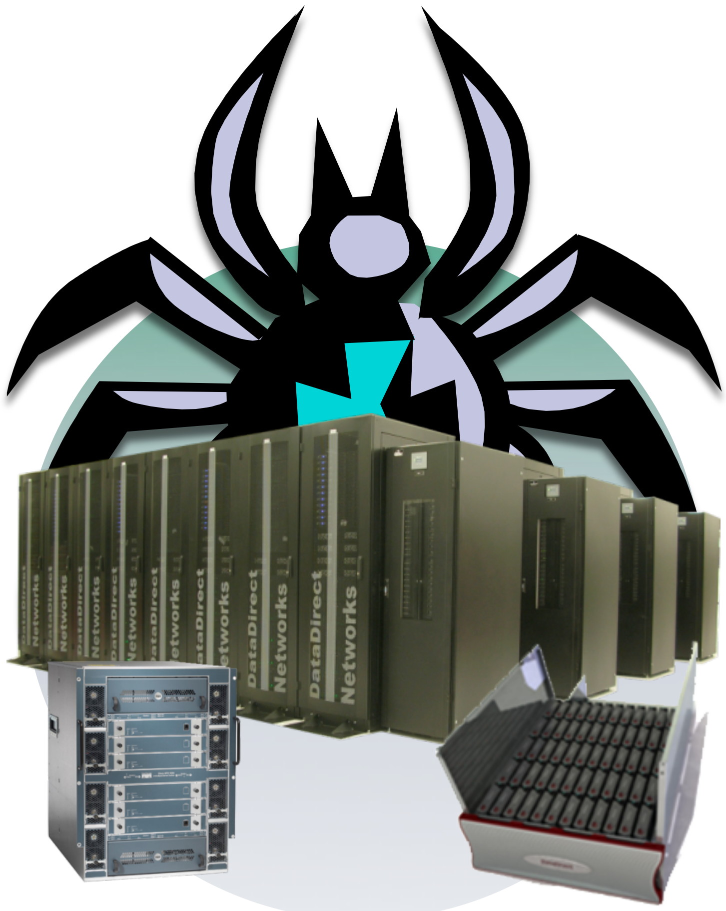
2015 Parallel File System Requirements

OAK RIDGE LEADERSHIP COMPUTING FACILITY

U.S. DEPARTMENT OF ENERGY

OAK RIDGE National Laboratory

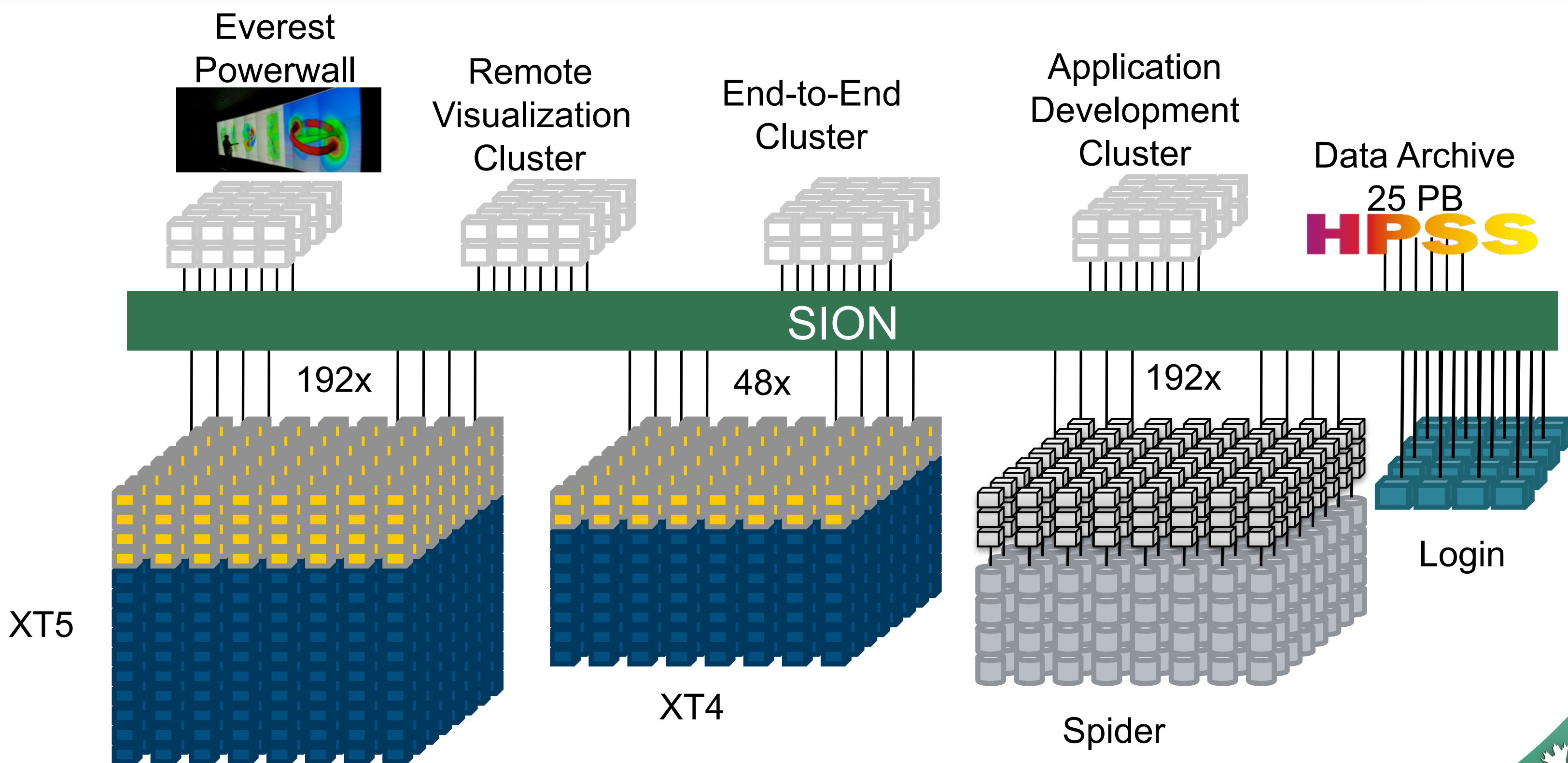
Where are we today: Center-wide File System



- "Spider" provides a shared, parallel file system for all systems
  - Based on Lustre file system
- Over 10 PB of RAID-6 Capacity
  - 13,440 1 TB SATA Drives
- 192 Storage servers
  - 3 TeraBytes of memory
- Available from all systems via our high-performance scalable I/O network
  - Over 3,000 InfiniBand ports
  - Over 3 miles of cables
  - Scales as storage grows
- Collaborative effort was key to success
  - ORNL, Cray, DDN, SUN (LCE)

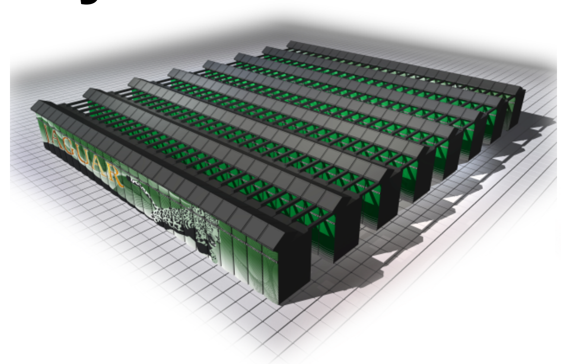
Benefits of Spider

- **Accessible from all major LCF resources**
- **Accessible during maintenance windows**
- **Decouples simulation platform procurement from storage system procurement**
  - **Allows file system to take an independent trajectory**
  - **Procurements can be planned to better coincide with vendor roadmaps**



