*Isilon
Intro - John Har
15 years of scale out leadership
>3.2 Exabytes shipped calendar 2016

Performance & Scale - Flash, Hybrid, Archive
Enterprise Grade - protection, security, compliance
Seamless Cloud - public, private, hosted
Analytics - deliver new insights
Lowest TCO - automated tiering

Isilons and OneFS
Single, scalable file system
Fully symmetric, clustered architecture
Truly multi-protocol data lake
Transparent tiering with heterogeneous clusters
Non-disruptive platform and OneFS upgrades

Industry Challenges
EDA - 7nm and 3D Chip designs
Life Sciences - population-scale genomics
M+E - 4K Content and Distribution
Enterprise - big data and analytics

Start small and scale
72TB - 924TB in 4RU
Scale to over 33PB in a single file system

Performance
250K IOPS, 15GB/s in just 4RU
Scale to 9M IOPS

Isilon F800 Architecture (Steve Soumpholphakdy)

Design Goals
Higher MTTDL
Smaller Fault Domains
- Fewer drives/node and nodes/pool
- Enables use of larger drives

Higher Serviceability
- Mirrored journal
- Boot from data media (no dedicated boot drives)

Higher Performance and performance density
- Higher compute/disk ratios
- Larger journals
- More CPUs per rack unit

Design Goals (2)
Higher disk density

- Isilon F800 All-Flash has 15 x 2.5” SSDs per RU

Industry standard components and trends

- Ethernet Backend
- Vault-based card-less NVRAM
- 3 write-per-day SSDs
- 4Kn drives

And

- No SPoF
- Forward looking

Isilon F800 All-Flash Hardware Chassis
Chassis is comprised of 2 node pairs
- Total of 4 nodes per chassis

Each node has its own power supply, but shares a power region with its peer
- 4 x 1450W power supplies
- Battery backup for journal vaulting

Drive Sleds
5 sleds per node
- 20 sleds total per chassis

3x 2.5” SSDs per sled
- Total of 15 x 2.5” SSDs per node
- Choice of 1.6TB, 3.2TB or 15.4TB SSDs

File system layout is sled-aware
- A given file uses one drive per sled
- allows sled removal for service without data unavailability, treated as temporarily-offline drives

Future-proofed for future flash media

Compute
CPU

- Intel Xeon E5-2697A v4 (Broadwell)
- 16 cores, 32 threads
- 2.6GHz

DRAM

- 4x 64GB DDR4

Modular design for future upgradeability and investment protection

Network Fabrics
Independent frontend and backend fabrics
Frontend
- 10 Gb/40Gb Ethernet

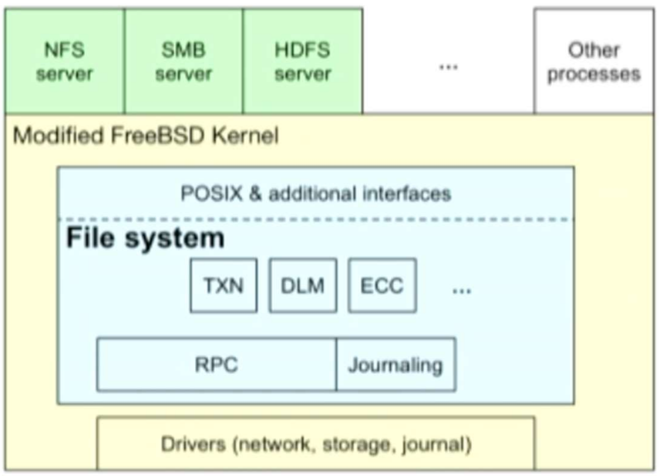
Backend

- Ethernet (40Gbs Ethernet new clusters, Isilon-provided switch)
- InfiniBand (InfiniBand (QDR) to join legacy clusters)

Met Design Goals

Isilon OneFS 8.1 Enhancements for Flash (Anton Rang)

OneFS Internals
Modified FreeBSD userspace & kernel
- Actively contributing improvements to FreeBSD (where it makes sense)
Protocol heads (NFS, SMB, etc) in userpsace
- Data normally kept within kernel via zero-copy system calls
File system in kernel
- POSIX interfaces + many enhancements (ReadDirPlus, file name encoding, range-based sync, SMB ACLs, NFS attribute returns, etc)
- Cross-node RPC (built on SDP over InfinBand, TCP over Ethernet
- Distributed Lock Manager
- Transactional disk updates (physical block journaling, large journal in main memory)



