

# The ASCI/DOD Scalable I/O History and Strategy

Run Time Systems and Scalable I/O Team

Gary Grider

CCN-8

Los Alamos National Laboratory

From LAUR 042787

05/2004

# Parallel File Systems and Parallel I/O

Why - From the ASCI implementation plan:

- ◆“Responsible for ensuring that ASCI applications have access to reliable, easy-to-use, high-performance input/output systems whose throughput rates and capacity are in balance with the computational speed and memory size of the ASCI platforms...”
- ◆“Provide standard parallel I/O interface to all ASCI Tri-Labs applications and throughout the HPC community – MPI-IO”

How – Leverage, Partnerships, Careful Planning, **and did I mention Leverage?**

- ◆Vendors
- ◆Standards bodies
- ◆Universities
- ◆HPC Community

In 1995-1996 “ Oh no, we don’t have a good have a good scalable file system story”

For Sandia, LLNL, LANL and DOD, the need for a global parallel file system was there from the beginning of clustered based parallel computing,

**few** solutions existed,

**none** were heterogeneous,

**none** were open source,

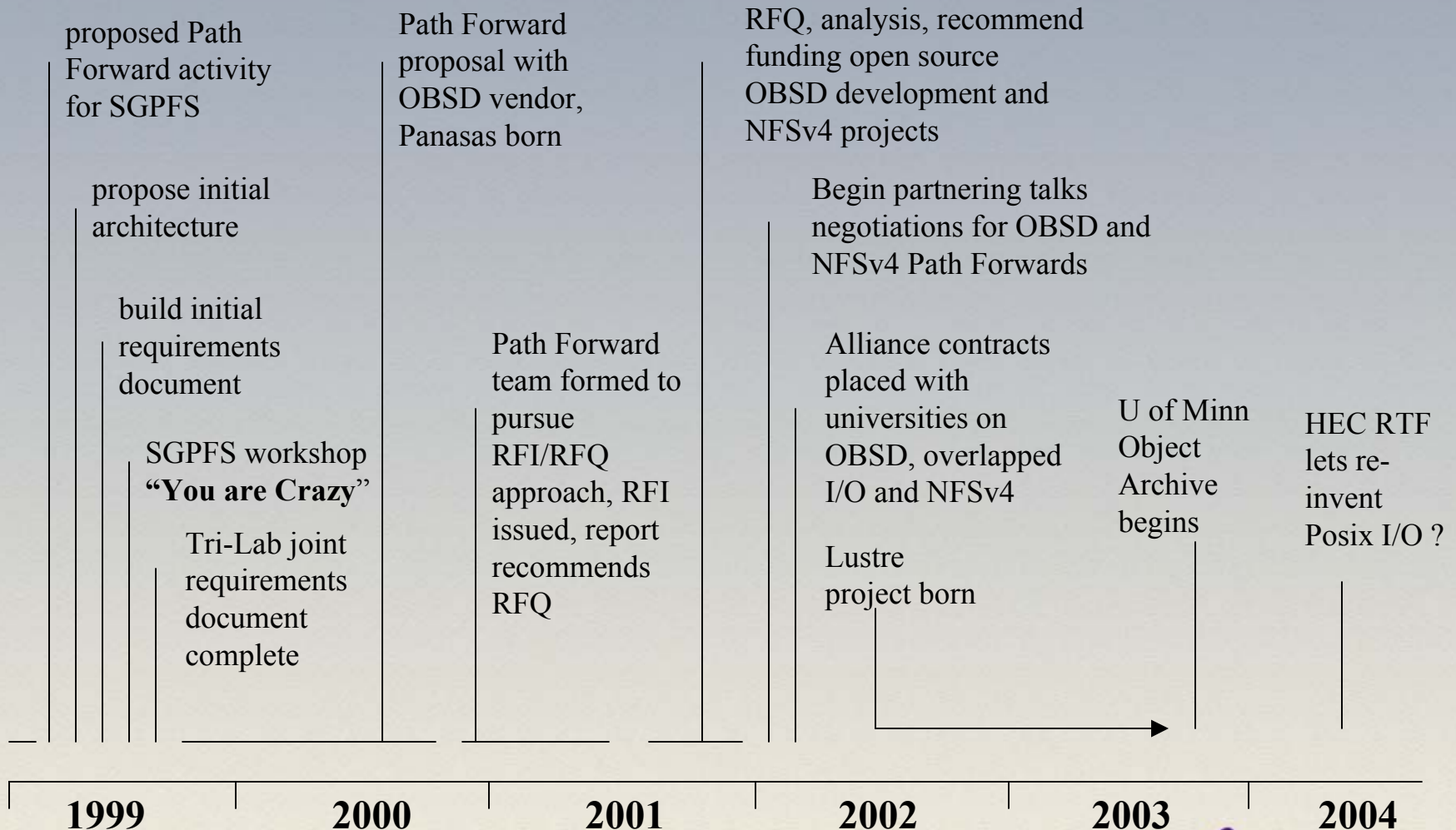
**none** were based on standards, and

**none** were secure on a public net.

This is primarily for our giant clusters, secondarily for our enterprise, and lastly across multiple enterprises/sites

We saw Linux clusters coming in the future which made the problem very real and very evident

