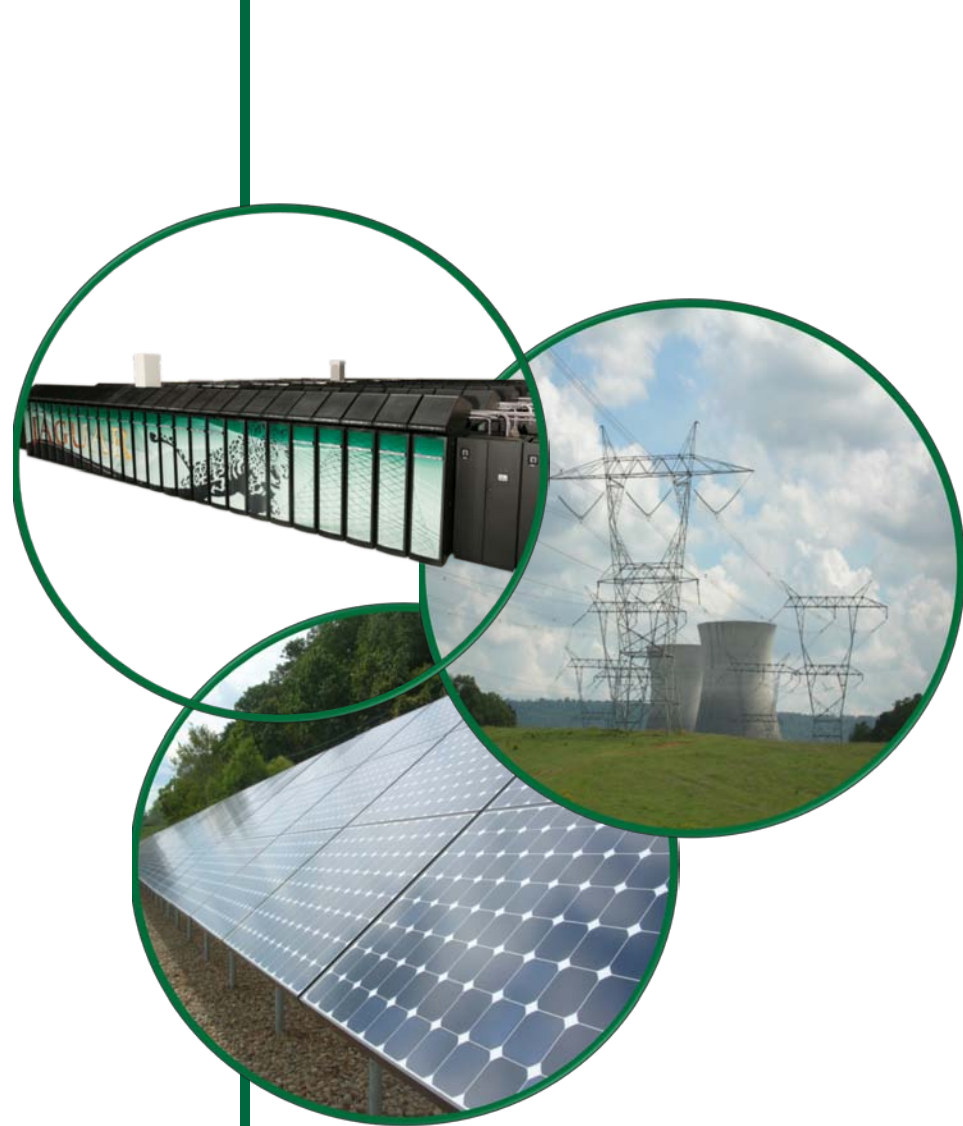


# The Spider Center-Wide File System

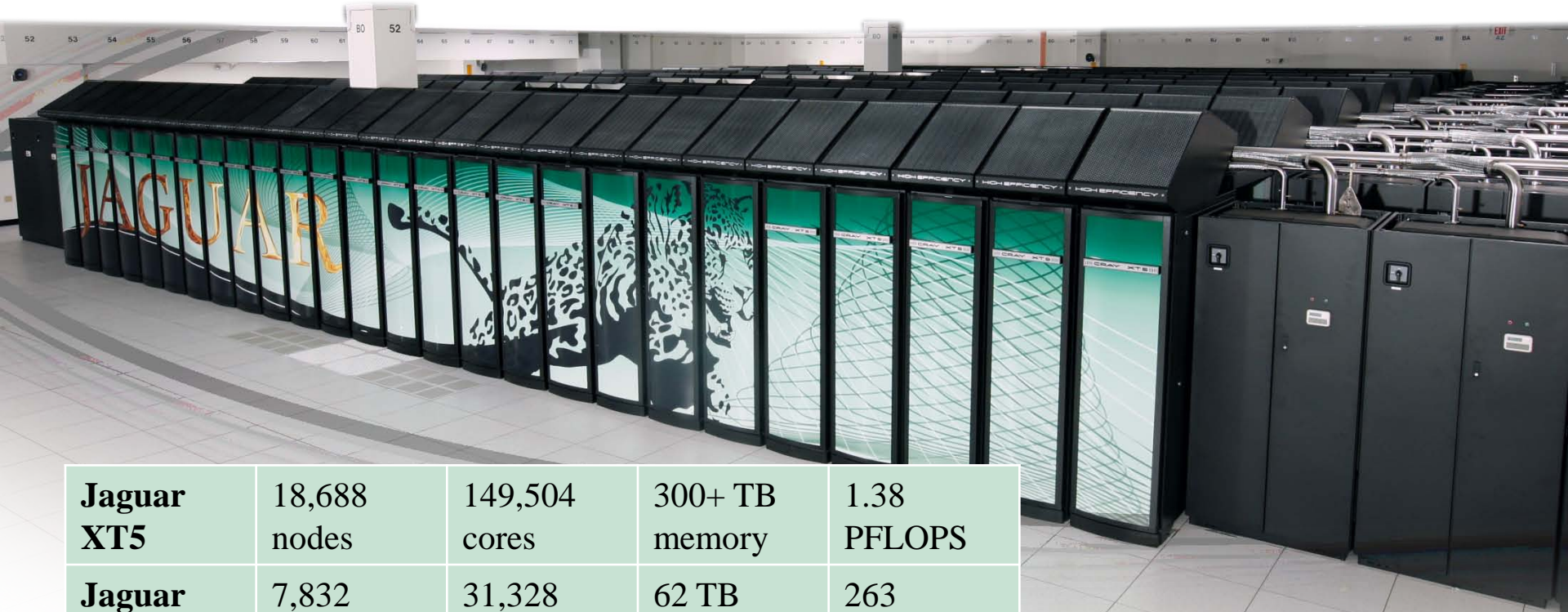
**Presented by Feiyi Wang (Ph.D.)**  
Technology Integration Group  
National Center of Computational Sciences

Galen Shipman (Group Lead)  
Dave Dillow, Sarp Oral, James Simmons,  
Ross Miller, Jason Hill

