Flash Memory Summit

Introducing the Need for NFS-SSD

Presenter: David Flynn, Hammerspace CEO and Founder

Flash Memory Summit

Primer: Standards-based Parallel File System and Data Orchestration

Why Parallel NFS is Relevant Now More Than Ever

The Current Reality:

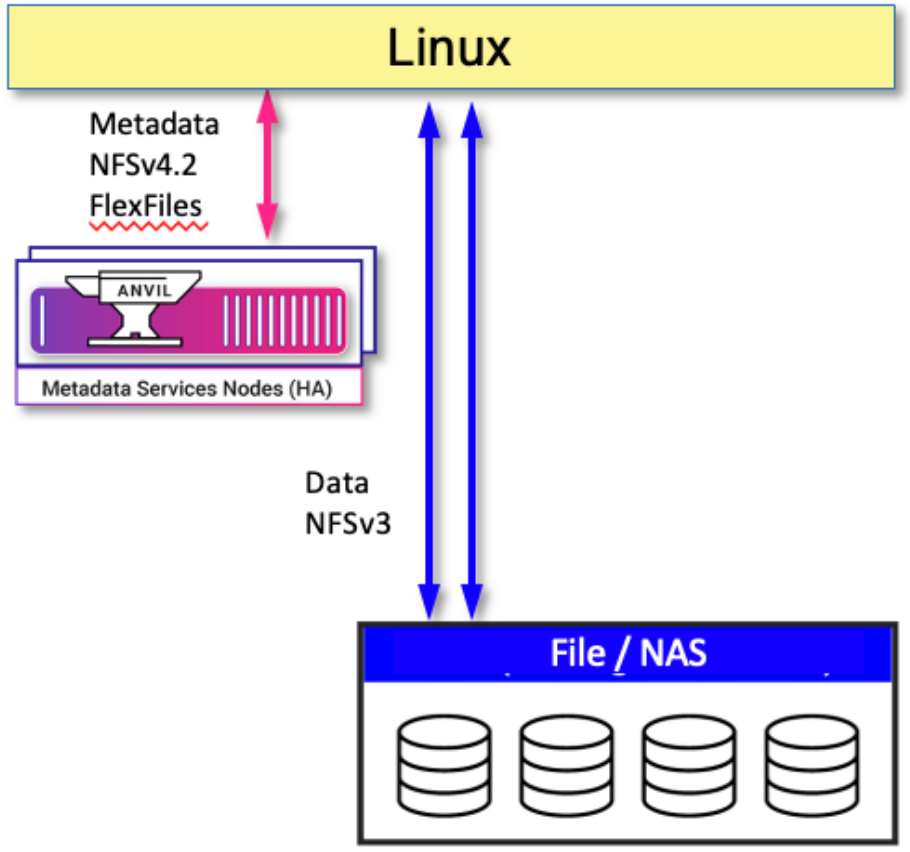
- Data orchestration is an absolute requirement across silos, sites, & clouds.
- High-performance requirements have gone mainstream.
- The world is moving to software-defined on commodity infrastructure.
- Linux is ubiquitous → enables a sophisticated, standards-based, open-source client to come built-in (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.

