

Lessons Learned from Proprietary HPC Cluster Software

James B. White III (Trey)

Richard A. Alexander

Linux Clusters: The HPC Revolution

June 27, 2001

The Center for Computational Sciences

DOE High Performance Computing Research Center at Oak Ridge National Laboratory

Outline

- **CCS proprietary clusters**
- **Cluster system software**
 - **System administration**
 - **Configuration-file maintenance**
 - **Security and account management**
 - **File systems**
 - **Resource management**
- **Conclusions**

CCS proprietary clusters

- **IBM RS/6000 SP**
 - 184 Winterhawk-II nodes (4-way SMPs)
 - SP switch
 - AIX 4.3, Parallel Environment 3.1
- **Compaq AlphaServer SC40**
 - 64 + 16 ES40 nodes (4-way SMPs)
 - Quadrics interconnect
 - Tru64 5.0, SC 1.0 (Tru64 5.1, SC 2.0)

System administration

- **smit** (System Management Interface Tool)
 - GUI tool for system administration
 - Generates scripts from clicks
 - Scripts can be reused and edited
- Distributed shells
 - **dsh** – simple, versatile, effective
 - **pdsh** – open-source equivalent
(from Jim Garlick of LLNL)

Configuration-file maintenance

- How to avoid configuration “skew”
- Single system image
 - **CFS** (Cluster File System) on ≤ 32 nodes
- Many system images: **supper**
 - Automatically copies files to all nodes
- Cfengine
 - Open-source **supper** equivalent
 - More versatile

Security and account management: Requirements

- **Centralized account database**
- **Synchronized across clusters**
- **Changes propagate immediately**
- **Modifiable from any node**
- **Highly available**
- **With secure authentication and account modification**

Solution: DCE

- **Distributed Computing Environment**
- **Meets all our requirements, and more**
 - **Secure threads and remote procedure calls**
 - **Directory service**
 - **Security service (Kerberos 5)**
 - **Distributed time service**
- **Project of the Open Group**
 - **Source available, but not really “open”**

Problem: DCE

- **Very complex**
- **Requires much time and effort**
- **AIX product for years, but...**
 - **Stability problems remain**
 - **Quiet failures on random nodes**
- **Newer and even less stable on Tru64**
- **Dying?**
- ***Time for an “open” replacement?***

File systems: Categories

- **Root operating-system (OS) files**
- **User home directories**
- **High-performance “scratch” space**
- **Archival storage**

Root OS files

- **Requirements**
 - Read-only access to many small files
 - Fast write access to node-specific files and virtual-memory pages
- **JFS (Journaled File System)**
 - Separate local file system for each node
 - Brute force
 - Configuration skew
- **CFS (Cluster File System)**

CFS

- **Root file system shared among nodes**
 - Supports node-specific files
- **Only scales to 32 nodes**
 - Large systems have multiple CFS domains
 - No tools to manage domains
- **Makes SC more fragile**
- **Breaks POSIX semantics**
 - `rm` applied to node-specific files

Better solution?

- **Use two file systems for OS files**
- **Cached network file system for read-only**
- **Fast, local file system for write**

User home directories: Requirements

- **Low-latency access to files**
- **High-availability access across the center**
- **Client-side caching**
- **Support for memory mapping**
- **Per-user quotas**
- **Non-intrusive backups**
- **Non-intrusive storage management**
- **Security**

Solution: DFS

- **Distributed File Service**
 - DCE application
- **Meets all requirements**
- **ACLs (access-control lists)**
- **Location-independent naming**
 - Moving file sets doesn't affect DFS clients
- **Read-only clones**
 - `$HOME/yesterday/`

Problem: DFS

- DCE application
- Overly complicated ACLs
 - Confusing translation to Unix permissions
- Random (though rare) instability on **SP**
- Not yet supported on **SC**
 - Per-node clients break CFS model
 - DFS to NFS to CFS (slow!)
 - Requires “occasional” **dfs_login**

Other solutions?

- **AFS**
 - **DFS without DCE**
 - **Open source**
 - **Not integrated, not developed**
- **NFS v3**
 - **Security?**
 - **Location-dependent naming**
- **NFS v4**
 - **Many AFS/DFS features**

High-performance “scratch” space: Requirements

- **High-bandwidth access to large files**
- **No backups?**
- **No memory mapping?**

GPFS (General Parallel File System)

- **Files striped across SP server nodes**
- **Big bandwidth**
- **Client and server caching**
- **Improved metadata and small-file access**
- **Fairly robust**
 - **NERSC uses GPFS for home directories**
- **Single SP only**
 - **Can be NFS/DFS exported**

SCFS (SC File System)

- **SC v2.0 (just released)**
- **Fast client for access to “remote” CFS**
 - **One CFS domain SCFS-exports to others**
 - **Native over Quadrics interconnect**
 - **All within the same SC**
- **Server caching only**
- **No striping over servers**
- **Stability?**