HPC Advisory Workshop  
Changsha, China, October 28, 2009



# National Center for Computational Sciences

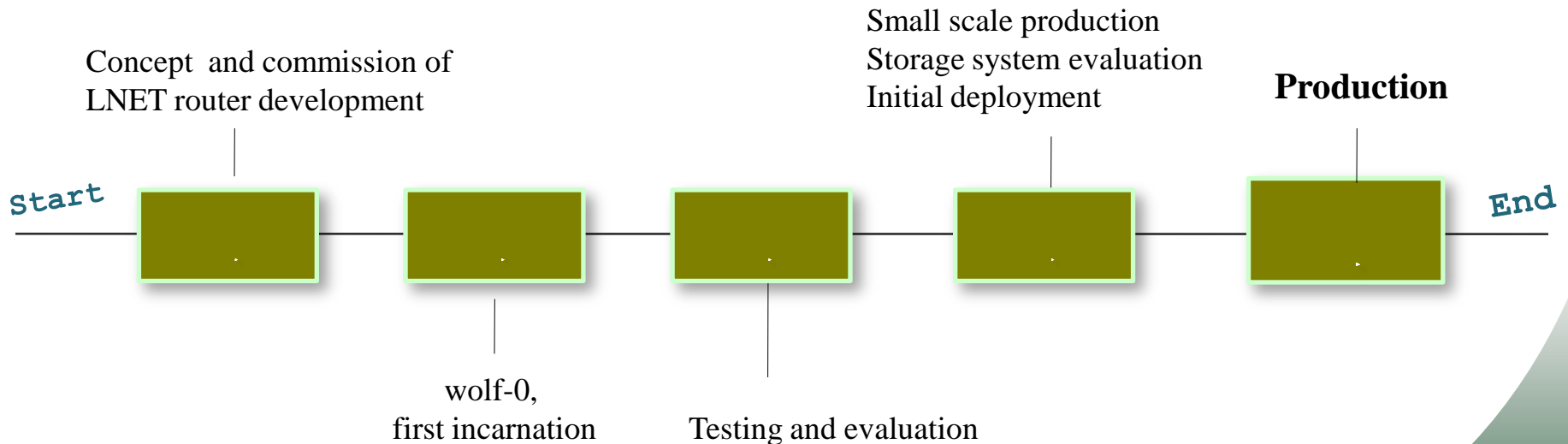


<b>Jaguar XT5</b>	18,688 nodes	149,504 cores	300+ TB memory	1.38 PFLOPS
<b>Jaguar XT4</b>	7,832 nodes	31,328 cores	62 TB memory	263 TFLOPS
<b>Lens</b>	Linux cluster for data analysis and visualization			
<b>Smoky</b>	Development cluster			
<b>Eugene</b>	IBM Blue Gene/P			
<b>HPSS</b>	25 PB mass storage system			

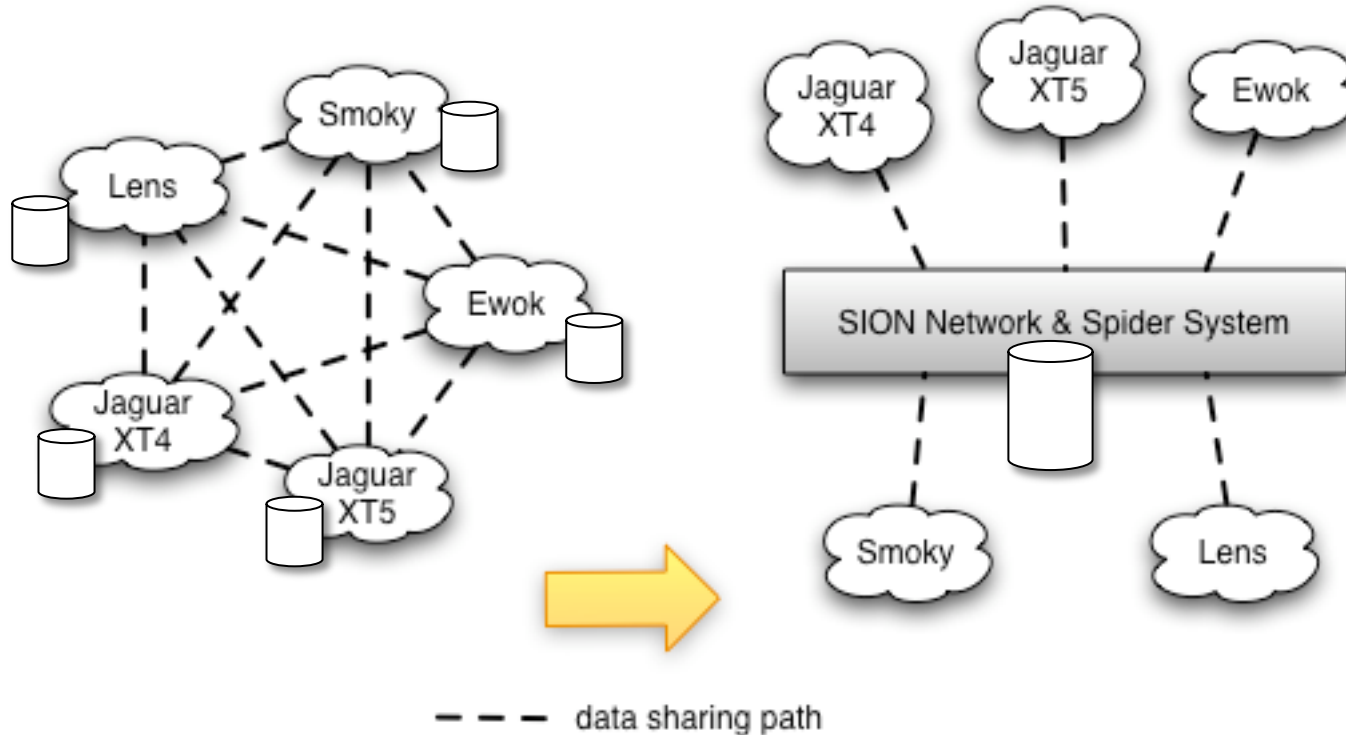
XT5 Hex-core upgrade:  
224,256 compute cores

# What is Spider?

- Spider is a center-wide file system at NCCS
- Spider is
  - the world's *largest Lustre* file system
  - the world's *fastest Lustre* file system
- Spider history

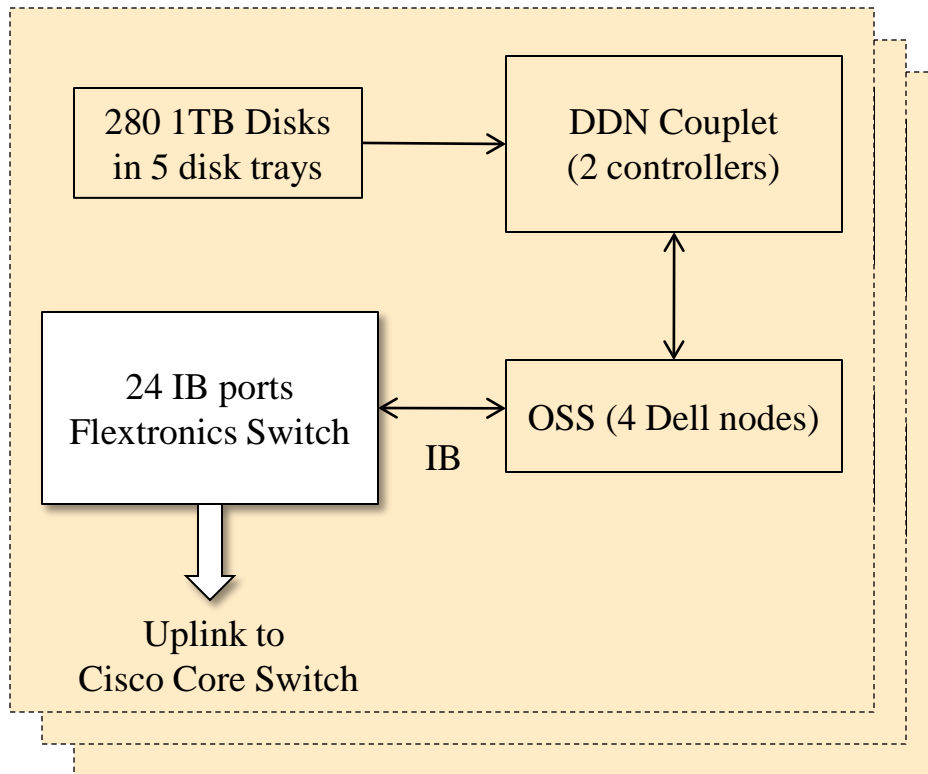


# Why Spider?



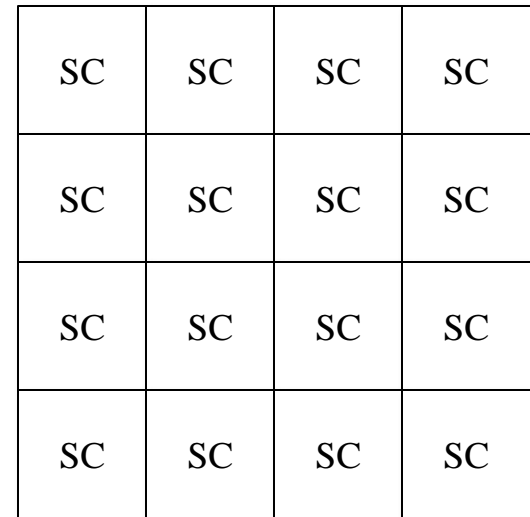
- Eliminate per compute-system storage acquisitions
- Eliminate “file system island”
- Mitigate configuration and maintenance overhead
- Accessibility during system maintenance window