# Historical Time Line for R&D



## FS Requirements Summary

From Tri-Lab File System Path Forward RFQ (which came from the Tri-labs file systems requirements document)

<ftp://ftp.lanl.gov/public/ggrider/ASCIFSRFP.DOC>

POSIX-like Interface, Works well with MPI-IO, Open Protocols,  
Open Source (parts or all), No Single Point Of Failure , Global  
Access

***Global name space, ...***

***Scalable bandwidth, metadata, management, security ...***

***WAN Access, Global Identities, Wan Security, ...***

***Manage, tune, diagnose, statistics, RAS, build, document,  
snapshot, ...***

***Authentication, Authorization, Logging, ...***

# FS Requirements Detail 1

- 3.1 POSIX-like Interface
- 3.2 No Single Point Of Failure
- 4.1 Global Access
  - 4.1.1 Global Scalable Name Space**
  - 4.1.2 Client software**
  - 4.1.3 Exportable interfaces and protocols**
  - 4.1.4 Coexistence with other file systems**
  - 4.1.5 Transparent global capabilities**
  - 4.1.6 Integration in a SAN environment**
- 4.2 Scalable Infrastructure for Clusters and the Enterprise
  - 4.2.1 Parallel I/O Bandwidth**
  - 4.2.2 Support for very large file systems**
  - 4.2.3 Scalable file creation & Metadata Operations**
  - 4.2.4 Archive Driven Performance**
  - 4.2.5 Adaptive Prefetching**
- 4.3 Integrated Infrastructure for WAN Access
  - 4.3.1 WAN Access To Files**
  - 4.3.2 Global Identities**
  - 4.3.3 WAN Security Integration**