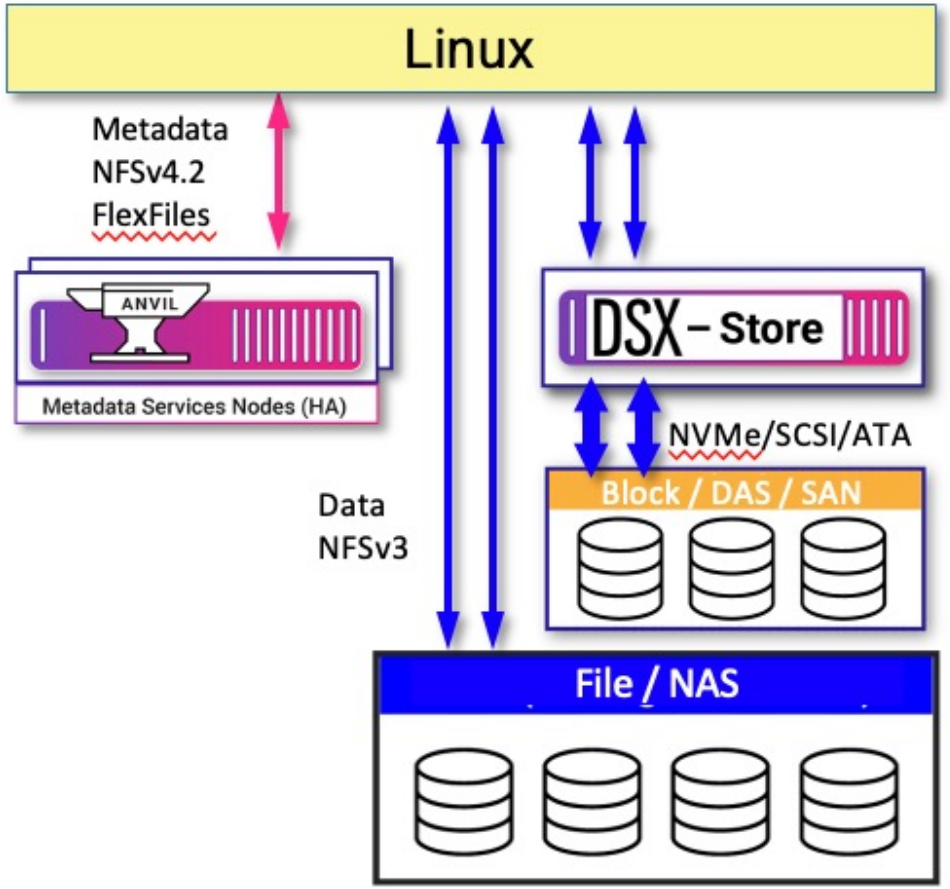
Hammerspace Architecture Overview

- 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



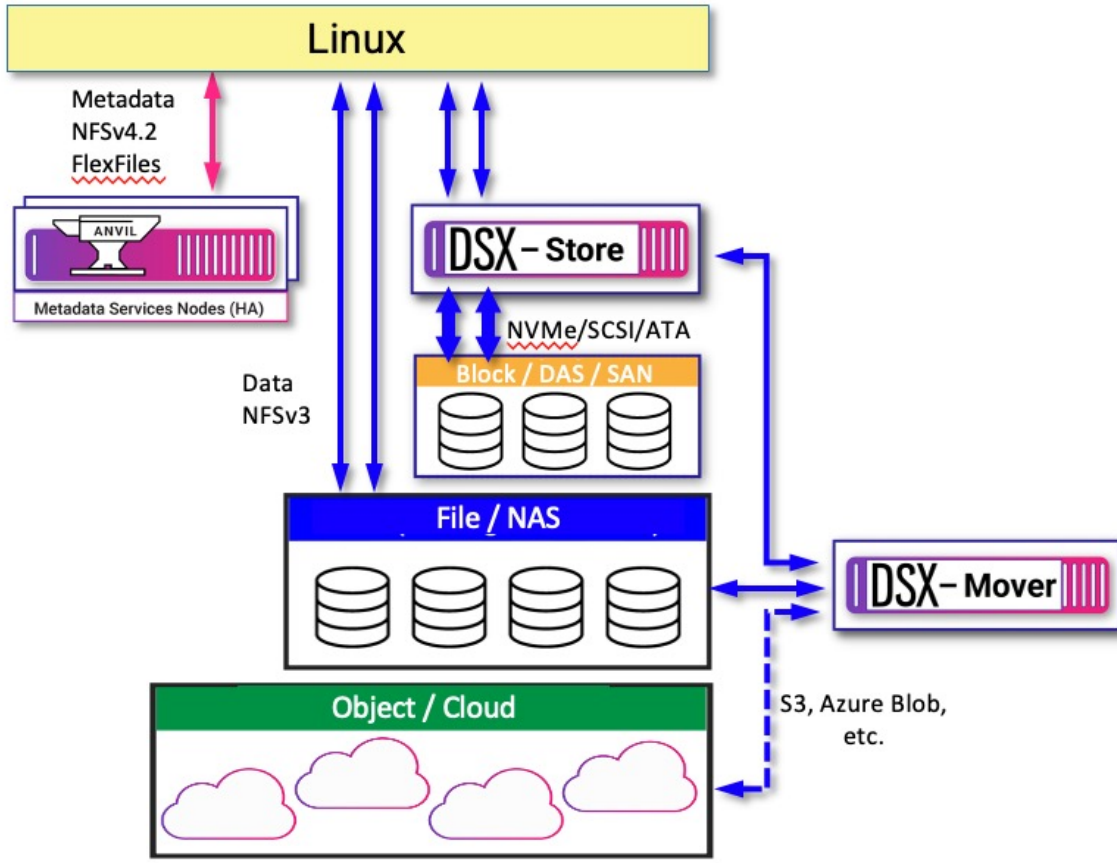
DSX – Store Function

- 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 clones
- Client can mirror writes to multiple DSX nodes
- Or use erasure encoded groups of DSX nodes