# Building Block: Scalable Cluster

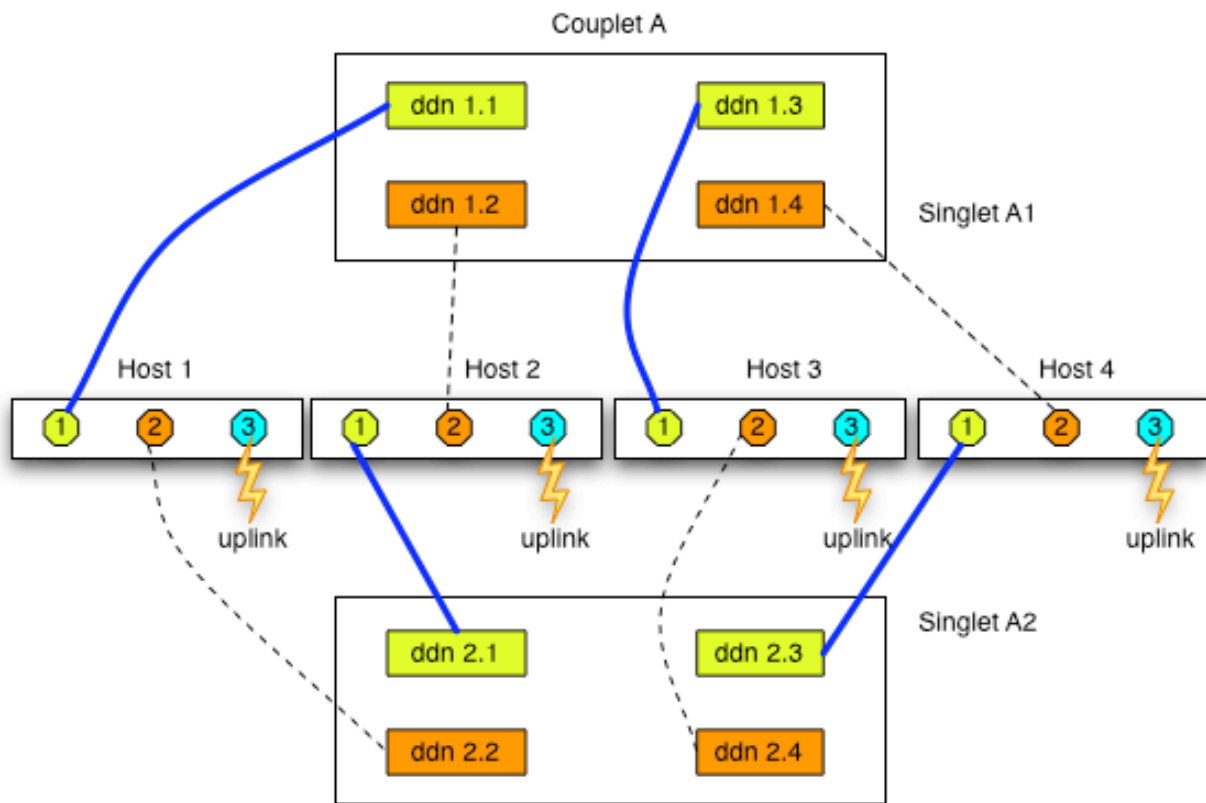


**A Scalable Cluster (SC)**

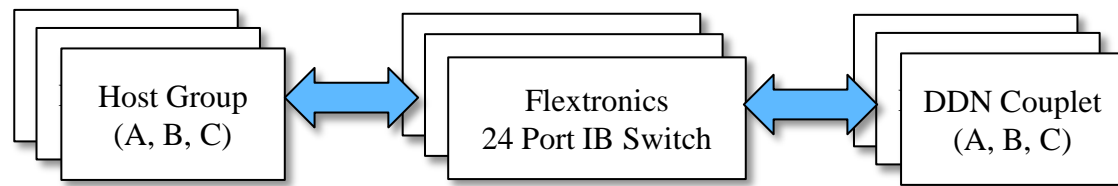
16 SC Units on the floor  
2 racks for each SC



# Host to Controller Redundancy

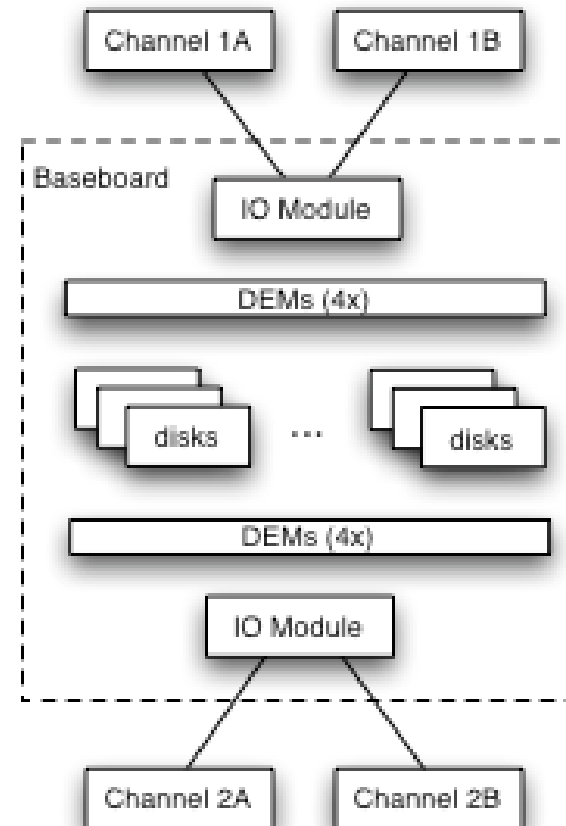
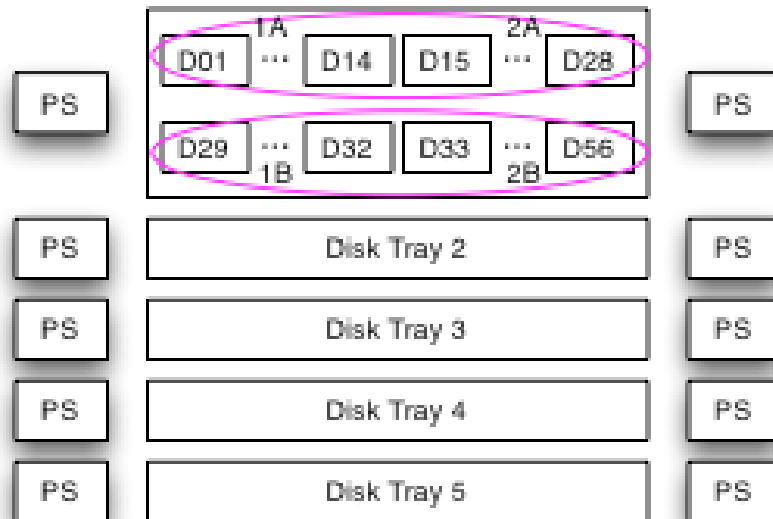
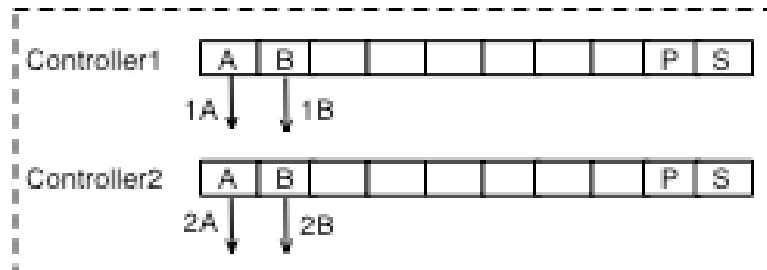