## FS Requirements Detail 2

### 4.4 Scalable Management & Operational Facilities

**4.4.1 Need to minimize human management effort**

**4.4.2 Integration with other Management Tools**

**4.4.3 Dynamic tuning & reconfiguration**

**4.4.4 Diagnostic reporting**

**4.4.5 Support for configuration management**

**4.4.6 Problem determination GUI**

**4.4.7 User statistics reporting**

**4.4.8 Security management**

**4.4.9 Improved Characterization and Retrieval of Files**

**4.4.10 Full documentation**

**4.4.11 Fault Tolerance, Reliability, Availability, Serviceability (RAS)**

**4.4.12 Integration with Tertiary Storage**

**4.4.13 Standard POSIX and MPI-IO 4.4.14 Special API semantics for increased performance**

**4.4.15 Time to build a file system**

**4.4.16 Backup/Recovery**

**4.4.17 Snapshot Capability**

**4.4.18 Flow Control & Quality of I/O Service**

**4.4.19 Benchmarks**

## FS Requirements Detail 3

### 4.5 Security

#### **4.5.1 Authentication**

#### **4.5.2 Authorization**

#### **4.5.3 Content-based Authorization**

#### **4.5.4 Logging and auditing**

#### **4.5.5 Encryption**

#### **4. 5.6 Deciding what can be trusted**

## It Has to Scale with Our Machine Appetite

### Aggregate Bandwidth Rates for One Parallel Job Simulation & Physics Model Aggregate FS Requirements

|                                               | 1999            | 2003            | 2005             | 2008              |
|-----------------------------------------------|-----------------|-----------------|------------------|-------------------|
| <b>Teraflops/Clients</b>                      | <b>3.9 / 6K</b> | <b>30 / 12k</b> | <b>100 / 50K</b> | <b>400 / 100k</b> |
| <b>Memory Size (TB)</b>                       | <b>2.6</b>      | <b>13-20</b>    | <b>32-67</b>     | <b>44-167</b>     |
| <b>I/O Rates (GB/s)<br/>N to N and N to 1</b> | <b>4 – 8</b>    | <b>20-60</b>    | <b>50-200</b>    | <b>80-500</b>     |

## Other things have to Scale Too

### File System Attributes

|                                                       | 1999             | 2002              | 2005                | 2008                |
|-------------------------------------------------------|------------------|-------------------|---------------------|---------------------|
| Teraflops                                             | 3.9              | 30                | 100                 | 400                 |
| Memory size (TB)                                      | 2.6              | 13-20             | 32-67               | 44-167              |
| File system size (TB)                                 | 75               | 200 - 600         | 500 -2,000          | 20,000              |
| Number of Client Tasks                                | 8192             | 16384             | 32768               | 65536               |
| Number of Users                                       | 1,000            | 4,000             | 6,000               | 10,000              |
| Number of Directories                                 |                  |                   |                     |                     |
|                                                       | $5.0 \cdot 10^6$ | $1.5 \cdot 10^7$  | $1.8 \cdot 10^7$    | $1.8 \cdot 10^7$    |
| Metadata rates (random full stat or completed insert) | 500/sec<br>1 mds | 2000/sec<br>1 mds | 20,000/sec<br>n mds | 50,000/sec<br>n mds |
| Number of Files                                       | $1.0 \cdot 10^9$ | $4.0 \cdot 10^9$  | $1.0 \cdot 10^{10}$ | $1.0 \cdot 10^{10}$ |

## Other Requirements Besides Scalability

Based on Standards (ANSI T10 is now acceptance of Draft 1)

Security more like AFS/DFS but better

Content based security, born on marks, hooks for end to end encryption, extensible attributes, etc.

Real transactional security on the SAN, not simple zoning and other poor attempts (ANSI T10)

Global, Heterogeneous, Protocol Agnostic, open source, open protocols (ANSI T10), (NFSv4.X IETF), (formal POSIX enhancements), all in progress

POSIX behavior with switches to defeat parts

Lazy attributes, byte range locks, etc.

WAN behavior like AFS/DFS but better

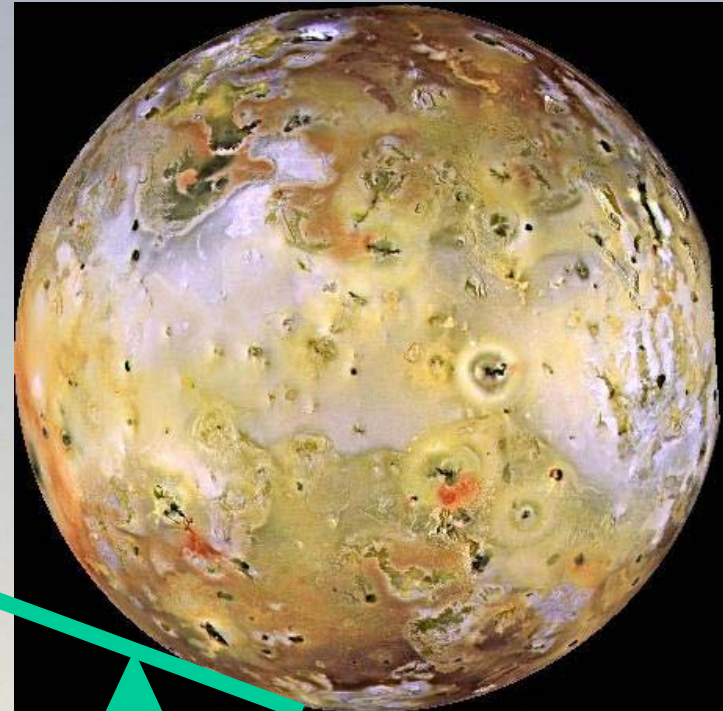
Including ACL's, GSS, multi domain, directory delegation, etc.