Max filesize 4TB

Optimizing for Flash Storage
Requirements to address
CPU Cycles
Memory bandwidth & latency
Network bandwidth & latency
“Disk” subsystem
- Hard to build deep enough queues to take full advantage of SSD
- Alternating reads and writes too frequently leads to poor performance
- Sorting operations can be harmful

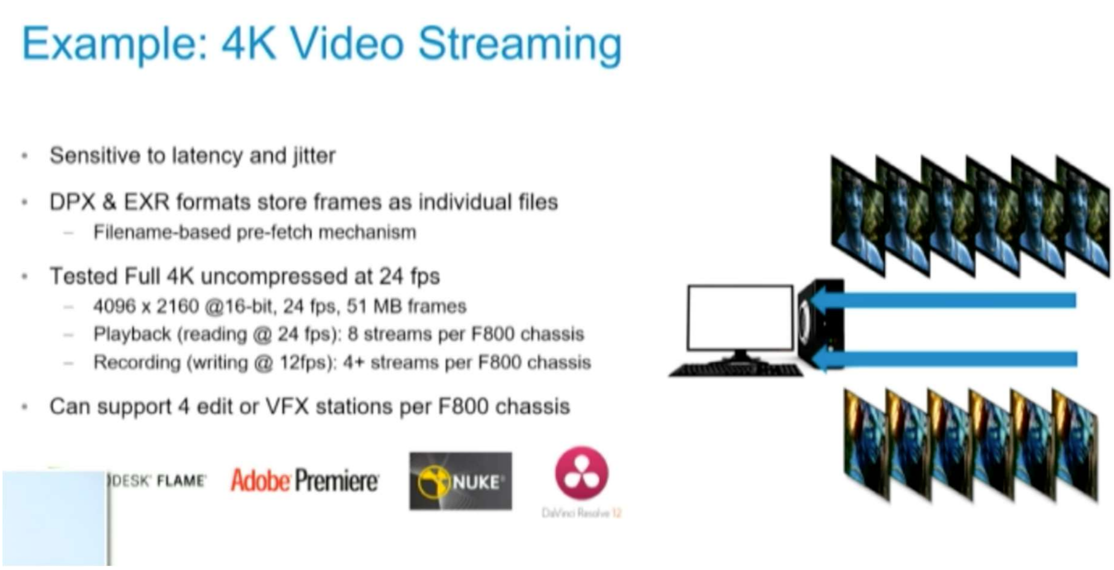
OS-induced latency
Lock Contention
- More CPU cores, needed to drive flash storage, increases contention
- Contention becomes worse with faster storage (primarily for reads, since writes are journaled)

Isilon OneFS 8.1
Optimised CPU usage
- improved profiling tools

- FreeBSD kernel optimisations, file system optimisations, userspace optimisations
Improved network stack
- adopted DC-TCP (Data centre TCP) with explicit congestion notification (ECN)
- introduced higher-frequency timers, and new algorithms for retransmission for Ethernet incast mitigation
- hold locks for shorter intervals by restructuring transmission code
Rewrote OneFS I/O scheduler
- grouped reads and writes more aggressively
- adapted behaviour dynamically based on pre-drive measured latencies
Improved journal
- latency improved 50%

Examples of CPU Optimizations
FreeBSD
- rewrite of TLB invalidation to scale better and run more quickly
- rewrote various primitive functions (memcpy, copyin, etc) to use new or improved instructions
- implemented multi-threaded buffer cache daemon
- added interrupt balancing
File system
- Rewrote drive selection code to reduce CPU usage during writes
- Merged related data structures to reduce TLB misses
- Used unmapped buffers in more places (reduce KVA management overhead and TLB shoot-downs)
- Reduced lock contention / spin time in the transaction manager

Aggregate Throughput
15.4GB/s read per chassis
7.7GB/s write per chassis, 1.9GB/s per node



Example: Streaming Analytics with Kafka

- Multiple producers of data streams, with multiple consumers processing streams in real-time
- Saturates CPU on 16 cores on 8 compute nodes with one Isilon F800
- Supports > 1M transactions per second on one Isilon F800
- Just 21 ms average latency during benchmarks

Single Nitro Chassis (4 Nodes) + Eight 2 core Compute Nodes