Logical Connections



Physical Connections

# Hardware Redundancy at Controller



# Spider: A System At Scale

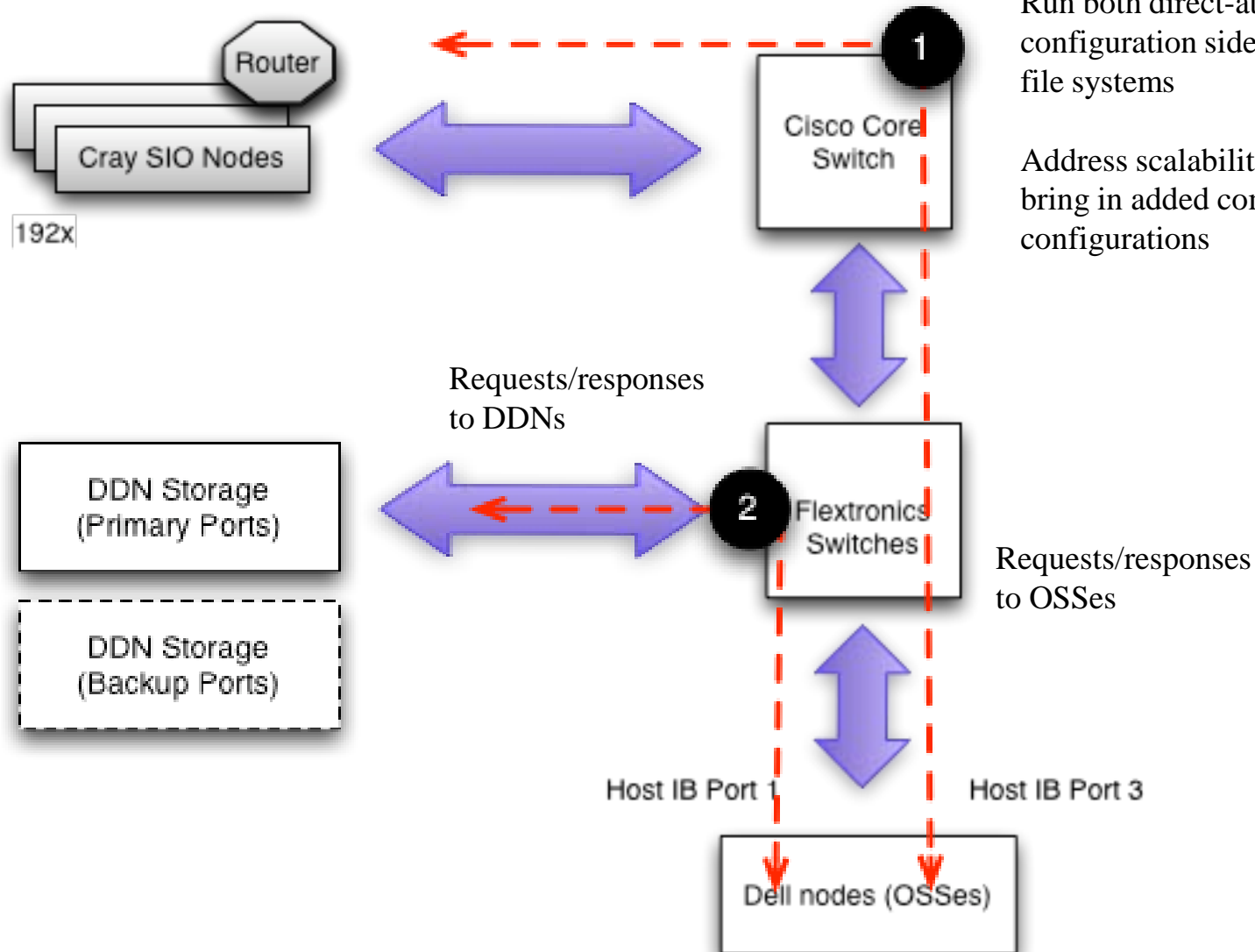
- 13,440 1TB drives
- Over 10.7 PB of RAID 6 Capacity
- 192 storage servers
- Over 3 TB of memory (Lustre OSS)
- Available to many compute systems through high-speed network:
  - Over 3,000 IB ports
  - Over 5 kilometer of cables
- Over half of a megawatt power for disks
- Over 26,000 client mounts for I/O
- Peak I/O performance: 240 GB/second
- Current Status
  - **in production** use on all major NCCS computing platforms



# Challenges

- From Direct-Attached Mode to Routed Mode
- Lesson from Server Side Client Statistics
- Hard bounce, client eviction and I/O shadowing
- Journaling
- Topology aware I/O and fine grained routing
- Reliability Modeling

# Direct-Attached vs. Routed Configuration



Run both direct-attached and routed configuration side-by-side for two file systems

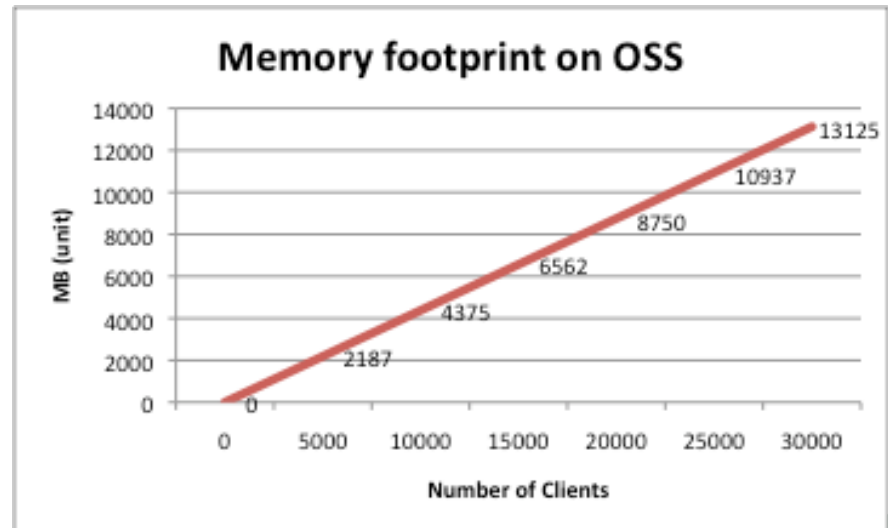
Address scalability issues first, without bring in added complexity by routed configurations

# Server Side Client Statistics

- Problem
  - During our full scale test, OOM occurred
- Tracing
  - Server side maintains client statistics in 64 KB buffer for each client for each OST/MDT.