Scalable management (sorry, scalability keeps coming up)

A product, supported by a market larger than the Tri-Labs, only glimpses of products now, and already nearing 100 sites trying out one of the 3-4 solutions, expected to be in the 1000's in 1-2 years.

# I promised you Leverage, Collaborations, Planning, and Leverage

Vendors,  
other labs,  
universities,  
standards, etc.



# The SIO External Contracts (alliances/collaborations) An integral part of the strategy!

**Use Alliances and other partnerships to fill the holes  
in the strategy not filled by SGPFS development!**



- We told our “SGPFS” developers to not assume no byte range locking to get immense scaling. This left a semantic hole in the architecture for overlapped regions based I/O. Northwestern was chosen to fill this hole.
- We anticipated issues in our “SGPFS” development in dealing with scaling metadata and security. UCSC was chosen to research these issues in advance of our development efforts.
- We wanted to provide access to “SGPFS” for legacy systems, over WAN, assist with DFS end of life, and other issues. Michigan was chosen to fill this hole.
- We were luckily able to capitalize on Argonne collaboration to gain needed features for MPI-IO and to package in support for the “SGPFS” effort within ROMIO, the basis for many/most MPI-IO implementations

# Vendor Collaborations

## Solution for Linux clusters and enterprise class heterogeneous global parallel file systems

- ◆ HP/CFS/Intel Lustre Path Forward for object based secure global parallel file system
  - Very scalable bandwidth, good non scaled metadata
  - Being used at LLNL
  - Completed first 3 milestones
  - Starting metadata scaling work
  - Production quality/removing some of the too specific leveraging will take time
- ◆ Panasas
  - Very scalable bandwidth, good non scaled metadata
  - Being used on viz clusters in open, Pink, and being deployed on Lightning, metadata scaling started
  - Production quality also takes time

# Vendor Collaborations (Continued)

## MSTI's MPI-IO

- ◆Advanced features
- ◆ADIO layers for Panasas and Lustre

## IBM's Storage Tank

- ◆Beginning to work with IBM-Ohio Supercomputer Center for bandwidth scaling etc.
- ◆HSM design for Object file systems
  - Tied to our University of Minnesota Intelligent Storage Consortia work
- ◆NFSv4.X as native client for SAN and Object file systems
  - Tied to our University of Michigan NFSv4 work, In IETF now!
- ◆End to End secure file systems
  - Tied to our University of Minnesota work

# Additionally, Our University Partnerships

## Michigan

- ◆Assisting in design and testing of NFSv4 for multi-domain enabled secure NFS, NFS parallel extensions, and NFS in a non IP environment, NFS as a client to SAN and OBFS file systems, NFS server load balancing in a cluster file system setting
- ◆Some results are showing up in Linux 2.5/2.6 kernel, pNFS IETF work starting

## Northwestern

- ◆Move coordination up to MPI-IO, out of the file system for overlapped I/O

## UCSC

- ◆Study on object storage efficiency and clustered metadata scaling, to guide our two object file system future design and development activities

## ANSI T10 OBSD reference at CMU

Minnesota (Intelligent Storage Consortia) and STK to develop first infinite (HSM) object based device to enable parallel object archive

- ◆Leverage existing commercial HSM software (multiple copies in parallel)
- ◆This “Intelligent Storage Consortia”, is one place to foster the Object model and what all can be done with it
- ◆Other interesting OBSD implementations, databases, real time, SDM, etc. (feeds our POSIX extension work)
- ◆End to end secure file system work (SFS with Storage Tek and NSA)

## Collaborations with other labs

### ANL

MPICH MPI-2 reference  
MPI-IO advanced features  
and ADIO  
PVFS2 explorations  
NetCDF  
GridFTP

