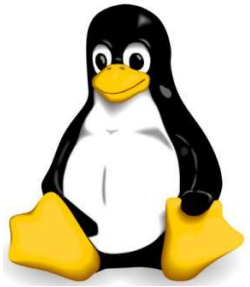


Linux Cluster

- **Linux Cluster Introduction**
  - **-Clustering technology**
  - **-Architecture and components**
  - **-Classifications**
- NFS/NIS
- Resource Management and Scheduling
- MPI/OpenMP

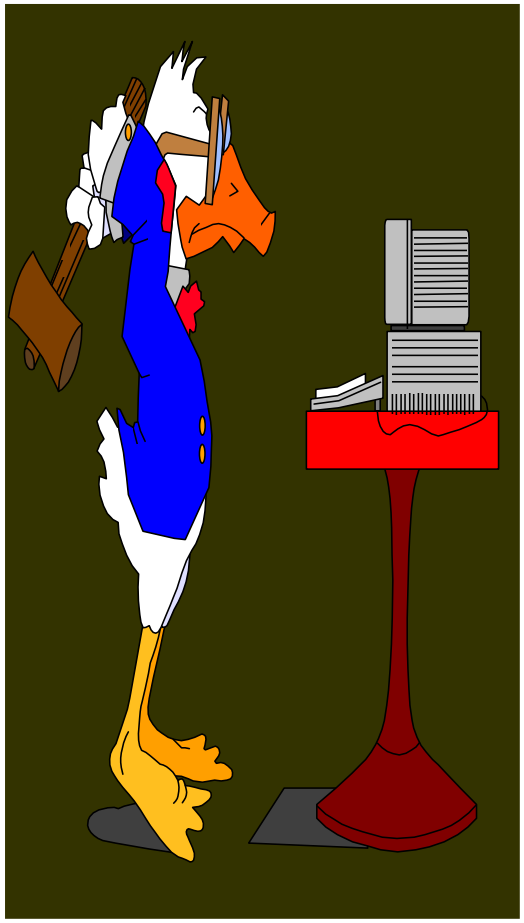
What is Cluster ?

- A cluster is a type of parallel and distributed processing system, which consists of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource.
- A node
  - a standalone single or multiprocessor system with memory, I/O facilities & OS, it can handle independent user workloads
- A cluster:
  - generally 2 or more computers (nodes) connected together
  - in a single cabinet, or physically separated & connected via a LAN (local area network)
  - a shared storge usually provides persistent storage for user files and directories, user programs, input data, and result data associated with the jobs that run on the cluster
  - appear as a single system to users and applications
  - provide a cost-effective way to gain features and benefits

Key Operational Benefits of Clustering

- High Performance
- Expandability and Scalability
- High Throughput
- High Availability

Recap: Major issues in Cluster design



- Enhanced Performance (performance @ low cost)
- Enhanced Availability (failure management)
- Single System Image (look-and-feel of one system)
- Size Scalability (physical & application)
- Fast Communication (networks & protocols)
- Load Balancing (CPU, Net, Memory, Disk)
- Security and Encryption (clusters of clusters)
- Distributed Environment (Social issues)
- Manageability (admin. And control)
- Programmability (simple API if required)
- Applicability (cluster-aware and non-aware app.)

Clustering Technologies

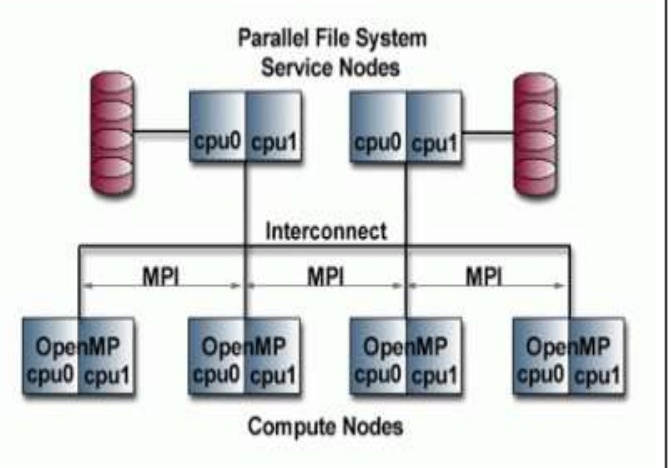
- **What makes a bunch of computers on a network to become a computational cluster**

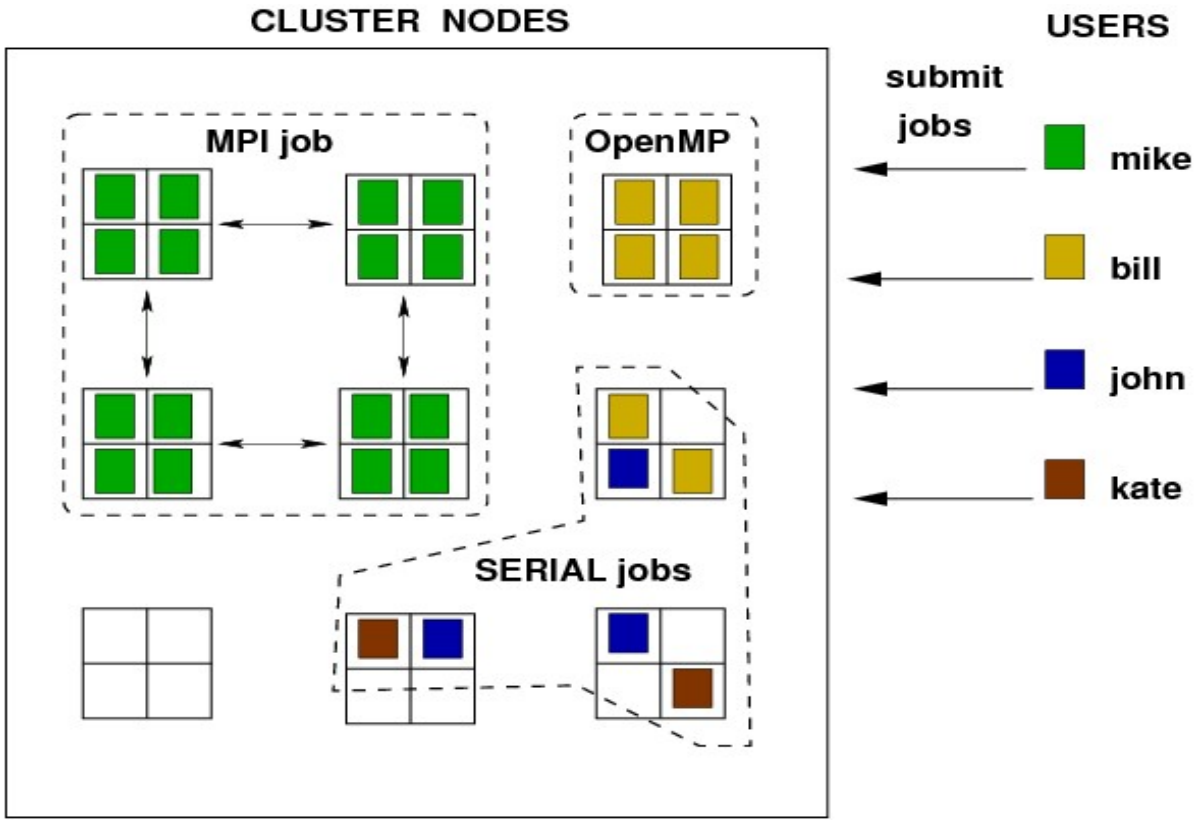
Single system Image (SSI).
For example, OpenMosix. SSI is smart system software that spreads operating system functions across systems and involves modifications of the Linux kernel.