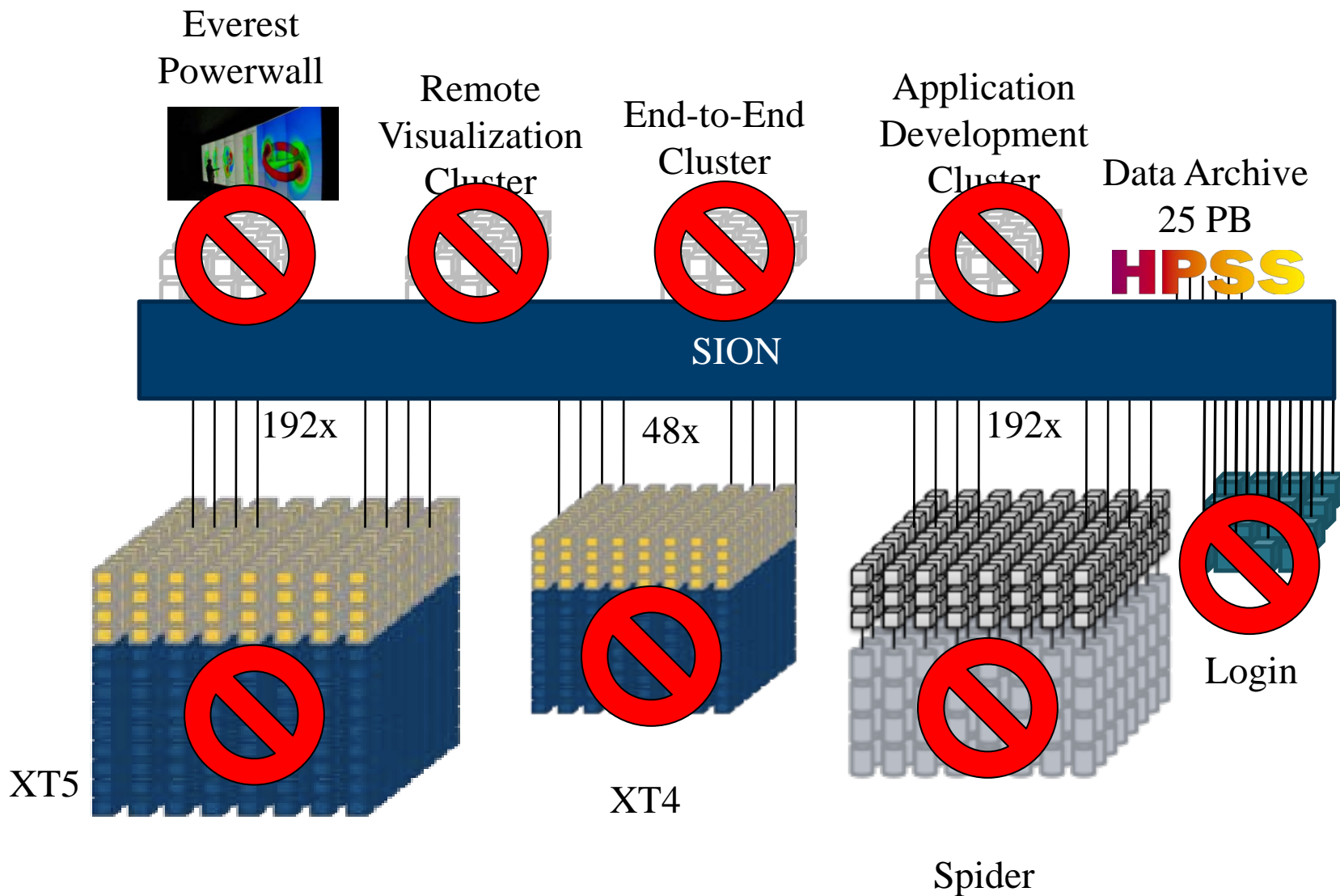
- Solutions

- Move server side stats to client side
- Reduce memory required per client stats (4KB)



# Survive a Bounce:

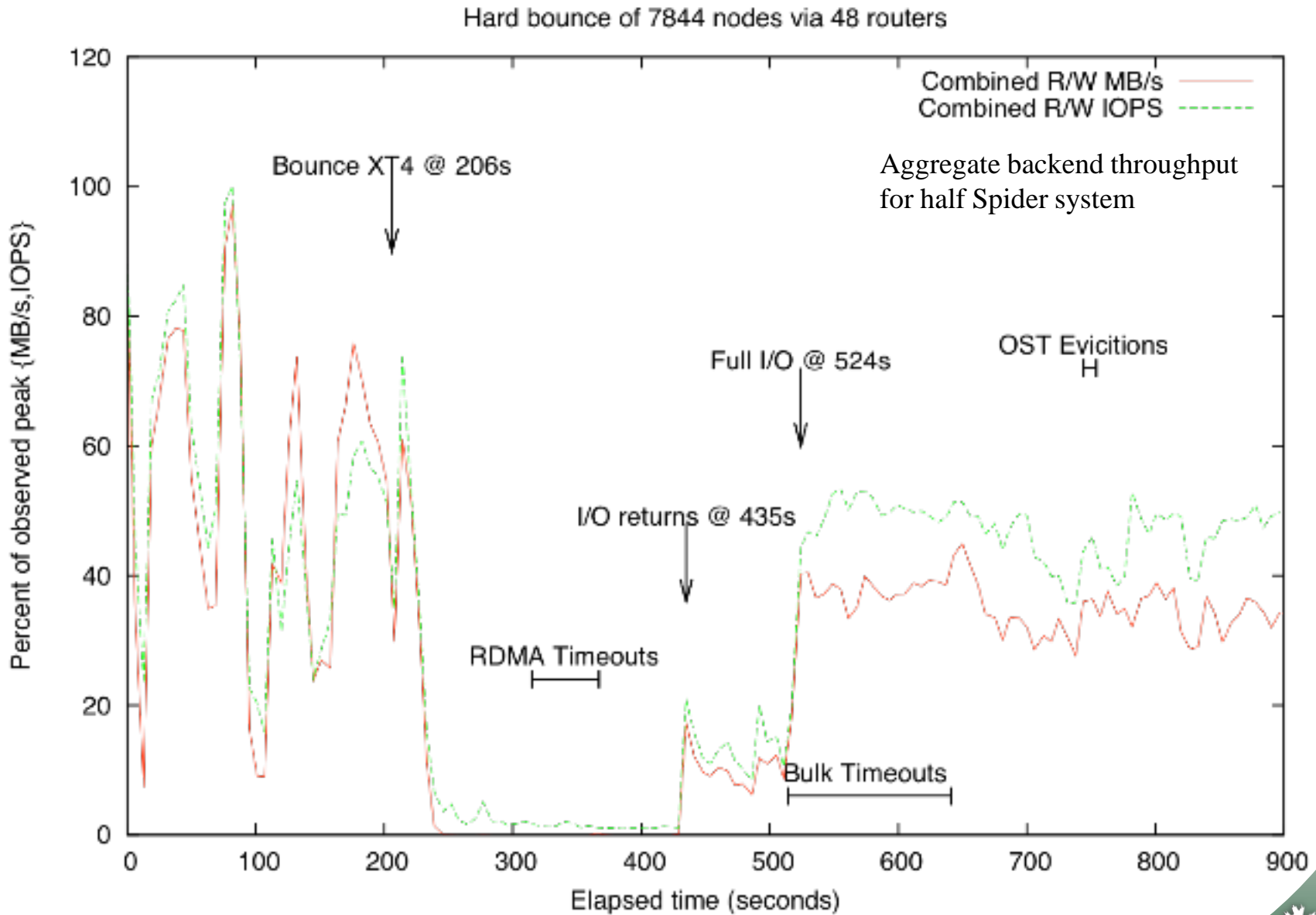
No single point of failure, transparency is key



# Client Eviction

- Jaguar XT5 has over 18K clients
  - A hardware event such as link failure will result a reboot
  - With 18K clients evicted!
- Problem:
  - Each client eviction requires a synchronous write (update to `last_rcvd` log file), done by every OST for each evicted client.
- Solution:
  - Make synchronous write to be asynchronous
  - Batching and parallelizing eviction process if necessary

# Demonstration of Surviving a Bounce



# Filesystem: Journaling Overhead

Table 1: XDD baseline performance

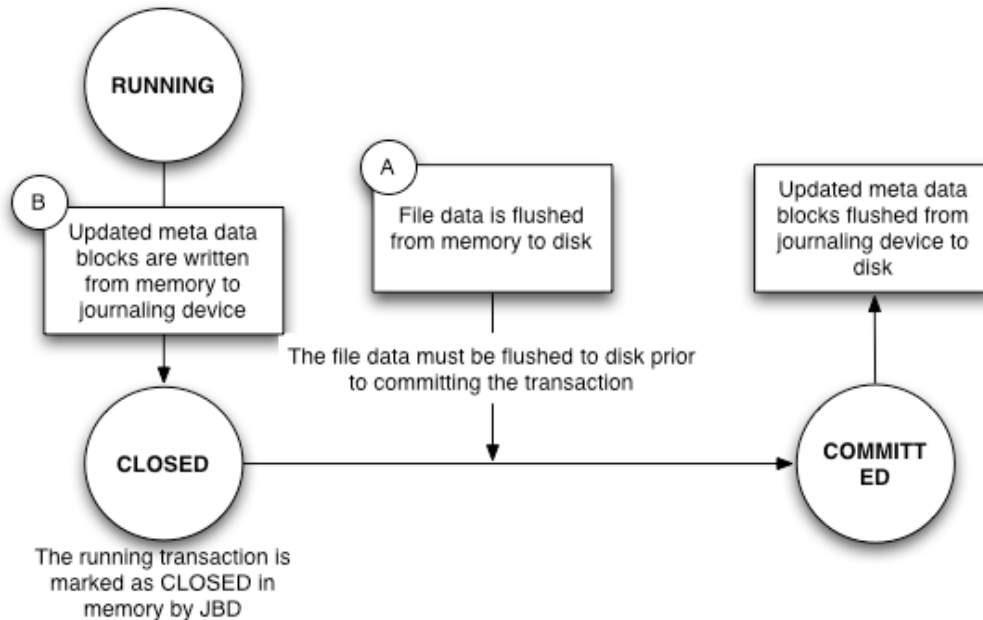
		Read	Write
Single LUN	Sequential	685.62	235.45
	Random	101.74	96.77
28 LUN	Sequential	5570.15	5608.15
	Random	2753.87	2530.5