### PNNL

Lustre

### Brookhaven

Panasas

### OSC

Storage Tank as parallel file  
system



## Storage Device Pursuits

**We continually look at new device developments (mostly NDA)**

- Mems and Mram (below memory and above disk, for metadata, caching, memory mirroring for possible new fault tolerance model instead of defensive I/O)
- Holographic devices – will we ever see them?
- Low power (spin up/down), low maintenance (large% spare), extremely dense disk ideas (bricks, ice cubes, etc.), perhaps putting more and more of the archive on this type of device if costs make sense, if reliability can be managed, etc.)
  - ◆What is the interface, block, object (T10), tape devices/robot, etc.
  - ◆Reliability, will +1 technologies work, will it require more than +1 for archive applications, removable bulk backup, +2 or other more reliable ideas

# High End Computing Revitalization Task Force Identified R&D Targets

Lee Ward of Sandia, Tyce McClarty of LLNL, and I were the I/O reps at HEC

Overwhelming idea in the I/O area was

If we get all these scalable and parallel file systems, and if we get devices that are smart, and if we want to extend the idea of PIM all the way to the storage devices, and if we have a secure way to ask devices to work for us, and if we have interesting OBSD's, and if we get NFS to be a good file system client for SAN and OBFS, etc.

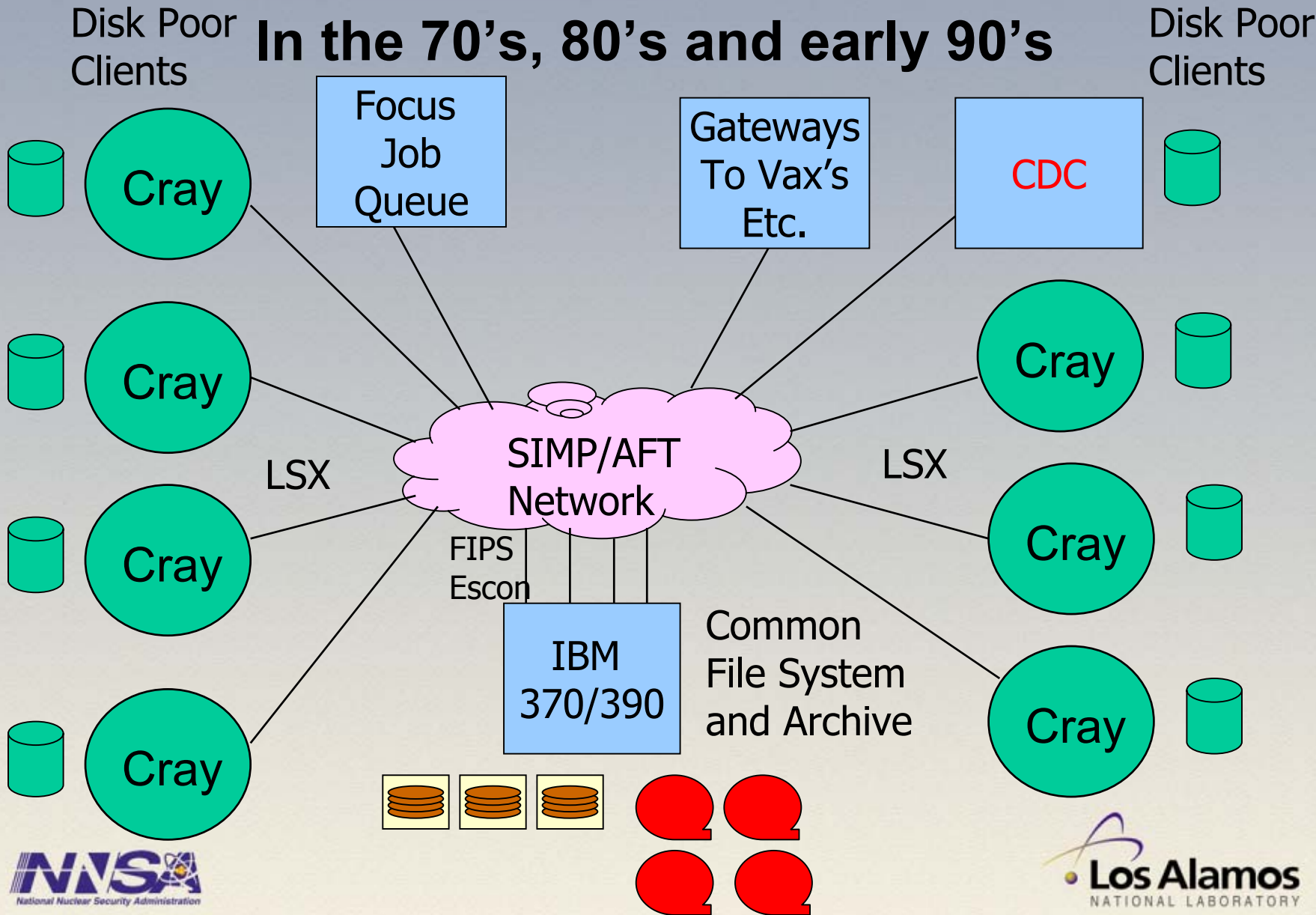
How can we possibly utilize such things if we are forced to go through an the current POSIX (open, read, seek, write, close)?