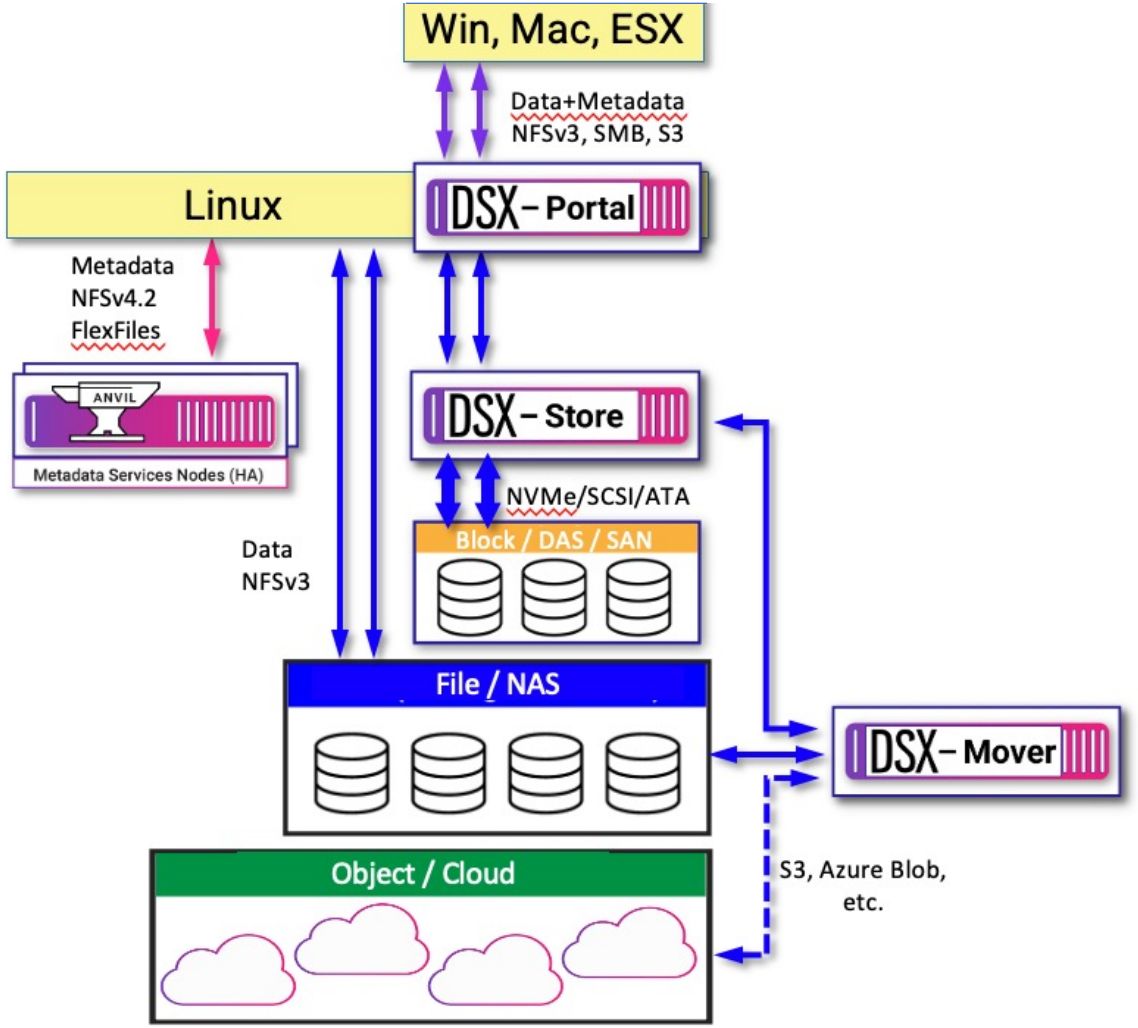
DSX – Mover / Cloud Mover Function

- 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



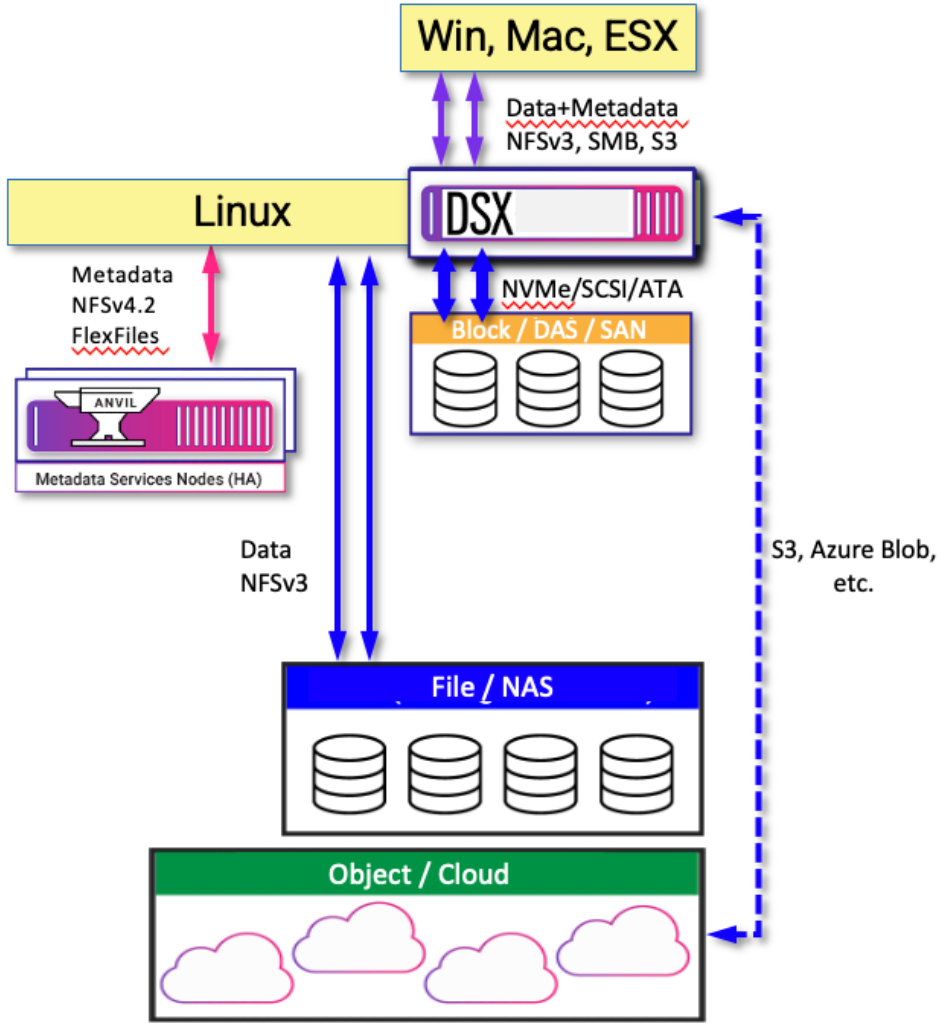
DSX – Portal Function – Legacy Client Support

- 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

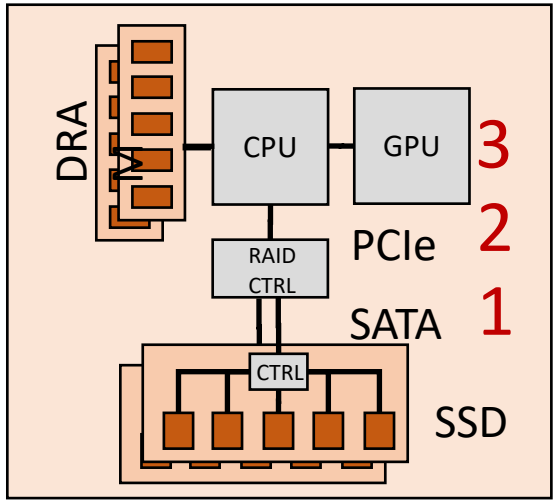


NFS4.2 – Target Advantage Areas

- NFS 4.2 - Sweet Spots:
  - Scale-out distributed high-performance file-based workloads.
  - Stateful file access at scale globally across block, file, & object.
  - No client software required – included in standard Linux distributions.
  - Runs on commodity hardware.
  - Supports any on-prem or cloud storage of all types from any vendor.
- With Hammerspace:
  - Supports decentralized environments:
    - Global file system spanning silos & sites
  - Actionable metadata, including custom metadata driving objective-based policies across any storage type and location.

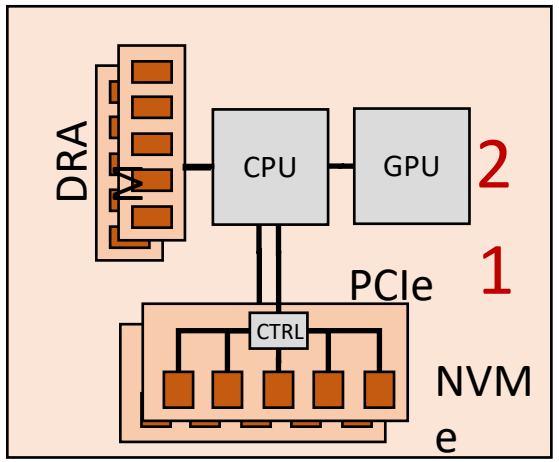
Benefits of Embedding NFS in eSSDs

Direct Attached Storage



The RAID controller is the bottleneck and adds an additional serial data retransmission.