Filesystem benchmark  
using IOR: ~ 25%

Analysis Result:

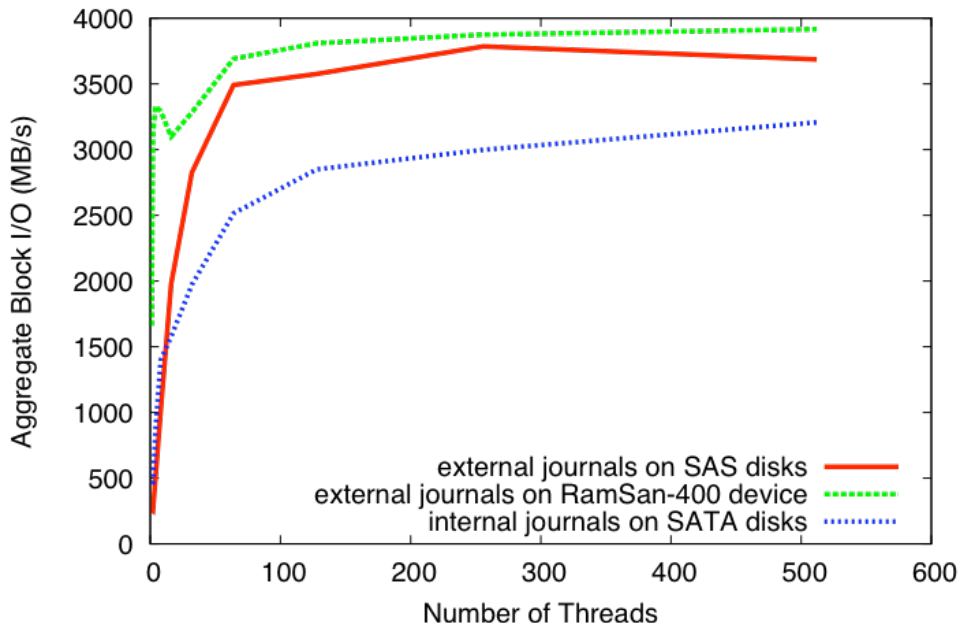
Large number of 4KB writes  
in addition to expected 1MB  
writes so even **sequential  
write exhibits “random”  
I/O behavior**

Problem: ldiskfs journaling



# Two Solutions: Hardware-based and Software-based

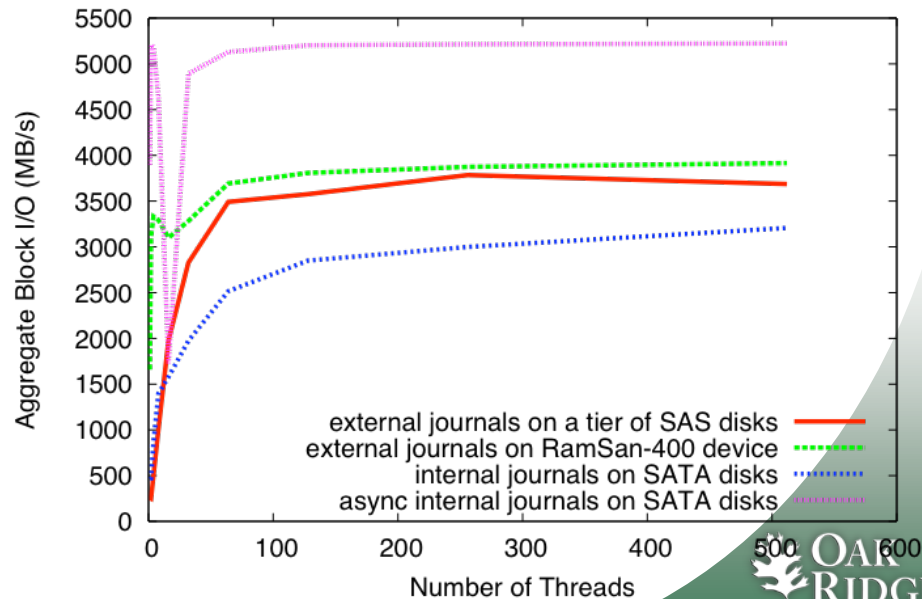
## Hardware-based External Journaling Solutions



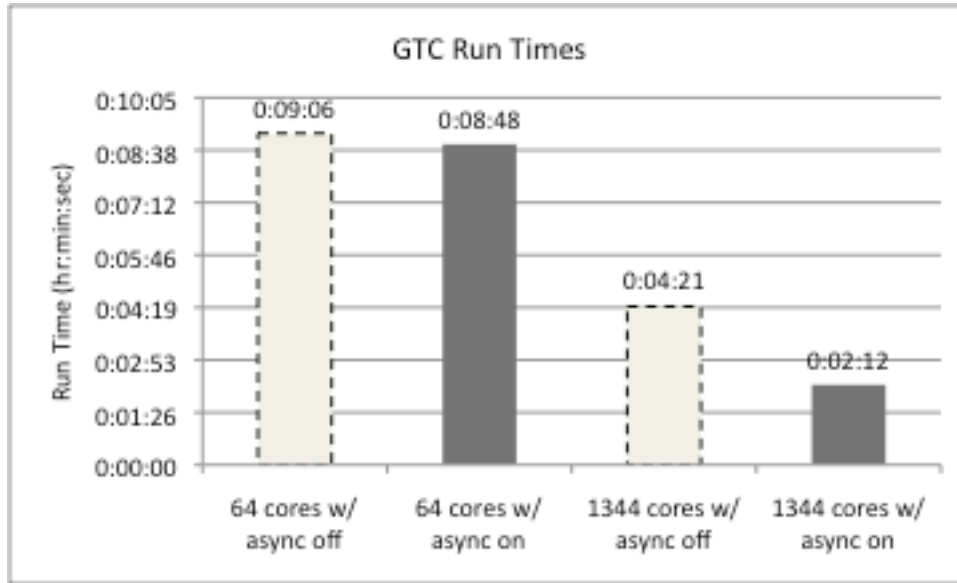
RamSan-400, 3292 MB/sec or 58.7% of the our baseline performance