Spider Status

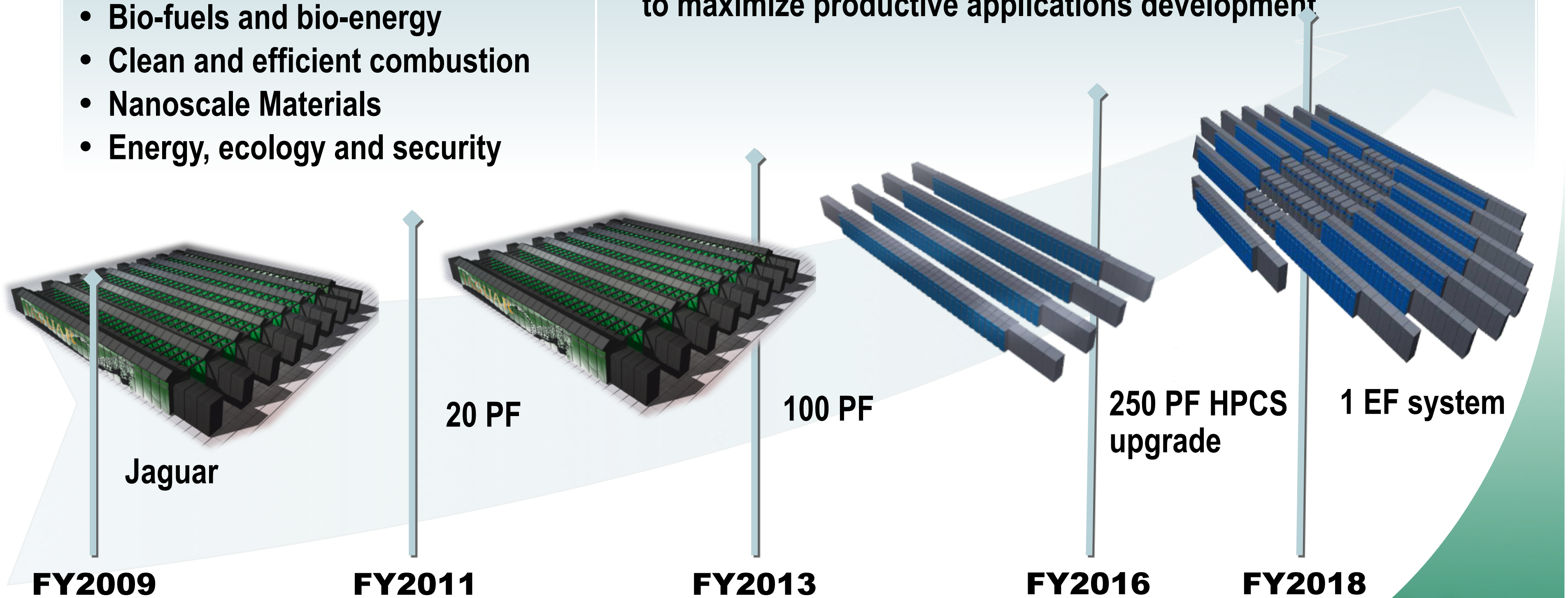
- **Demonstrated bandwidth of over 200 GB/s on direct attached storage**
- **Demonstrated stability on a number of LCF systems**
  - Jaguar XT5
  - Jaguar XT4
  - Smoky
  - Lens
  - All of the above..
    - Over 26,000 clients mounting the file system and performing I/O



- **Early access on Jaguar XT5 and Lens today!**
  - General Availability this Summer

LCF Roadmap for delivering an Exascale System for Science in a decade

Mission: Deploy and operate the computational resources needed to tackle global challenges	Vision: Maximize scientific productivity and progress on the largest scale computational problems
• Climate change • Terrestrial sequestration of carbon • Sustainable nuclear energy • Bio-fuels and bio-energy • Clean and efficient combustion • Nanoscale Materials • Energy, ecology and security	• Providing world class computational resources and specialized services for the most computationally intensive problems • Providing a stable hardware/software path of increasing scale to maximize productive applications development



2015 – Leadership Computing

- In 2016 our 100 petaflop platform will be upgraded to 250 petaflops
  - A roadmap based on deliverable technologies
  - Actively working with vendors on these technologies
  - A 154x increase in total flops
  - Unlikely to see a similar increase in total system memory 30x would put as near 10 Petabytes of system memory

2015 – I/O Performance Requirements

- 1 GB/s / TFLOP
  - Checkpoint time assumes 75% of total system memory

	TFLOP	GB/s	Checkpoint (Seconds)
Current System ->	1380	1380	217
2011 System->	20000	20000	70
2013 System ->	100000	100000	30
2016 System ->	250000	250000	24

  - These bandwidths are not achievable due to budgetary constraints

2015 – I/O Performance Requirements

- **I/O performance requirements driven by system memory**
  - **Checkpoint 75% of system memory in *x* seconds**