Direct Attached Storage: NVMe

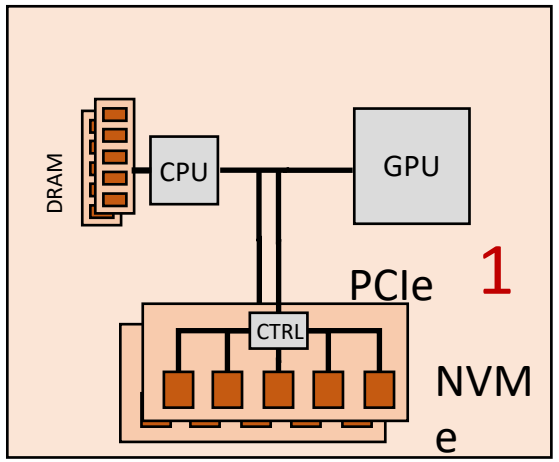


NVMe eliminates the RAID controller

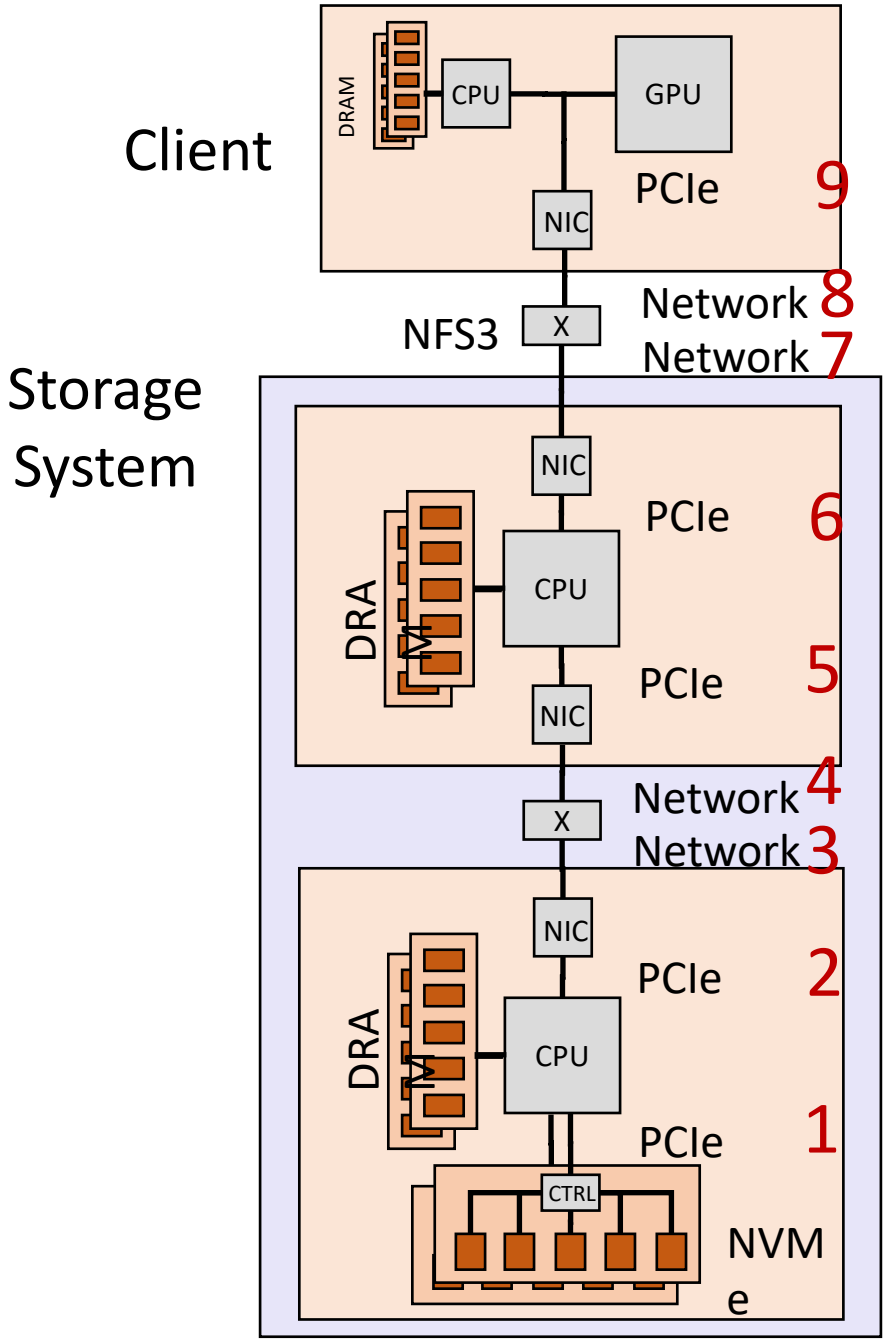
Direct Attached Storage: NVMe and GPU Direct

GPU Direct eliminates the host CPU and memory

But, what about with shared storage?