Async Journaling: 5223 MB/sec or 93% of our baseline performance

## Hardware- and Software-based Journaling Solutions



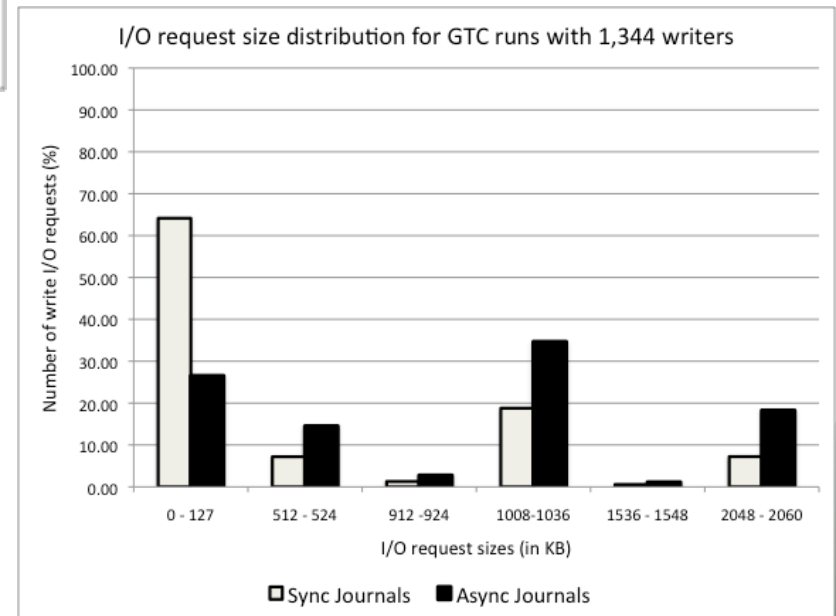
# Application Performance



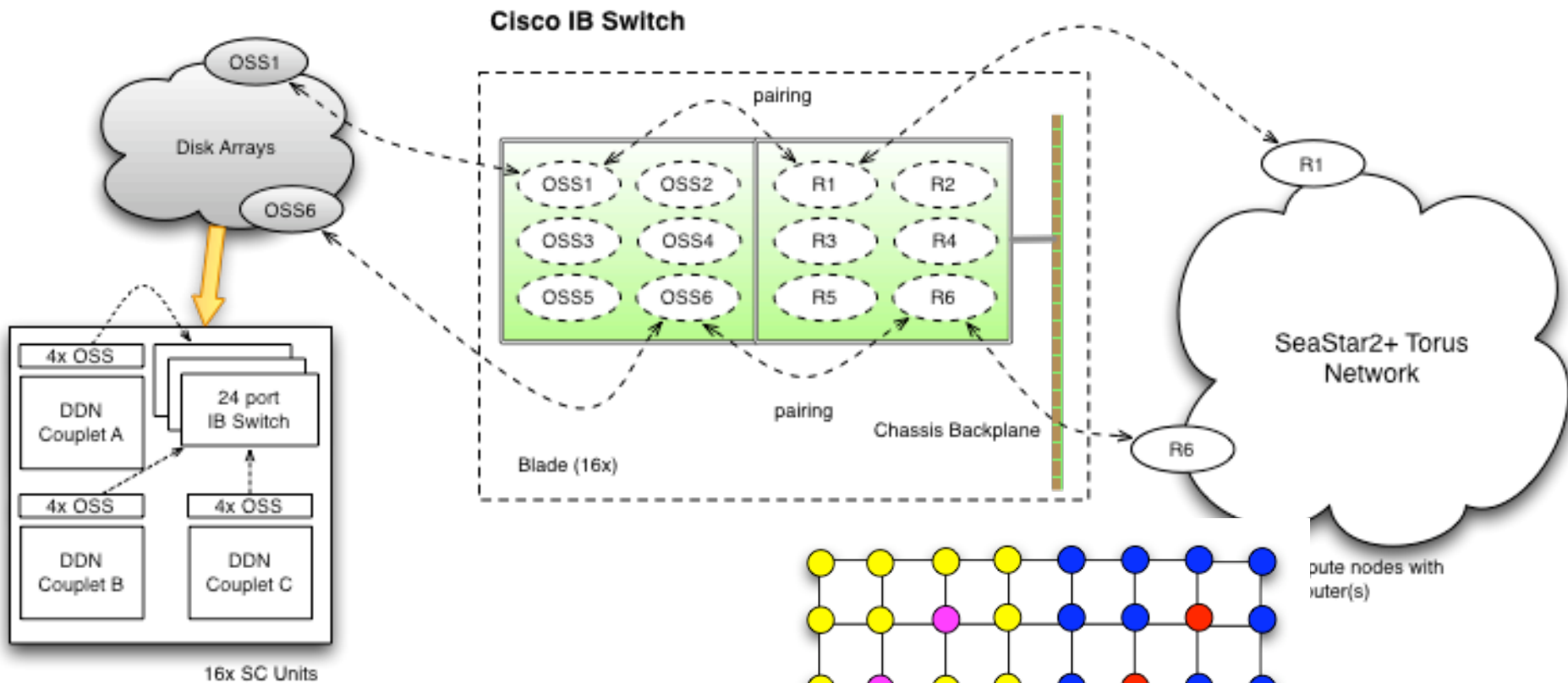
Histogram of I/O requests observed on DDN couplet.

Async-journaling decreases the number of small I/O requests substantially, from 64% to 26.5%

For I/O intensive application such as GTC running at scale, when I/O time dominates, overall runtime reduction with async journaling is up to 50%



# Intelligent LNET Routing



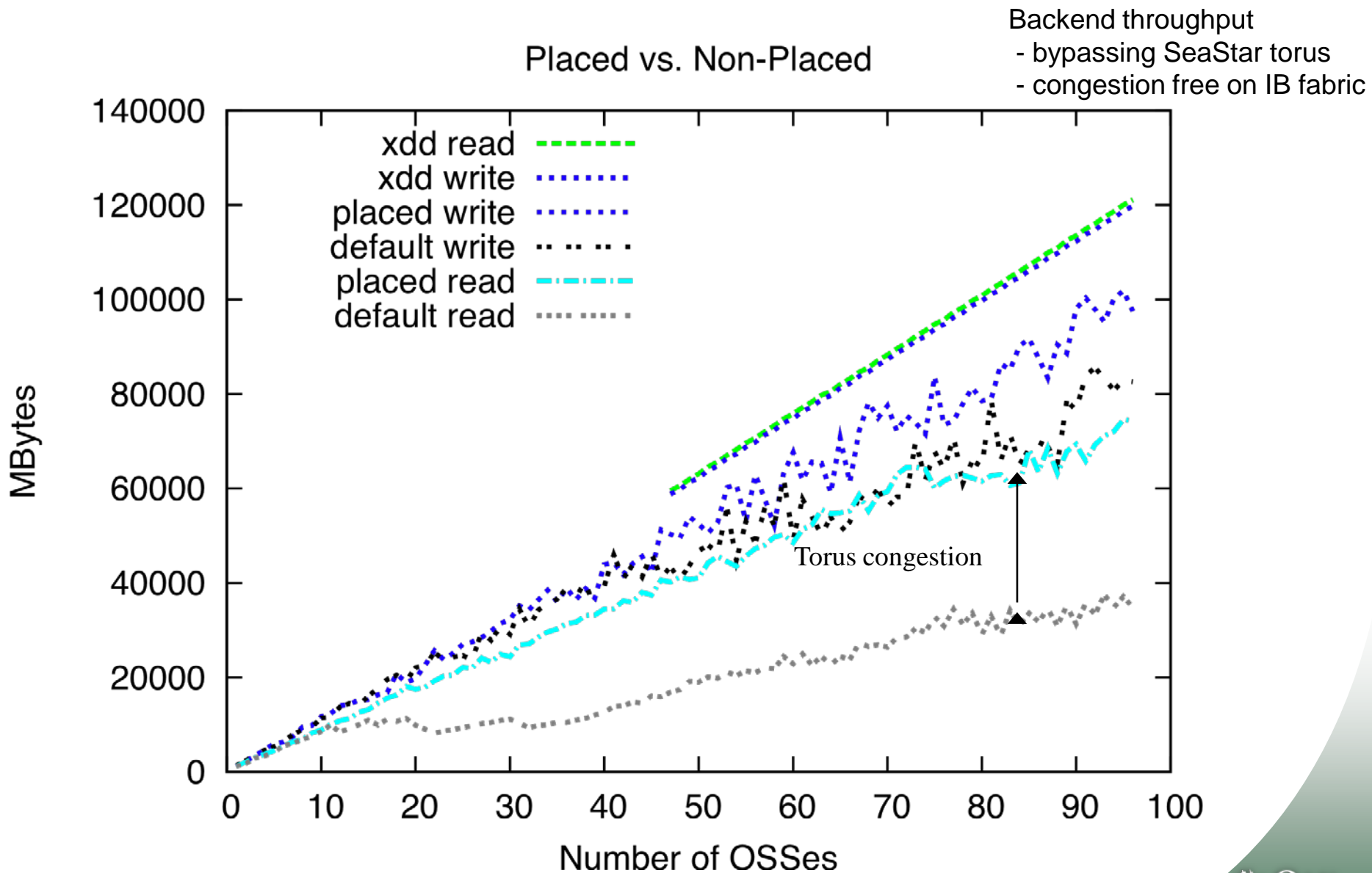
## Potential Network Congestion Points

- SeaStar2 Torus network
- LNET Routers
- Cisco IB Switch

# Strategies to Minimize Contention

- Pair routers and object storage servers on the same line card (crossbar)
  - So long as routers only talk to OSSes on the same line card contention in the fat-tree is eliminated
  - Required small changes to Open SM (not needed anymore with new wiring scheme)
- Optimized physical I/O nodes placement within the Torus network
- Assign clients to nearby routers to minimize contention
- Allocate objects to *nearest* OST managed by an OSS that is topologically close to you.