OR/AND **Single System Environment (SSE)**
A smart system software that runs in user space as layered service. SSE includes, for example:
API libraries for programming, such as PVM and MPI
Queue scheduling system (for example, LSF, Condor, PBS and Sun Grid Engine)

Job scheduling and management

- Departmental HPC clusters are usually shared between several people and run various types of computations, including parallel and serial.





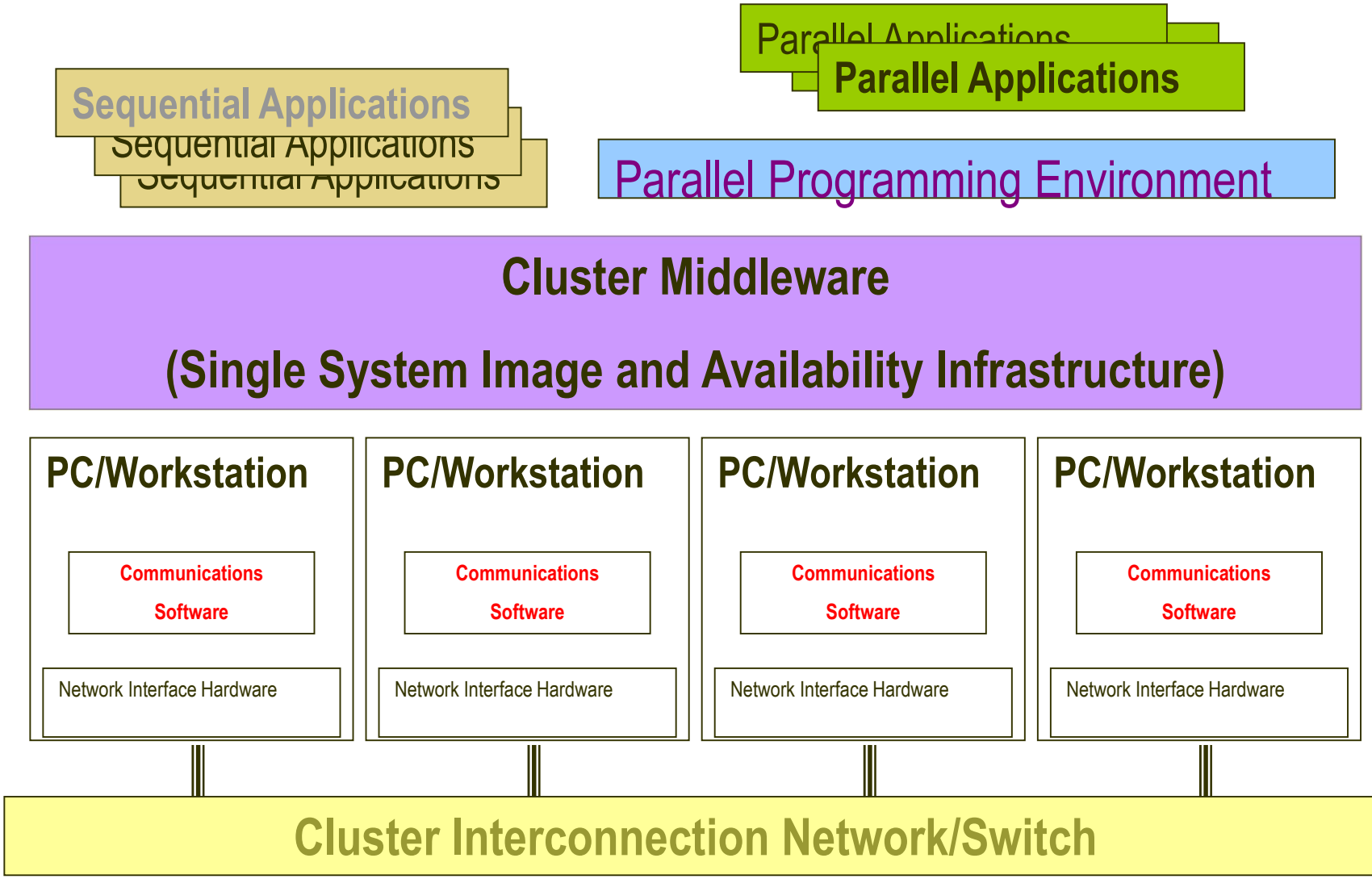
- On multi-user and multi-tasking cluster the resources should be managed through a queue scheduling system, such as **SGE, Torque, Maui, LSF, PBS, or Condor**, in order to avoid resource conflicts, bottle necks, and race conditions between user jobs

Comparison of cluster computing systems.