Furthermore, are trees the way to organize a baZILLION files? Is readdir and stat going to be able to service us much longer?

So, should we create a new I/O api legacy on which to build for the next several decades?

We are spinning up a "enhance Posix" effort as we speak!

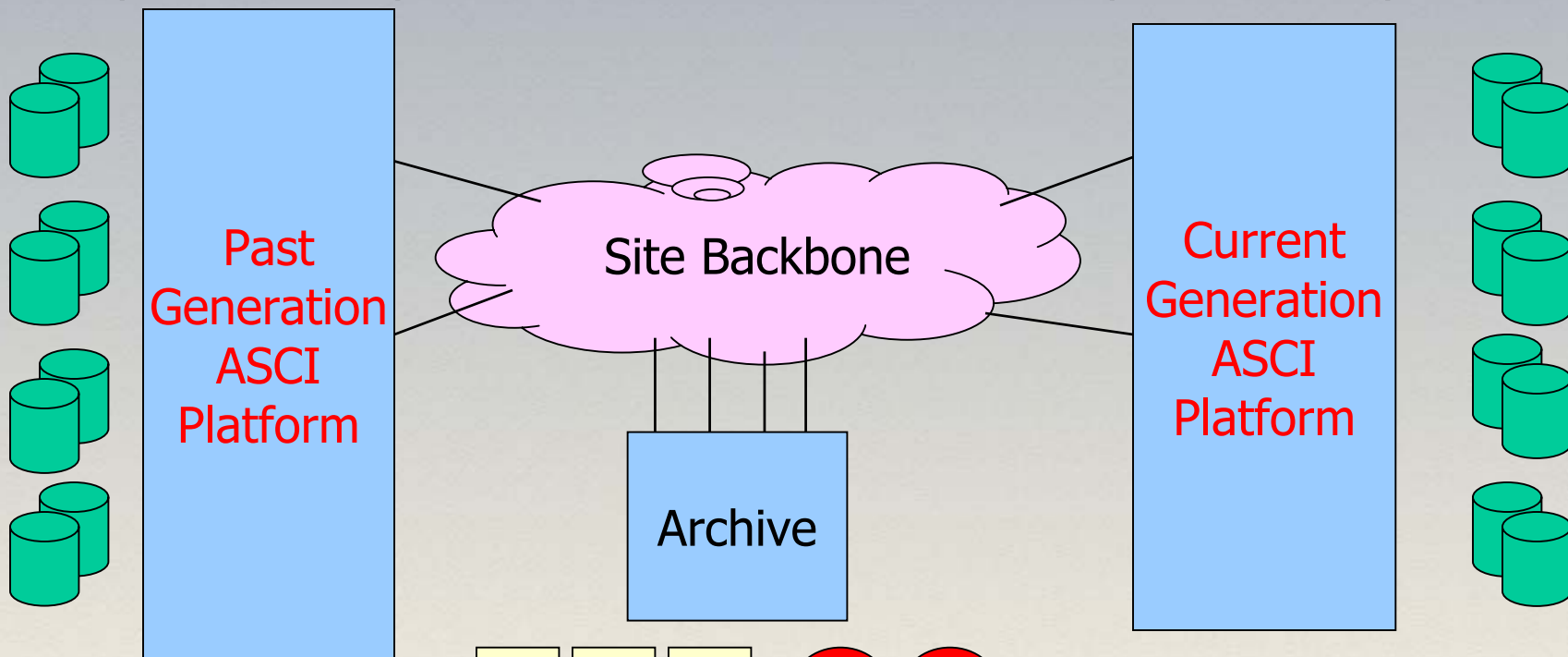
# In the 70's, 80's and early 90's



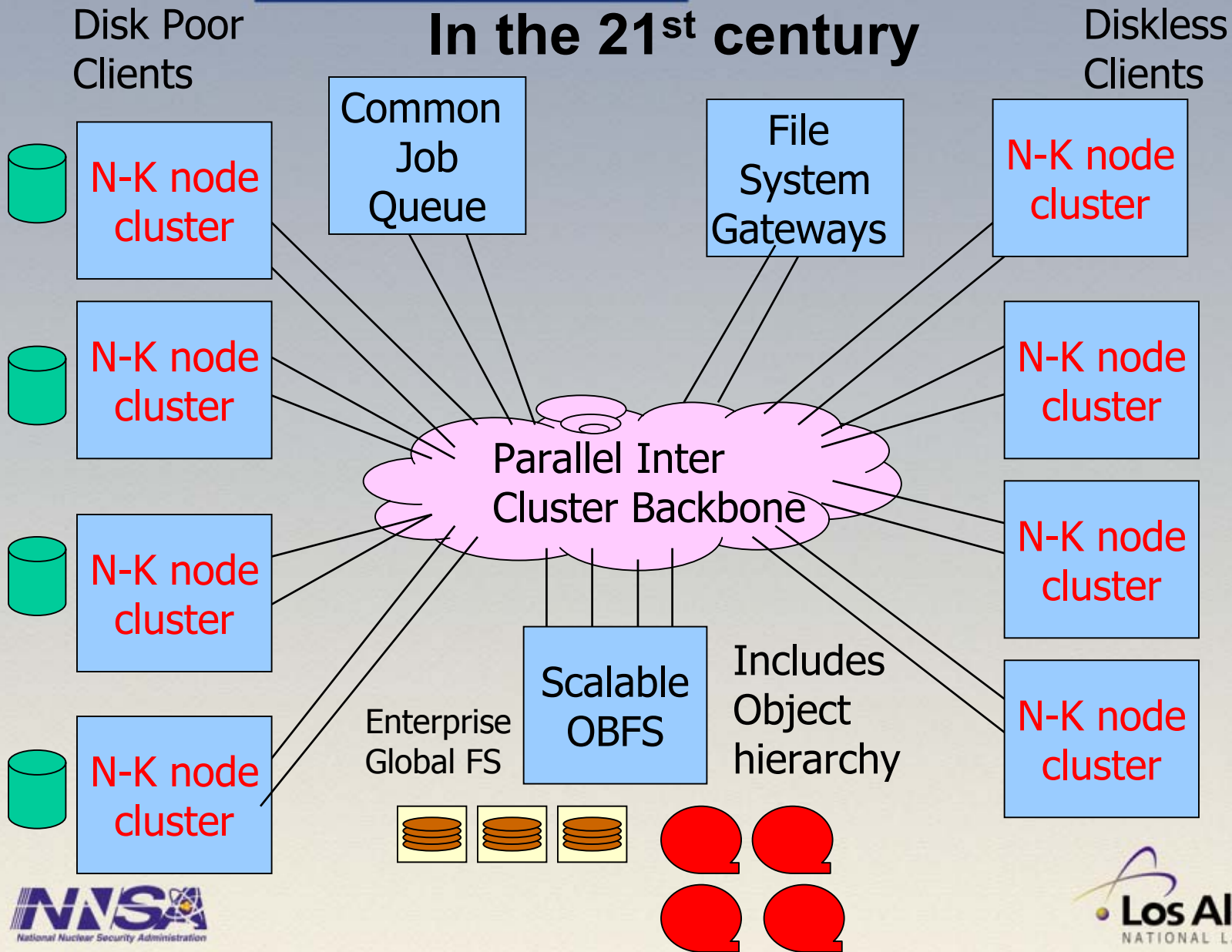
# In the Late 90's with ASCI

Disk Rich  
Supercomputer  
With its own  
parallel file system

Disk Rich  
Supercomputer  
With its own  
parallel file system



# In the 21<sup>st</sup> century



## Other interesting prototypes developed!

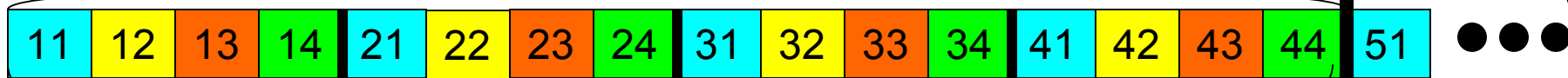
- Rait in Software
  - ◆ N to N Rait in software (implements raid 3 n+1)
    - ◆ Dialable width, block, etc.
    - ◆ Recovers from lost stripe
    - ◆ Scale tested 2+1 to 48+1, up to 2 GBytes/sec keeping pipes full at > 40 MB/sec and building RAIT stripe
    - ◆ Could be 1 to N Rait quite easily
    - ◆ IT WORKS!
    - ◆ Will likely add diagonal parity to protect against 2 stripe loss!
- Cross box parallel ls/find/tar