# Performance Results (xdd vs. IOR, Placed vs. Non-Placed)



# Reliability Modeling (1)

Spider is a system at scale, what would be quantitative expectation of its reliability?

## JBOD: Just A Bunch of Disks

Assuming each drive operates 24 hours a day, 365 days per year, MTTF is 1.2 Million hours  
We can calculate APR as:

$$(365 \times 24) / (1.2 \times 10^6) = 0.73\%$$

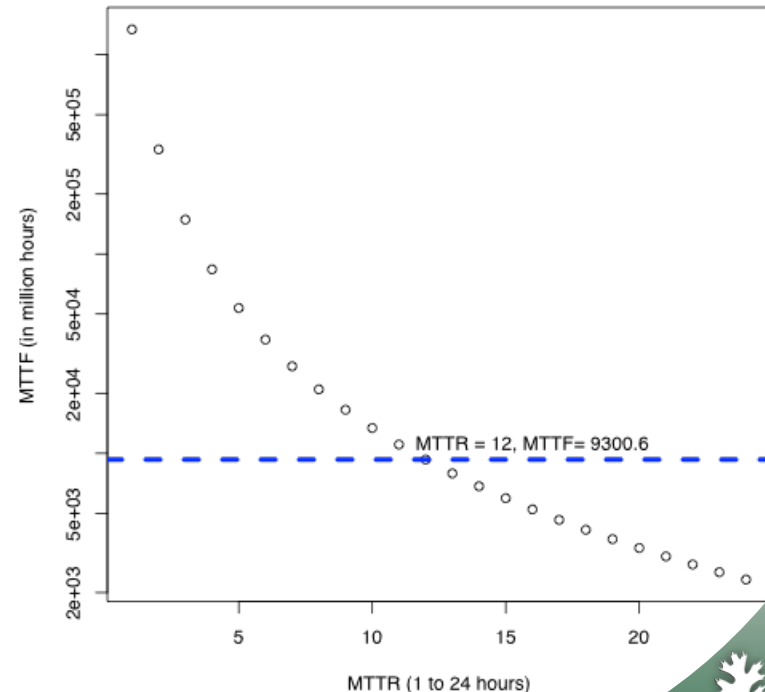
Spider system has 48 couplets, each with 280 1TB disks. Consider a scale factor of 4, then expected disk failures per year should be:

$$48 \times 280 \times 4 \times 0.73\% = 336$$

## RAID 6: Triple Disk Failures

$$\begin{aligned} MTTF_{\text{group}} &= E(X_1) \cdot E(X_2) \cdot E(X_3) \\ &= \frac{MTTF/S}{G+C} \cdot \frac{MTTF/S}{MTTR \cdot (G+C-1)} \\ &\quad \cdot \frac{MTTF/S}{MTTR \cdot (G+C-2)} \\ &= \frac{MTTF^3}{S^3(G+C)(G+C-1)(G+C-2)MTTR^2} \end{aligned}$$

One Couplet, Triple Disk Failures



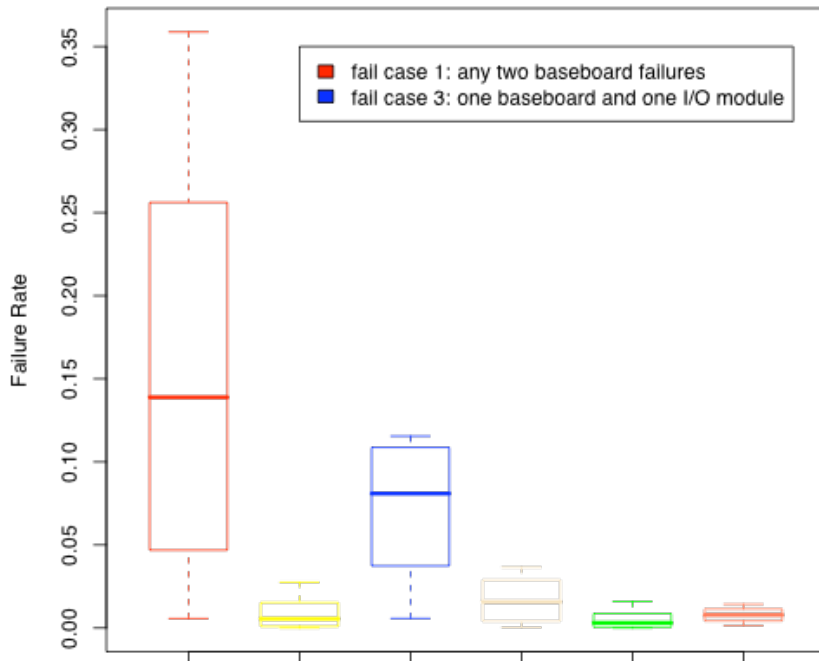
# Reliability Modeling (2)

## Double Disk Failures with Bit Error

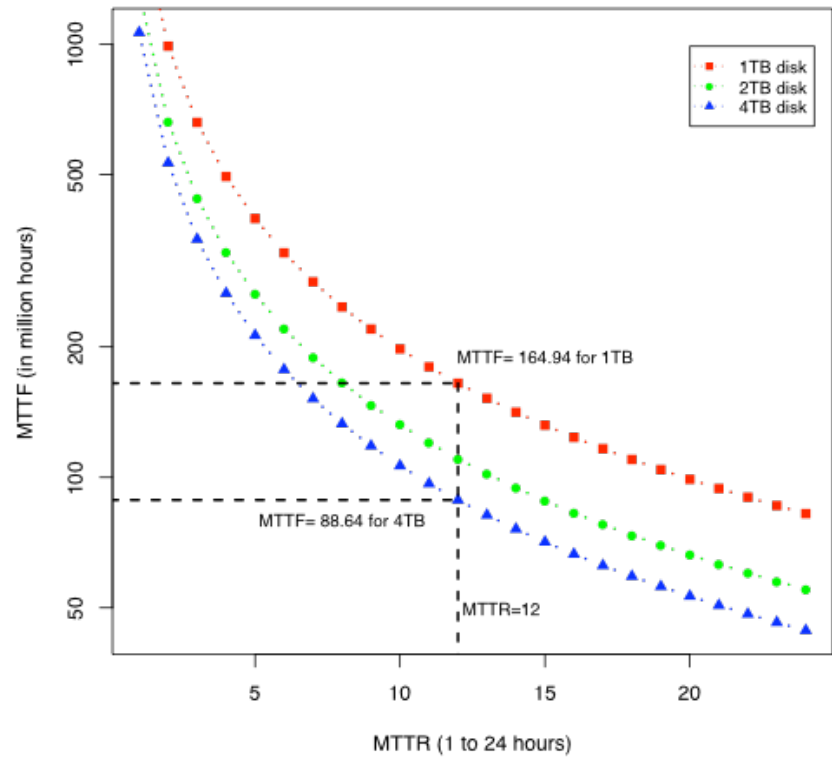
$$MTTF_{group} = \frac{MTTF/S}{G+C} \cdot \frac{MTTF/S}{MTTR(G+C-1)} \cdot \frac{1}{1 - (P_{succ})^{G+C-2}}$$

$$= \frac{MTTF^2}{S^2 MTTR (G+C)(G+C-1)(1 - (P_{succ})^{G+C-2})}$$

Comparison on Failure Cases



One Couplet, Double Disk Failure with Single Bit Error



## System Failures with Peripheral Components

Component	MTTF (in hours)
I/O Module	1,263,856
DEM	1,552,437
Baseboard	356,143

- Baseboard contributes 50% of failure rate
- Over first year, we expect 0.64 failed couplet on 1<sup>st</sup> year, 2.42 failed couplets by 2<sup>nd</sup> year

# Conclusion

- A system at this scale, we can't just pick up the phone and order one
- A close collaboration with vendors (Cray, Sun, DDN, and Cisco) and community at large. Lustre Center of Engineering (LCE) proves to be very helpful
- No problem is small when you scale it up
- As we are ahead of the curve, we are plowing the road for the community:
  - Driving Lustre scalability through collaborations with key stake holders.
  - Had two Lustre scalability workshops this year at ORNL

# Thank you!

**Contact Information**  
**Phone: (865) 574-7209**  
**Email: fwang2@ornl.gov**



Technology Integration Team