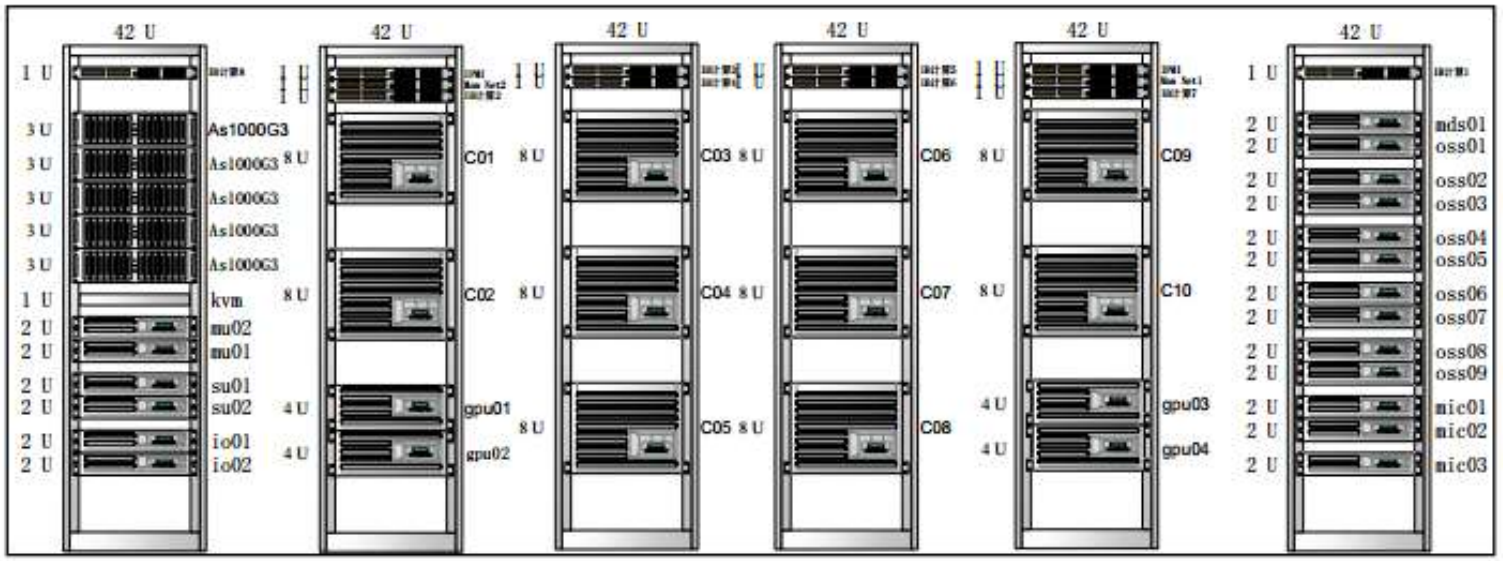
System	Job processing type	QoS attributes	Job composition	Resource allocation control	Platform support	Evaluation method	Process migration
Enhanced MOSIX [12]	Parallel	Cost	Single task	Decentralized	Heterogeneous	User-centric	Yes
Gluster [14]	Parallel	Reliability (no point of failure)	Parallel task	Decentralized	Heterogeneous	N/A	Yes
Faucets [116]	Parallel	Time, cost	Parallel task	Centralized	Heterogeneous	System-centric	Yes
DQS [19]	Batch	CPU memory sizes, hardware architecture and OS versions.	Parallel Task	Decentralized	Heterogeneous	System-centric	No
Tycoon [20]	Sequential	Time, cost	Multiple task	Decentralized	Heterogeneous	User-centric	No
Cluster-on-demand [22]	Sequential	Cost in terms of time	Independent	Decentralized	Heterogeneous	User-centric	No
Kerrighed [23,24]	Sequential	Ease of use, high performance, high availability, efficient resources management, and high customizability of the OS	Multiple task	Decentralized	Homogeneous	System-centric	Yes
OpenSSI [25]	Parallel	Availability, scalability and manageability	Multiple task	Centralized	Heterogeneous	System-centric	Yes
Libra [26]	Batch, sequential	Time, cost	Parallel	Centralized	Heterogeneous	System-centric, User-centric	Yes
PVM [28]	Parallel, concurrent	Cost	Multiple task	Centralized	Heterogeneous	User-centric	Yes
Condor [50,51]	Parallel	Throughput, productivity of computing environment	Multiple task	Centralized	Platform support	System-centric	Yes
REXEC [30]	Parallel, sequential	Cost	Independent, single task	Decentralized	Homogeneous	User-centric	No
GNQS [31]	Batch, parallel	Computing power	Parallel processing	Centralized	Heterogeneous	System-centric	No
LoadLeveler [32]	Parallel	Time, high availability	Multiple task	Centralized	Heterogeneous	System-centric	Yes
LSF [90]	Parallel, Batch	Job submission simplification, setup time reduction and operation errors	Multiple task	Centralized	Heterogeneous	System-centric	Yes
SLURM [91]	Parallel	Simplicity, scalability, portability and fault tolerance	Multiple task	Centralized	Homogeneous	System-centric, User-centric	No
PBS [92]	Batch	Time, jobs queuing	Multiple task	Centralized	Heterogeneous	System-centric	Yes

- **Hameed Hussain, et al., A survey on resource allocation in high performance distributed computing systems, Parallel Computing 39 (2013) 709–736**

Cluster Computer Architecture



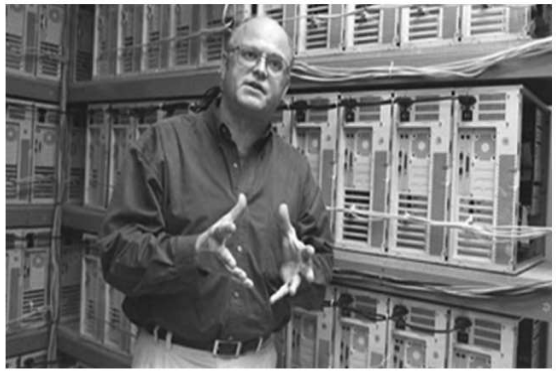
Inspur Cluster @ NPU



Cluster Components

Prominent Components of Cluster Computers (I)