  - ◆ **Truly parallel, not just threads, parallel cross box or on box or both**
  - ◆ Does stat() ls/find in complete parallel, works for any shaped tree, even grass
  - ◆ Creates parallel lists of files, which piped into gtar makes truly completely parallel tar
  - ◆ 100's of file/sec in parallel, with good SGDFS class file system 1000's of files/sec
  - ◆ IT WORKS!

# Prototype Design

Parallel file

Superblock 1

Superblock 2



MPI R1  
 MPI-Fopen  
 Loop--N=0 to EOF-block  
 MPI-setview  
 (superblock-N+rank)  
 MPI-Fread (1,2,3,4)  
 calculate cross parity  
 MPI-Send  
 (distribute blocks+parity)  
 end loop -----



MPI R2  
 same as R1



MPI R3  
 same as R1

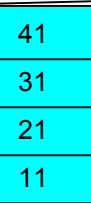


MPI R4  
 same as R1



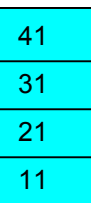
switch

MPI R5  
 open socket  
 MPI-recv  
 writesocket



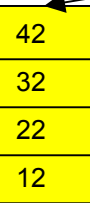
socket

mover 1  
 open socket  
 read socket  
 write device



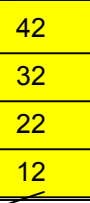
1

MPI R6  
 open socket  
 MPI-recv  
 writesocket



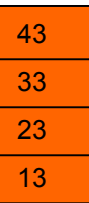