- DENTRY to INODE mapping
- File offset to block mapping
- Block to flash address mapping



Network Attached Storage

(e.g. NetApp, Isilon, Pure, Qumulo, Ceph)

DENTRY to INODE mapping

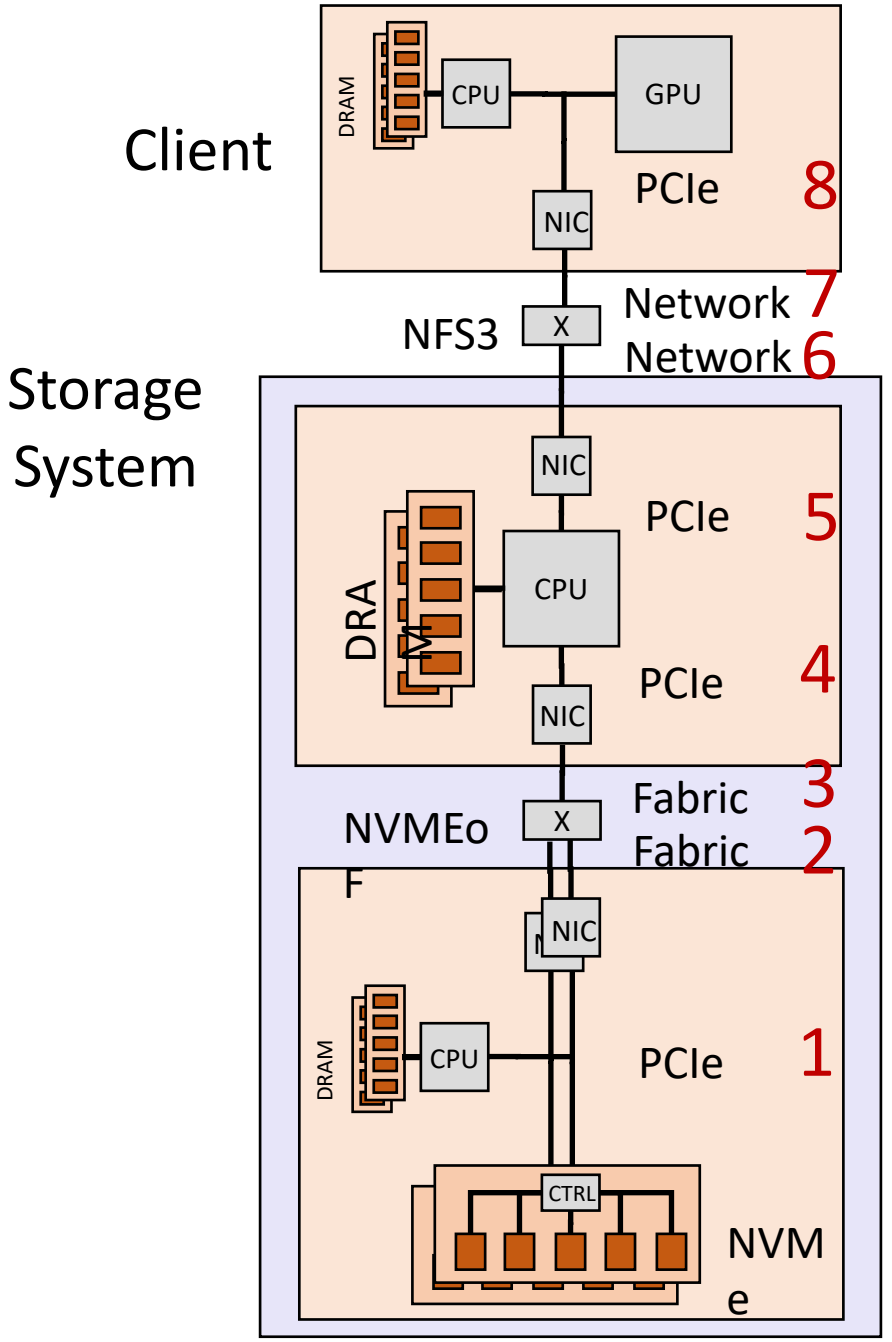
File offset to block mapping

The storage back-end host (CPU and memory) is the first bottleneck.

Block to flash address mapping

Network Attached Storage

(using NVMEoF, e.g. Vast, Weka)