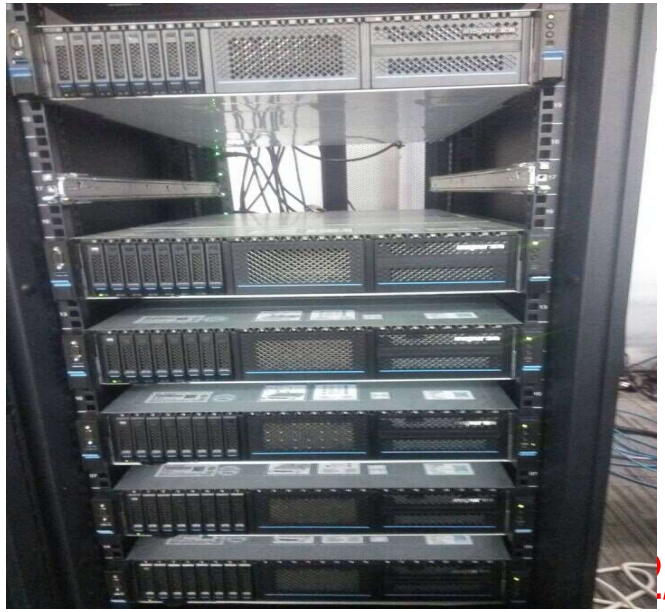
- **Multiple High Performance Computers/Nodes**
  - PCs
  - Workstations(COW)
  - SMPs (CLUMPS)
  - Blade servers
  - Distributed HPC Systems leading to Grid Computing



Thomas Sterling in front of a commodity cluster built as part of the Beowulf Project. Such commodity clusters are now frequently referred to as belonging to the Beowulf class of supercomputer.

- **Different functions**
  - login
  - management
  - I/O
  - Computing

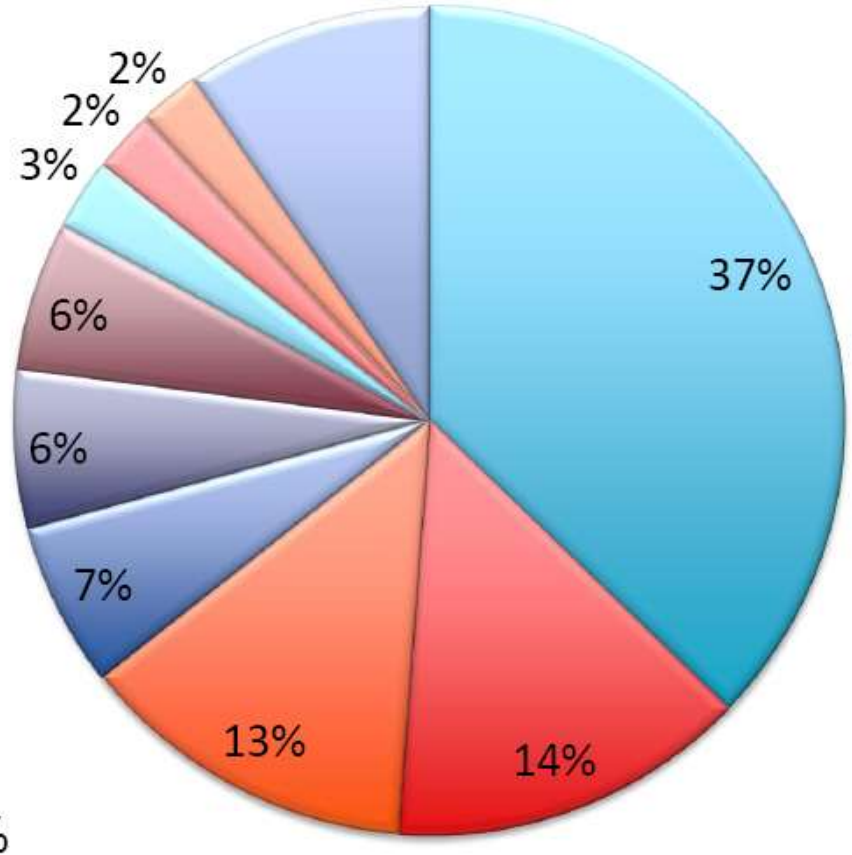
a self-built cluster at ASC student supercomputer challenge



System CPUs

- Processors
  - Intel x86-class Processors
    - Pentium Pro and Pentium Xeon
    - AMD x86, Cyrix x86, etc.
  - Digital Alpha – phased out when HP acquired it.
    - Alpha 21364 processor integrates processing, memory controller, network interface into a single chip
  - IBM PowerPC
  - Sun SPARC
    - Scalable Processor Architecture
  - SGI MIPS
    - Microprocessor without Interlocked Pipeline Stages

Processors / Systems



Intel 71%
AMD 13%
IBM 7%

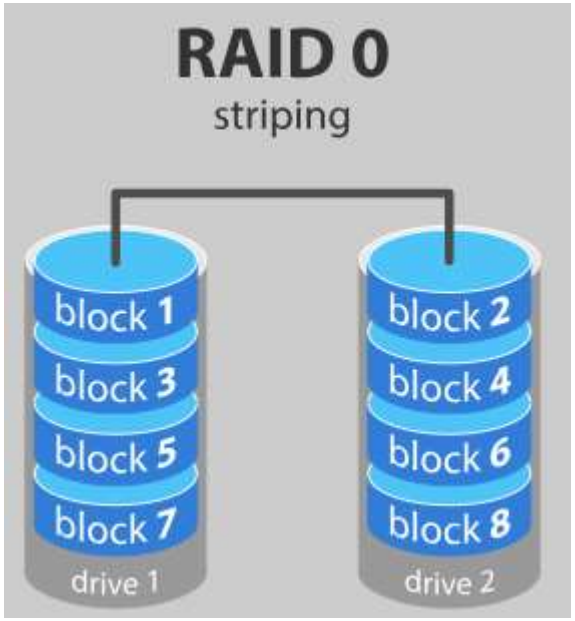
- Xeon E54xx (Harpertown)
- Xeon 51xx (Woodcrest)
- Xeon 53xx (Clovertown)
- Xeon L54xx (Harpertown)
- Opteron Quad Core
- Opteron Dual Core
- PowerPC 440
- PowerPC 450
- POWER6
- Others