socket

mover 2  
 open socket  
 read socket  
 write device



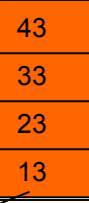
2

MPI R7  
 open socket  
 MPI-recv  
 writesocket



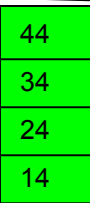
socket

mover 3  
 open socket  
 read socket  
 write device



3

MPI R8  
 open socket  
 MPI-recv  
 writesocket



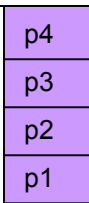
socket

mover 4  
 open socket  
 read socket  
 write device



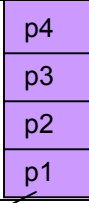
4

MPI R9  
 open socket  
 MPI-recv  
 writesocket



socket

mover p  
 open socket  
 read socket  
 write device



p

## Summary

**We have been at this SIO game for a while**

**We have been building strong leveragable partnerships**

**We have shaped standards (ANSI T10, IETF NFS4, POSIX)**

**Linux and Enterprise class secure global parallel file systems being deployed now and growing**

- ◆ **Lustre in use, moving into programmatic role**
- ◆ **Panasas in use, moving into programmatic role**
- ◆ **Building relationship with Storage Tank**
- ◆ **Headed towards enterprise class common parallel file systems**