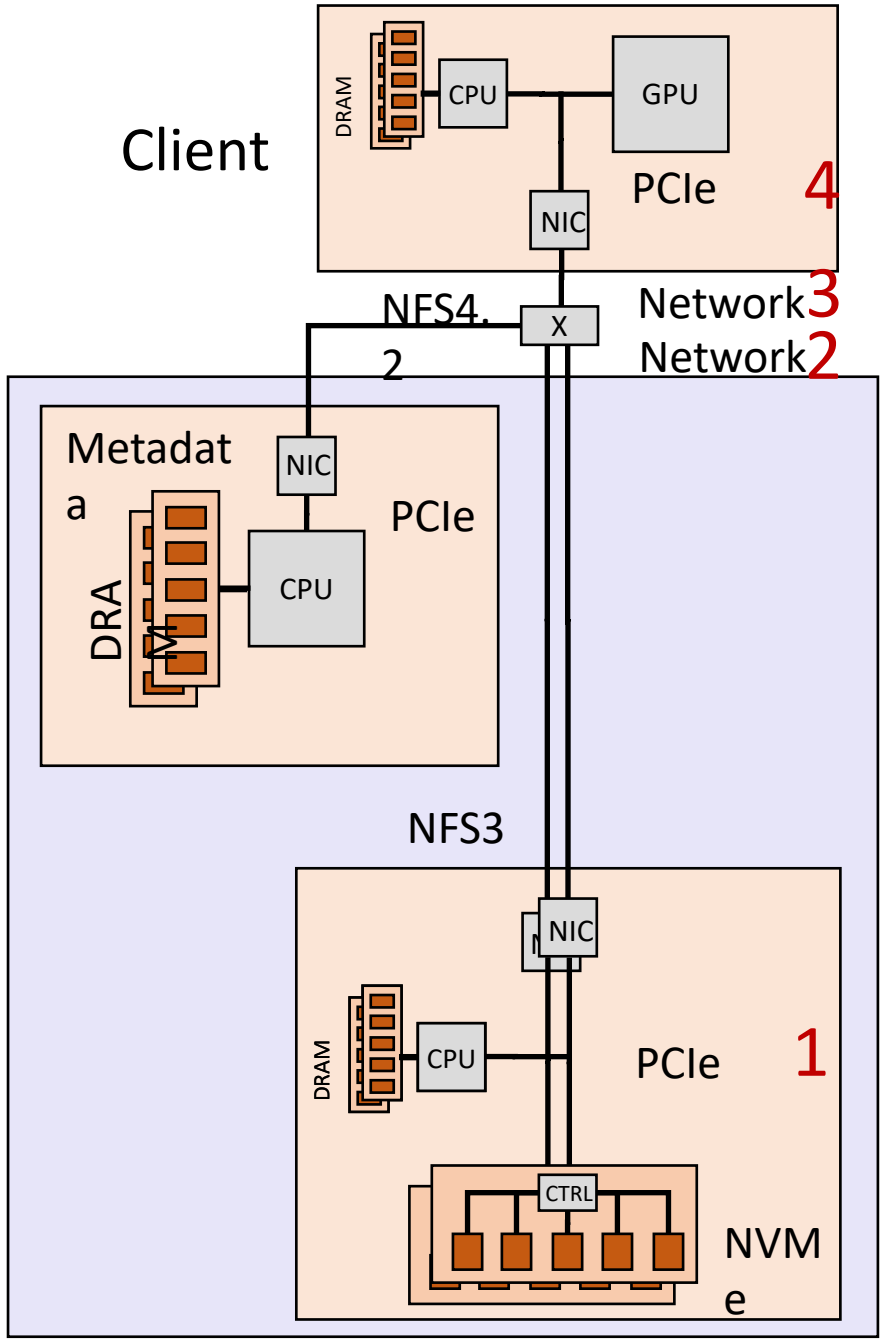
DENTRY to INODE mapping

File offset to block mapping

Block to flash address mapping

The file server front end is an even bigger bottleneck.

Network Attached Storage
(using NFS4.2, e.g. Hammerspace)

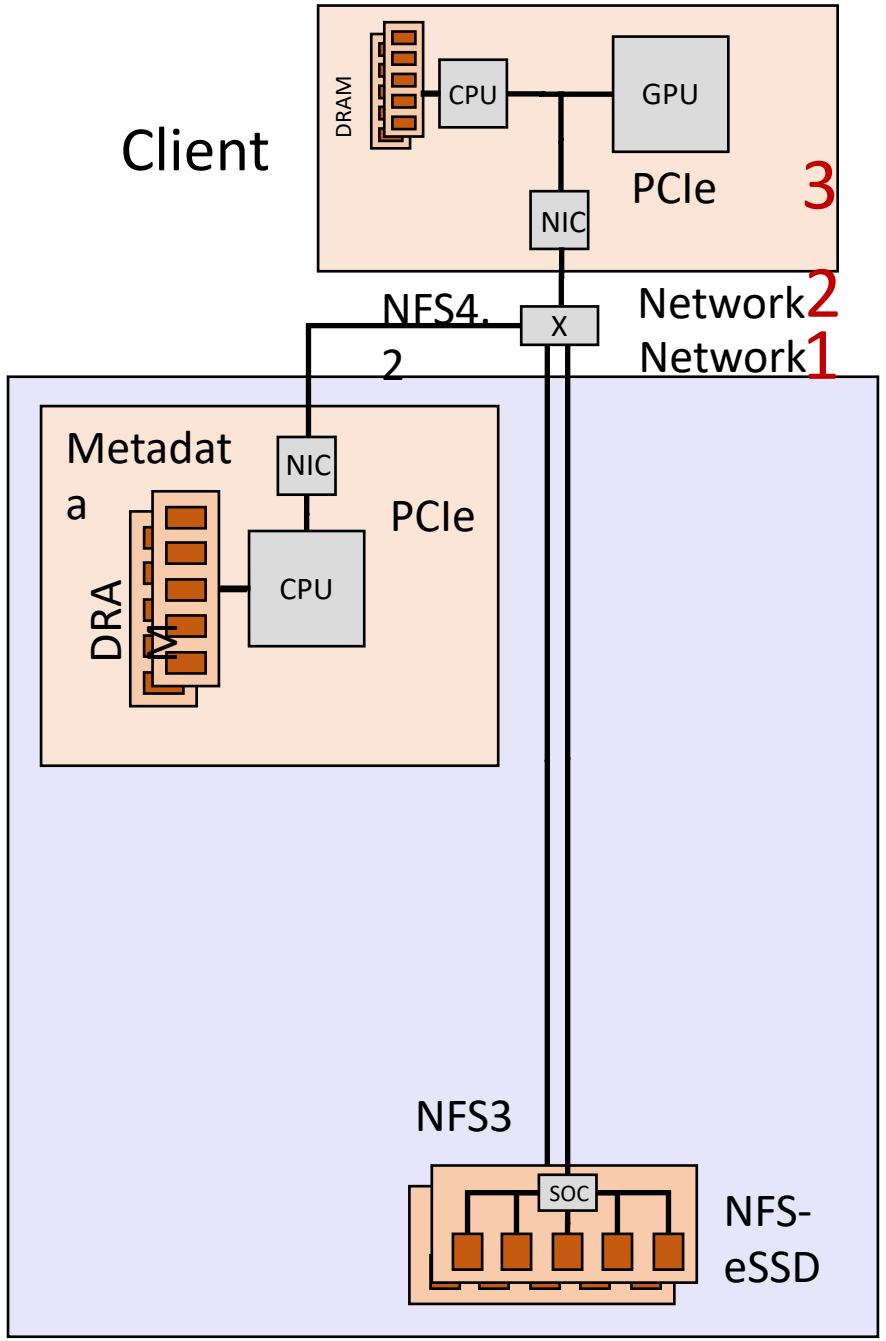


NFS4.2 has no bottlenecks, eliminates 4 of 9 data retransmissions, and doesn't need NVMEoF – or even an internal network!