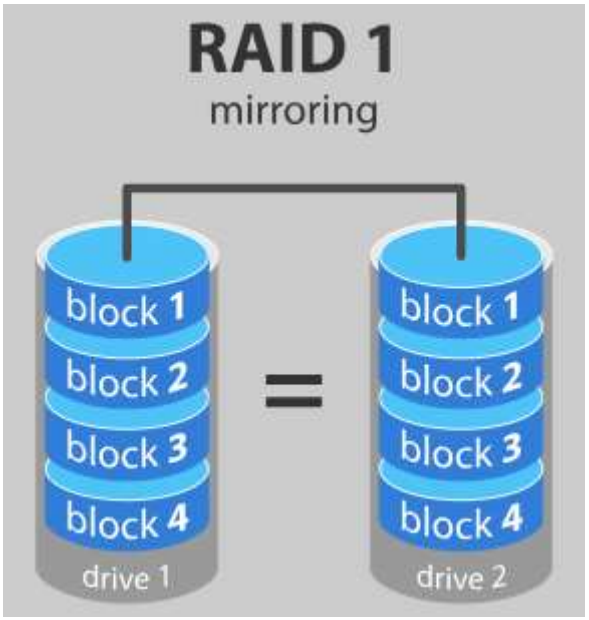
Processor Generation	Count	System Share (%)	Rmax (GFlops)	Rpeak (GFlops)	Cores
Intel Xeon E5 (Haswell)	216	43.2	218,747,944	348,737,038	9,910,640
Intel Xeon E5 (Broadwell)	132	26.4	144,531,684	208,484,155	8,317,396
Intel Xeon E5 (IvyBridge)	61	12.2	89,983,462	146,722,123	6,814,650
Intel Xeon E5 (SandyBridge)	22	4.4	25,810,861	35,937,149	1,876,780
Power BQC	16	3.2	45,974,153	53,896,809	4,210,688
Intel Xeon Phi	13	2.6	57,586,248	105,432,729	2,412,816
Xeon 5600-series (Westmere-EP)	6	1.2	8,757,393	17,605,405	555,268
SPARC64 XIfx	5	1	10,422,200	11,530,310	354,384
Intel Xeon E7 (Haswell-Ex)	4	0.8	2,366,376	4,124,774	114,528
Opteron 6200 Series "Interlagos"	4	0.8	19,920,100	30,334,435	820,280
POWER7	3	0.6	2,456,600	2,949,481	98,784
Xeon 5500-series (Nehalem-EP)	2	0.4	930,000	1,510,046	71,880
IBM Power8+	2	0.4	1,114,400	1,686,430	15,536
Intel Xeon E7 (IvyBridge)	2	0.4	1,104,900	2,034,432	112,320
Intel Xeon E7 (Broadwell)	2	0.4	1,975,615	3,345,408	95,040
Xeon Gold	2	0.4	2,465,257	3,790,848	49,360
Opteron 6300 Series "Abu Dhabi"	1	0.2	474,000	2,620,800	252,000
ShenWei	1	0.2	795,900	1,070,160	137,200
SPARC64 IXfx	1	0.2	1,043,000	1,135,411	76,800
Xeon 5500-series (Nehalem-EX)	1	0.2	1,050,000	1,254,550	138,368
Opteron 6100-series "Magny-Cours"	1	0.2	1,110,000	1,365,811	142,272
Xeon Platinum	1	0.2	6,227,200	9,957,427	148,176
SPARC64 VIIIfx	1	0.2	10,510,000	11,280,384	705,024
Sunway	1	0.2	93,014,594	125,435,904	10,649,600

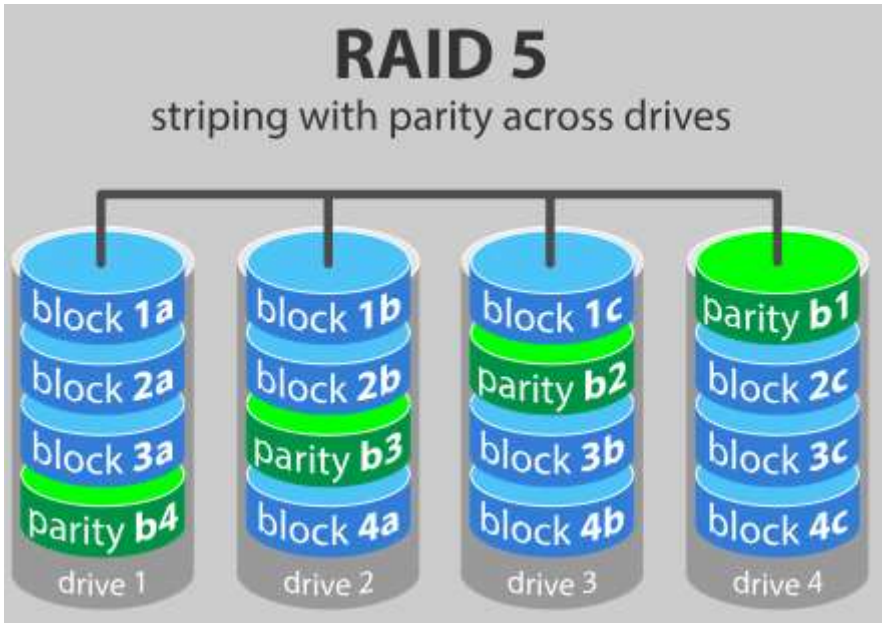
System Disk

- **Disk and I/O** (Local disk in a node + shared storage)
  - Overall improvement in disk access time has been less than 10% per year
  - Speed-up obtained from faster processors is limited by the slowest system component
  - Parallel I/O
    - Carry out I/O operations in parallel, supported by parallel file system based on hardware or software RAID (Redundant Arrays of Independent Disks)

RAID (Redundant Array of Inexpensive Disks)