Checkpoint time in seconds		1125	720	360
	System Memory (TB)	Bandwidth (GB/s)		
Current System ->	300	200	313	625
2011 System->	1400	933	1458	2917
2013 System ->	3000	2000	3125	6250
2016 System ->	6000	4000	6250	12500

Total Capacity

- Ability to store 30 full system (75% of memory) checkpoints
  - 1400 TB system memory -> 31 Petabytes
  - 3000 TB system memory -> 67 Petabytes
  - 6000 TB system memory -> 135 Petabytes

- Capacity requirements are achievable
  - Evolving use-cases may require substantially more capacity
  - Bandwidth requirements may increase capacity dramatically

Functionality

- **File system name space can span multiple sites**
  - Implications for file system semantics?
- **pNFS interoperability - Cray**
  - makes Lustre site-wide storage accessible from non-Lustre clients

Performance and Capacity

- Storage space
  - 100's of Petabytes of capacity
- I/O Bandwidth
  - 6 – 12 Terabytes/second
- Number of files
  - ~1 Trillion files
- Metadata operation rates
  - 1,000,000 operations / second
    - Linear scaling of per MDS performance (CMD)
  - CMD required
  - Lightweight large scale query capability - LLNL

Performance and Capacity

- Tiered storage support
  - SSDs
  - Enterprise disk
  - Near line disk
  - Tape
- Lighter weight file system interfaces
  - Posix on interactive nodes
  - Lockless semantics on computes
    - APIs suitable for middleware libraries
- Ultra fast write/read for checkpoint/restart – CEA
- User tools for performance tuning - HPCS

Usability

- File system responsiveness
  - Interactive users should not be forced to tolerate response times on the order of minutes
  - Pathological jobs can substantially degrade interactive performance
    - See NWChem
  - QOS may play a role here – favoring interactive clients over computes

Manageability

- Millions of objects
  - O(100,000) clients
  - O(100,000) disks
- Transparently add/change infrastructure
  - Hardware, software
  - Version compatibility
    - more thanN-1, N, proxy with protocol conversion? - CEA
- Tiered storage
  - Seamless data migration
  - Enforcing purge/replication policies
  - Lustre as an HSM?

Manageability

- Accounting
- Policy manager and policy engine
  - Bandwidth percentage (Shares)
  - Quotas
  - Allocate by user, system, project etc.
  - Global mechanism (i.e. integrated with batch system)
- Operational Scalability (scaling not just for benchmarks but operationally with 2015 hardware for data and metadata) – HPCS

Manageability cont.

- Reference design for Lustre storage – Cray
  - requirements (ratios, facts/figures) for MDTs, OSTs, JDT’s (JDT = journal data targets, separate from the OSTs)
  - use of storage tiers (caches, SSDs, SAS, SATA)
- Open Data Management to interface to 3rd party backup, archive, HSM, ILM suppliers - Cray

RAS

- I/O performance lagging FLOPs
  - Storage system component growth will outpace component growth in our simulation platforms
  - Components will be continuously failing - SUN
- rapid end-to-end failover (ie from client to disk) - beyond end-to-end data integrity, rapid failover is needed to provide a higher reliability solution – Cray

Describing your I/O for high-performance

- **Beyond file system "hints" we need a common API to describe I/O operations**
- **Allows the file system to allocate objects on storage "classes" appropriate for subsequent I/O operations**

Rethinking I/O – Don't hide behind a FD

- Parallel file systems can learn from other parallel computing middleware's most notably MPI
  - Our codes describe global and local communication via MPI semantics in order to scale
- Simulations which scale must describe their I/O operations
  - Beyond environment variables, ioctl's and stupid pet tricks (knowing stripe alignment)
- Need a parallel I/O interface that expresses the lower level infrastructure in order to scale to 100,000 clients and achieve reasonable performance
  - This will be FS specific but can be abstracted by middle-ware layers

Meeting in the Middle

- APIs to describe I/O operations
  - Gives the file system the ability to optimize
- Intelligent file systems
  - We can't put all the onus on users and middle ware libraries
  - Ability to adapt based on changing workloads is critical
    - Adaptive routing
    - Adaptive block/object allocation