PFS (Parallel File System)

- **Striped across underlying file systems**
 - CFS
 - SCFS (v2.0)
- **No memory mapping**
- **Performance?**
- **Stability?**

Archival storage: Requirements

- **High-bandwidth access to LARGE files**
- **Many such files**
- **High availability center wide**
- **Password-free access from local systems**
- **Non-intrusive backups**
- **Non-intrusive management of storage media**
- **Security**

Solution: HPSS

- **High Performance Storage System**
- **Developed by IBM and DOE labs**
- **ftp, pftp, DFS, hsi**
- **Multiple tape silos with disk caching**
- **Password-free access using Kerberos**
- **More scalable than SGI/Cray Data Migration Facility (?)**

Problem: HPSS

- **Complex, uses DCE internally**
- **Not highly available**
 - **Requires weekly down time for maintenance**
 - **Supercomputers are *not* down**
 - **Batch jobs cannot assume HPSS is available**
- **Possible solution: HPSS spooler**
 - **Integrated with “scratch” system**
 - **Selected files are copied asynchronously**
 - **Deleted after successful copy to HPSS**

Resource management

- **Parallel job...**
 - **Control**
 - **Scheduling**
 - **Monitoring**
 - **Accounting**
- **Examples**
 - **LoadLeveler**
 - **RMS (Resource Management System)**

Resource management: Requirements

- **Control over interactive and batch jobs**
- **Backfill scheduling**
- **Job reservations**
- **Fair-share scheduling**
- **Event-driven scheduling**
- **Full integration with HPC resources**

Resource management: Requirements

- **Effective limits**
- **System and user prologs and epilogs**
- **Common interface across systems**
- **Centralized management and accounting**
- **Resource “banking”**
- **Distribution of DCE/Kerberos credentials**

RMS

- **Interactive only**
- **No backfill, prologs, epilogs, banking**
- **Does too little to be a batch system**
- **Does too much to easily integrate with existing batch systems (PBS, LSF)**
- **Incomplete distribution of credentials**
 - **None in v1.0**
 - **Kludge in v2.0**

LoadLeveler

- **No reservations or fair share**
 - Maui scheduler has these
 - But external-scheduler API isn't event driven
- **Copy of config files on each node**
- **No integrated “banking”**
- **No max usage of nodes by a user**
 - Max job count, max job size
- **Only supports AIX systems**
- **Credentials?**

Kerberos credentials done “right”

- **Login creds are forwardable & renewable**
- **Batch submission forwards creds**
- **Resource manager renews waiting creds**
- **Resource manager forwards creds to parallel nodes**
- **All steps use Kerberos API**

Kerberos credentials in reality

- **LoadLeveler** cheats by default
 - Makes cheap photocopy of creds
 - Doesn't work for general Kerberized apps
- **Newest LoadLeveler** can do it “right”
 - Fully DCE-ized
 - Needs multiple DCE entries for each node!
More trouble than its worth!
- **Is Kerberos the right security model for batch systems? If not, what is?**

Conclusions: System administration

- **Emphasize**
 - **Simplicity**
 - **Scriptability**
 - **Automation**
 - **Fault tolerance**
- **Avoid**
 - **Complex GUIs**

Conclusions: Security and account management

- **Replacement for DCE?**
- **Emphasize**
 - **Integration**
 - **Existing industry standards**
 - **Fault tolerance**
 - **Client availability over server capability**
- **What is the right security model for batch?**

Conclusions: File systems

- **We need**
 - **Performance scalability of GPFS**
 - **Security and maintenance scalability of DFS**
 - **Cross-platform support of NFS**
 - **With automatic data-migration capability**
- **Make it so!**
 - **Technology exists**
 - **Needs single vision and much development**
 - **Start with NFS v4?**

Conclusions: Resource management

- **We have straightforward requirements**
 - Other centers have very similar requirements
 - LoadLeveler, PBS, LSF, Condor, NQE, DQS, ... none meet all requirements
- **We need a *de facto* standard that**
 - Meets requirements
 - Is simple
 - Is fault tolerant
- ***Why is this still an issue?***

Contact information

Trey White & Richard Alexander

`whitejbiii@ornl.gov` `alexanderr@ornl.gov`

`http://www.ccs.ornl.gov/`

Cfengine

`http://www.iu.hioslo.no/cfengine/`

pdsh

`http://www.ecst.csuchico.edu/~garlick/pdsh/`

Maui scheduler

`http://supercluster.org/projects/maui/`

Qbank

`http://www.emsl.pnl.gov:2080/docs/mscf/qbank-2.8/`

The Center for Computational Sciences

DOE High Performance Computing Research Center at Oak Ridge National Laboratory