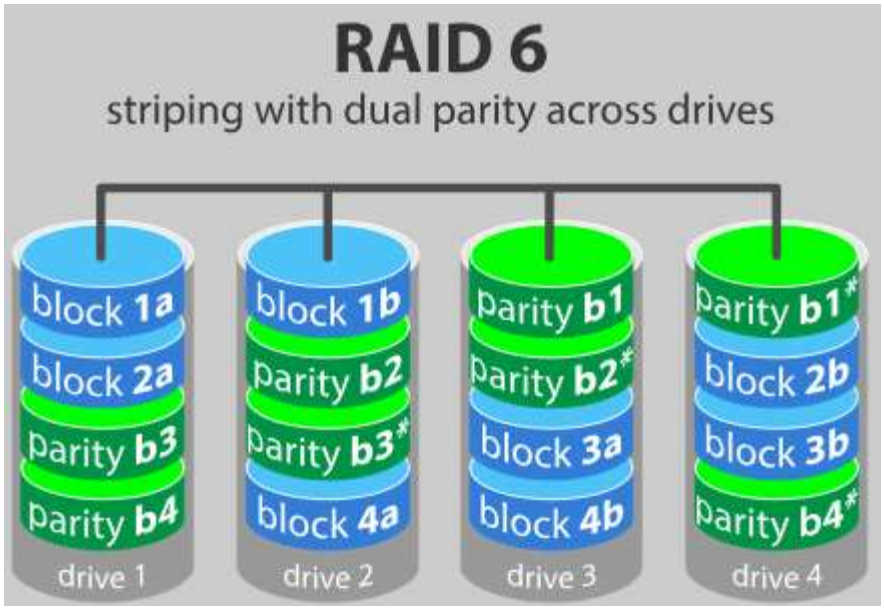


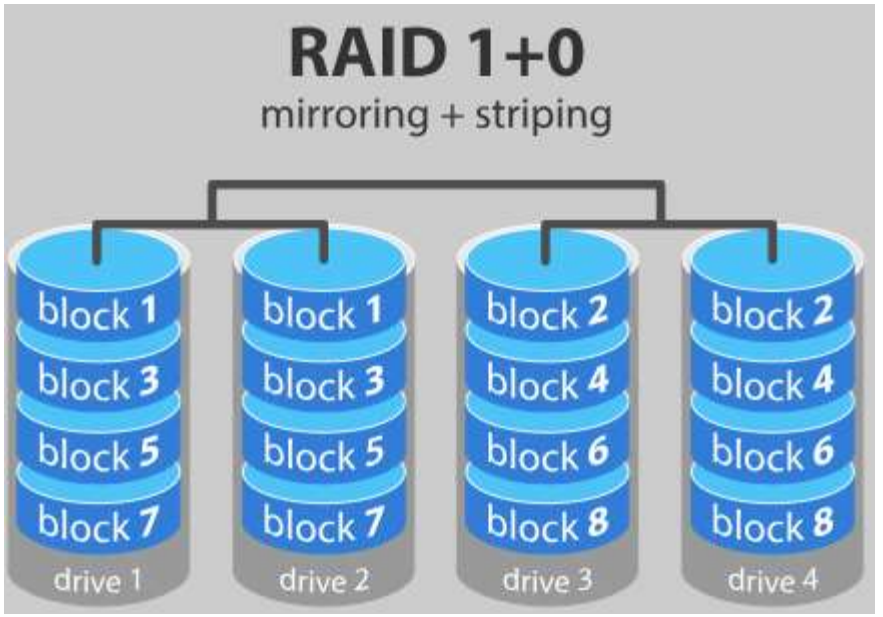
Advantages

- Read data transactions are very fast while write data transactions are somewhat slower (due to the parity that has to be calculated).
- If a drive fails, you still have access to all data, even while the failed drive is being replaced and the storage controller rebuilds the data on the new drive.

Disadvantages

- Drive failures have an effect on throughput, although this is still acceptable.
- This is complex technology. If one of the disks in an array using 4TB disks fails and is replaced, restoring the data (the rebuild time) may take a day or longer, depending on the load on the array and the speed of the controller. If another disk goes bad during that time, data are lost forever.





Advantages

- Like with RAID 5, read data transactions are very fast.
- If two drives fail, you still have access to all data, even while the failed drives are being replaced. So RAID 6 is more secure than RAID 5.

Disadvantages

- Write data transactions are slower than RAID 5 due to the additional parity data that have to be calculated. In one report I read the write performance was 20% lower.
- Drive failures have an effect on throughput, although this is still acceptable.
- This is complex technology. Rebuilding an array in which one drive failed can take a long time.

Advantages

- If something goes wrong with one of the disks in a RAID 10 configuration, the rebuild time is very fast since all that is needed is copying all the data from the surviving mirror to a new drive. This can take as little as 30 minutes for drives of 1 TB.

Disadvantages

- Half of the storage capacity goes to mirroring, so compared to large RAID 5 or RAID 6 arrays, this is an expensive way to have redundancy.

Commodity Components for Clusters (II): Operating Systems

- Operating Systems
  - 2 fundamental services for users
    - make the computer hardware easier to use
      - create a virtual machine that differs markedly from the real machine
    - share hardware resources among users
      - Processor - multitasking
  - The new concept in OS services
    - support multiple threads of control in a process itself
      - parallelism within a process
      - multithreading
      - POSIX thread interface is a standard programming environment
  - Trend
    - Modularity – MS Windows, IBM OS/2
    - Microkernel – provide only essential OS services
      - high level abstraction of OS portability