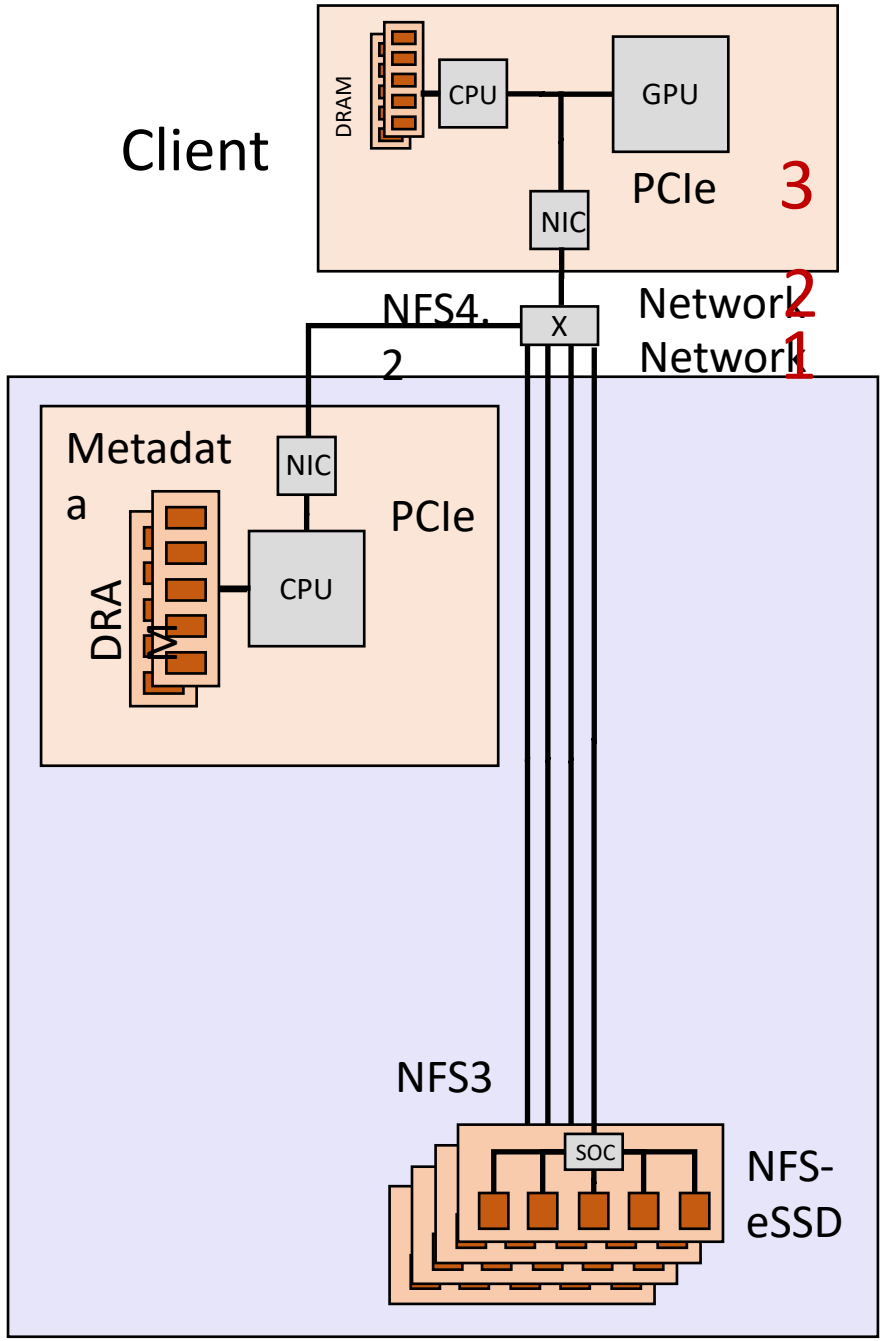
File offset to block mapping

Block to flash address mapping

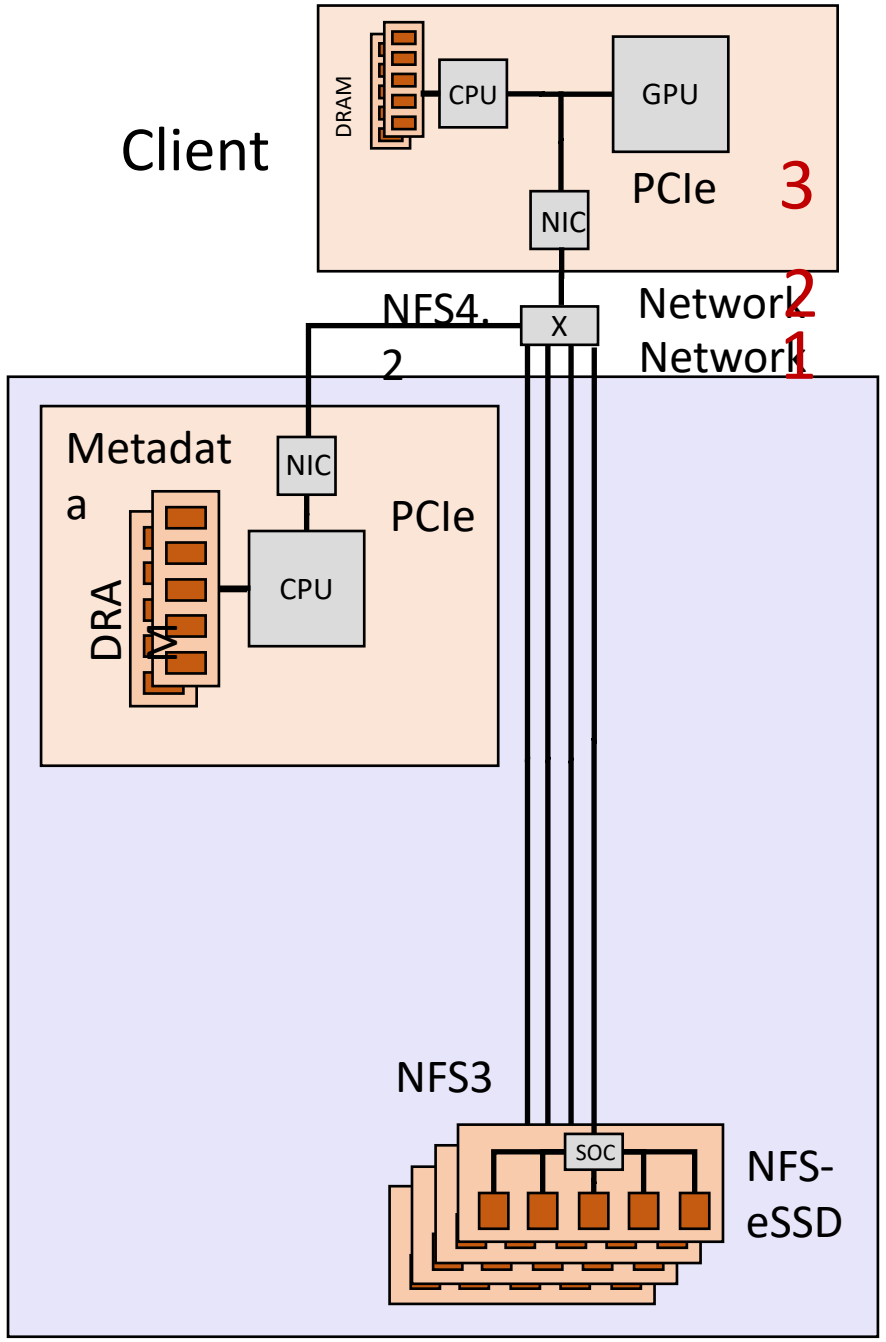
Network Attached Storage
(using NFS4.2 and NFS-eSSD)



