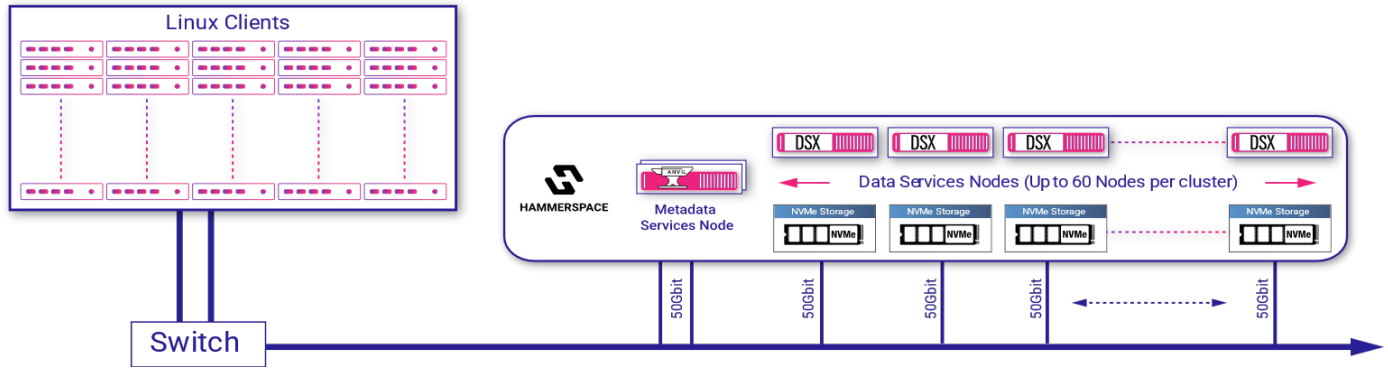
# Unstructured Data Orchestration System in Action



# Example: Linear Scalability Saturating Infrastructure

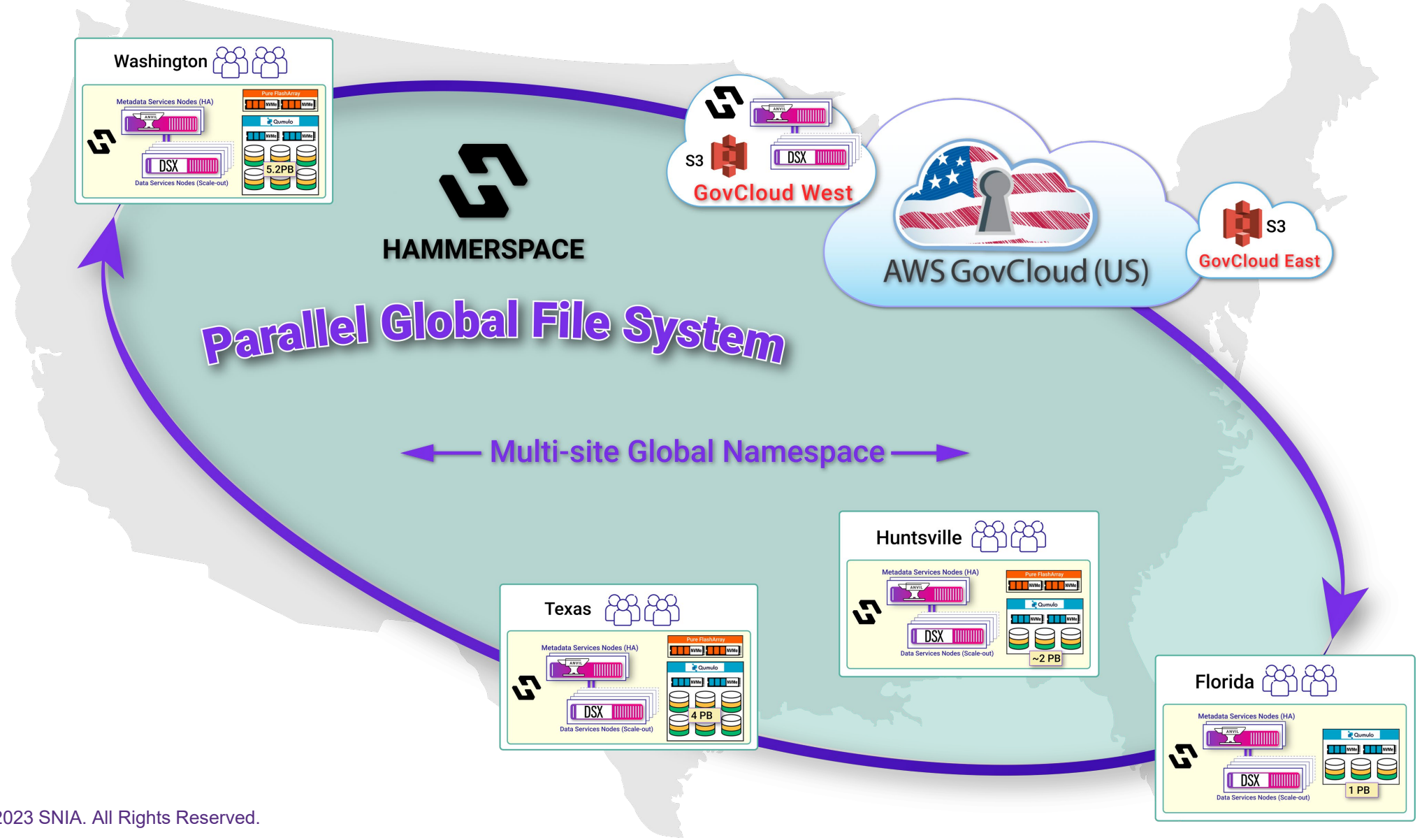
- Performance testing showed **linearly scale** from small to large:
  - Saturating the network for throughput-dependent workloads.
  - And saturating the backend storage for IOPS-dependent workloads.
- Testing showed **16 DSX nodes hit 1.17 Tbits/s** with 32kb file sizes with low CPU overhead.
- In testing for raw IOPS with this configuration, the same test using small 4k files achieved 6.17m IOPS.
- Testing showed linear scalability to limits of network and storage, by adding more nodes.




- **Test Suite:**
  - 192 clients
  - 16 DSX Nodes
    - Can scale to 500 DSX nodes per cluster x 16 clusters.
    - DSX Nodes can be mixed instance types.
- I/O Pattern Randomized – 90/10 R/W mix
- NFS Exports were mounted with NFS 4.2

# Data Orchestration Powering Space Exploration



# Summary

- NFS 4.2 solves global high-performance file access
- Flexfiles Layouts provide flexibility to bridge block, file & object at scale, globally
- Enables transparent live data mobility
- Supports software-defined commodity model
- Leverages existing ubiquitous NFS client
  - No third-party client required.
- Supports extreme scale-out & high-performance file workflows across silos, sites, & clouds

STORAGE DEVELOPER CONFERENCE

SDC 23

*BY Developers FOR Developers*

# Questions?