- **State of the art Operating Systems**
  - **Linux** (MOSIX, Beowulf, and many more)
  - **Microsoft NT** (Illinois HPVM, Cornell Velocity)
  - **SUN Solaris** (Berkeley NOW, C-DAC PARAM)
  - **IBM AIX** (IBM SP2)
  - **HP UX** (Illinois - PANDA)
  - **Mach (Microkernel based OS) (CMU)**
  - **Cluster Operating Systems (Solaris MC, SCO Unixware, MOSIX (academic project)**
  - **OS gluing layers** (Berkeley Glunix)

linus torvalds

redhat. debian ubuntu

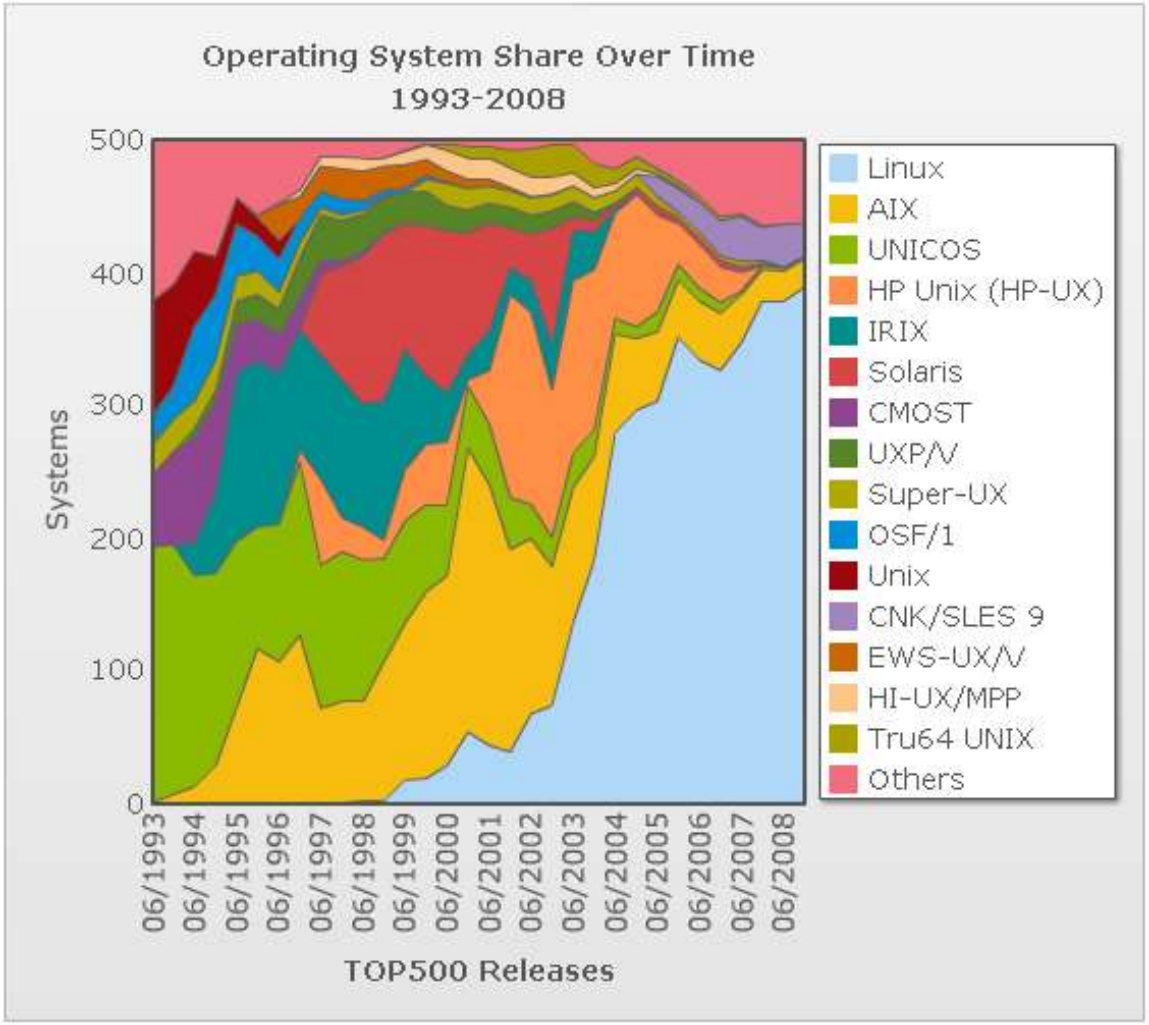
powered by: fedora linux

CentOS The Community ENTerprise Operating System

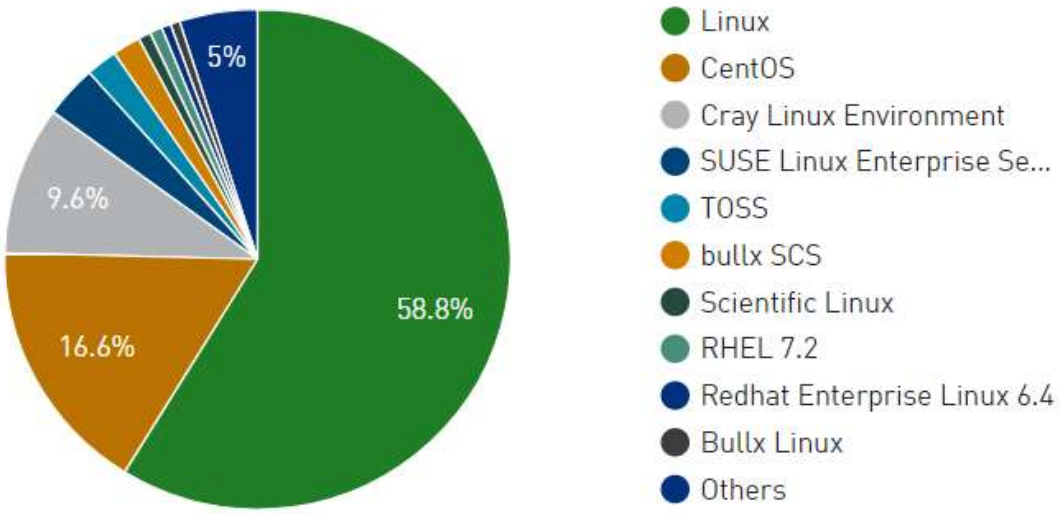
SuSE

红旗 Linux turbolinux

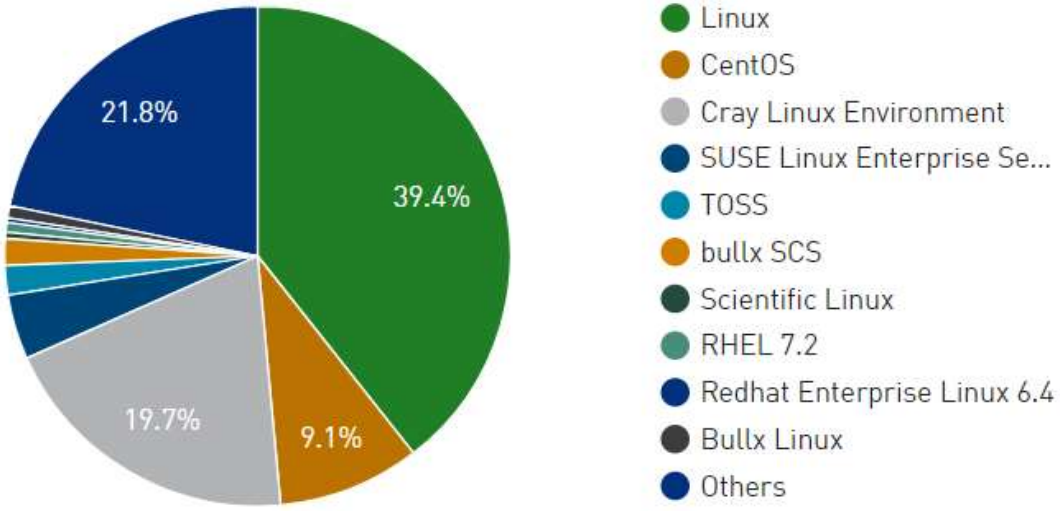
Operating Systems used in Top500 Powerful computers



Operating System System Share



Operating System Performance Share



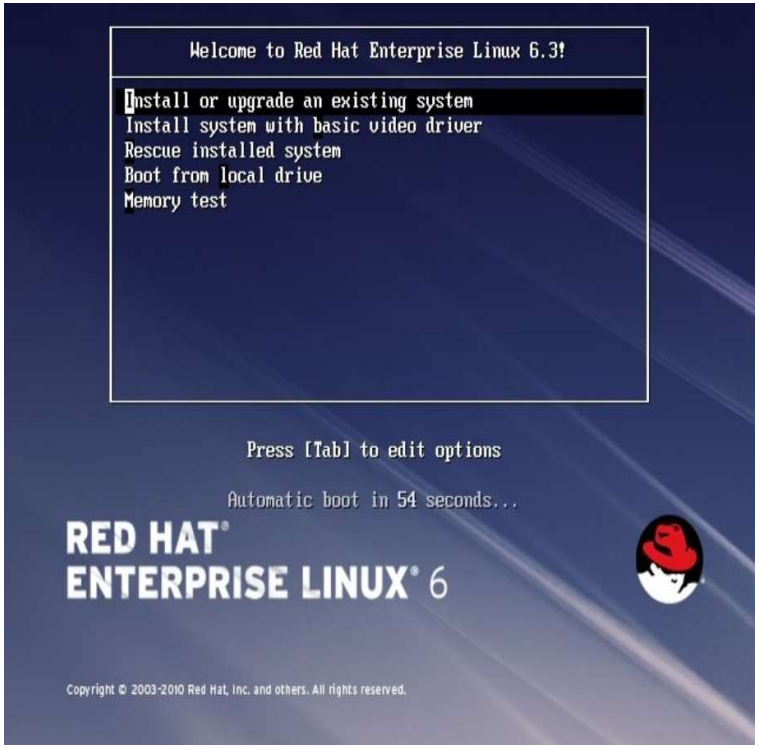
Operating System	Count	System Share (%)	Rmax (GFlops)	Rpeak (GFlops)	Cores
Linux	294	58.8	295,077,397	468,341,453	18,458,180
CentOS	83	16.6	68,234,142	126,617,437	6,455,356
Cray Linux Environment	48	9.6	147,748,346	210,095,979	5,363,588
SUSE Linux Enterprise Server 11	17	3.4	31,380,602	43,168,669	1,188,944
TOSS	10	2	14,228,087	16,573,455	496,584
bullx SCS	9	1.8	12,939,575	16,288,430	435,548
Scientific Linux	4	0.8	2,993,488	4,203,277	98,552
RHEL 7.2	4	0.8	4,738,901	5,395,687	149,300
Redhat Enterprise Linux 6.4	3	0.6	2,039,492	2,937,808	81,866
Bullx Linux	3	0.6	5,911,620	7,935,130	204,000