Network Attached Storage
(using NFS4.2 and NFS-eSSD)



NFS4.2 with proposed NFS-eSSDs eliminates 6 of 9 data retransmissions, eliminates the double mapping layers, and scales 1x1 with network ports!



Benefits

Lower latency

Lower power consumption

Lower operational (and capital) costs

Lower write amplification

Higher density without sacrifice of potential performance

Higher access density

Better inherent reliability, availability and serviceability

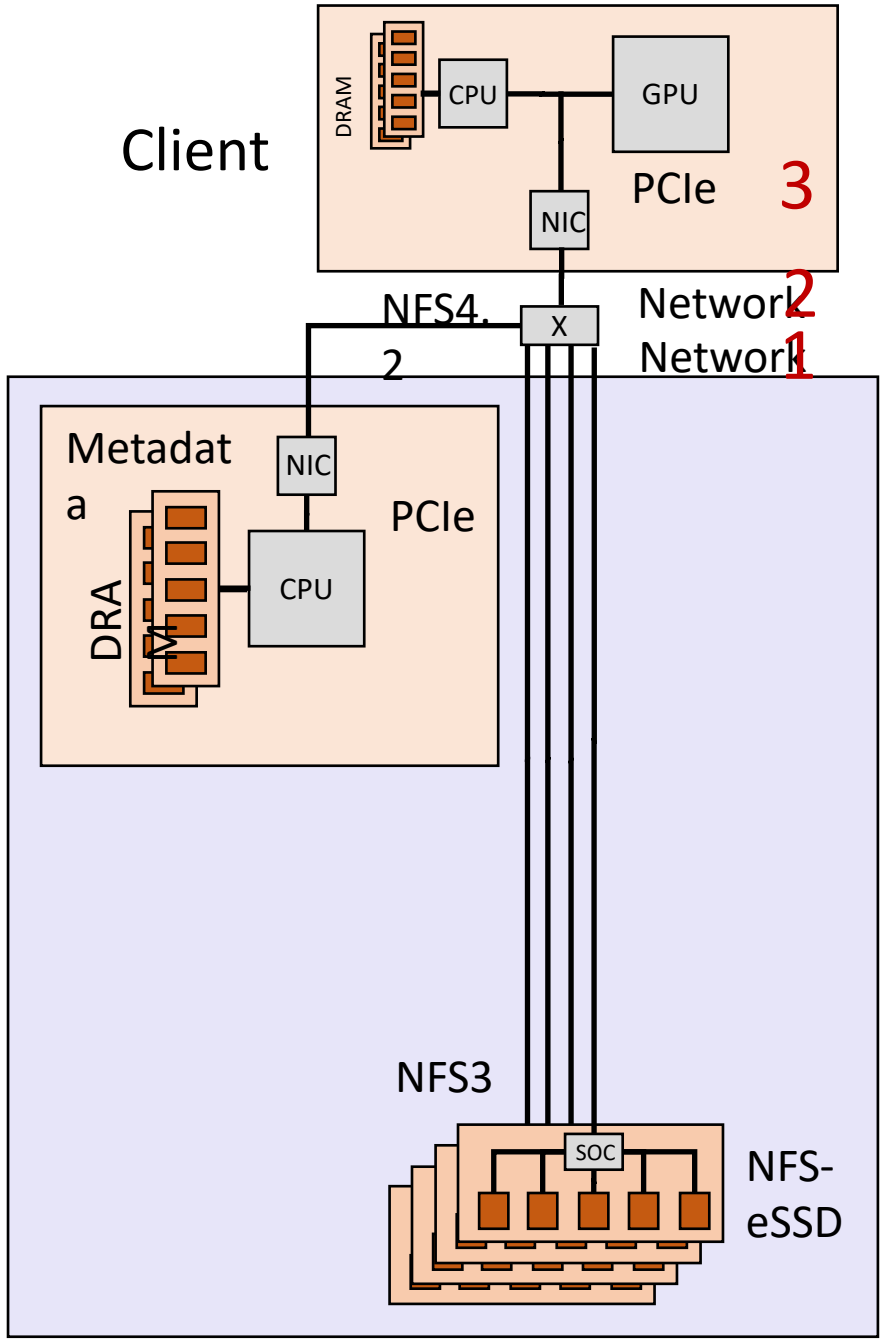
Much wider dynamic range of scale

- Scale up (hyperscale)
- Scale down (SOHO, maybe on USB-C)

Enables Computational Storage

- Compression, deduplication, encryption, erasure / error coding, copy / clone, filter, search, join, map reduce, etc. can be offloaded to the SSD now that it understands file layout

Why Now



AI/ML workloads demanding efficient performance

Data governance / cloud computing needs orchestration

Flash performance can easily saturate PCIe/Ethernet

E1.S and other form factors (density and power)

64-bit processor IP availability

Processor performance density

IPv6, RoCE

Embedded Linux with

High performance, lightweight filesystems (XFS)

High performance, lightweight NFS server (kNFSd)

Standardized Parallel NFS 4.2 Flexible Files

Thank You!

David.Flynn@Hammerspace.com