Ubuntu 14.04	3	0.6	4,434,300	6,712,960	82,960
SUSE Linux Enterprise Server 12 SP1	3	0.6	7,395,969	9,209,709	197,288
AIX	2	0.4	869,600	1,017,856	35,840
RHEL 6.8	2	0.4	1,384,140	1,556,890	46,336
bullx SUperCOmputer Suite A.E.2.1	2	0.4	2,596,000	3,191,270	147,744
Redhat Enterprise Linux 6	2	0.4	2,433,470	3,032,783	295,656
Redhat Enterprise Linux 6.5	2	0.4	2,987,745	4,115,251	105,216
Kylin Linux	2	0.4	35,934,090	57,976,934	3,294,720
Sunway RaiseOS 2.0.5	1	0.2	93,014,594	125,435,904	10,649,600
Redhat Enterprise Linux 7.2	1	0.2	459,830	508,032	15,120
Redhat Linux	1	0.2	460,200	694,886	4,736
RHEL 6.2	1	0.2	773,700	961,126	46,208
RHEL 7.3	1	0.2	802,400	1,417,152	17,760
Ubuntu Linux	1	0.2	3,307,000	4,896,512	60,512
SUSE Linux	1	0.2	6,227,200	9,957,427	148,176

Why do super computers use Linux?

- **1- Modular nature of Linux (can be modified)**
- **2- Generic Nature of Linux Kernel**
- **3- Scalability**
- **4- Open Source Nature**
- **5- Community Support**
- **6- Cost**
- Nice networking support
- Linux is more secure…

Red Hat Enterprise Linux

RHEL



Red Hat Enterprise Linux 9.0

- Introducing RHEL 9.0
- Download version 9.0
- Release Notes

Red Hat Enterprise Linux 8.6

- Introducing RHEL 8.6 GA
- Download version 8.6
- Release Notes

Red Hat Enterprise Linux 7.9

- Download version 7 or below
- Release Notes
- Red Hat Software Collections for RHEL 7

News

Announcing the GA release of Red Hat Enterprise Linux 9.0

May 18 2022 at 1:03 PM

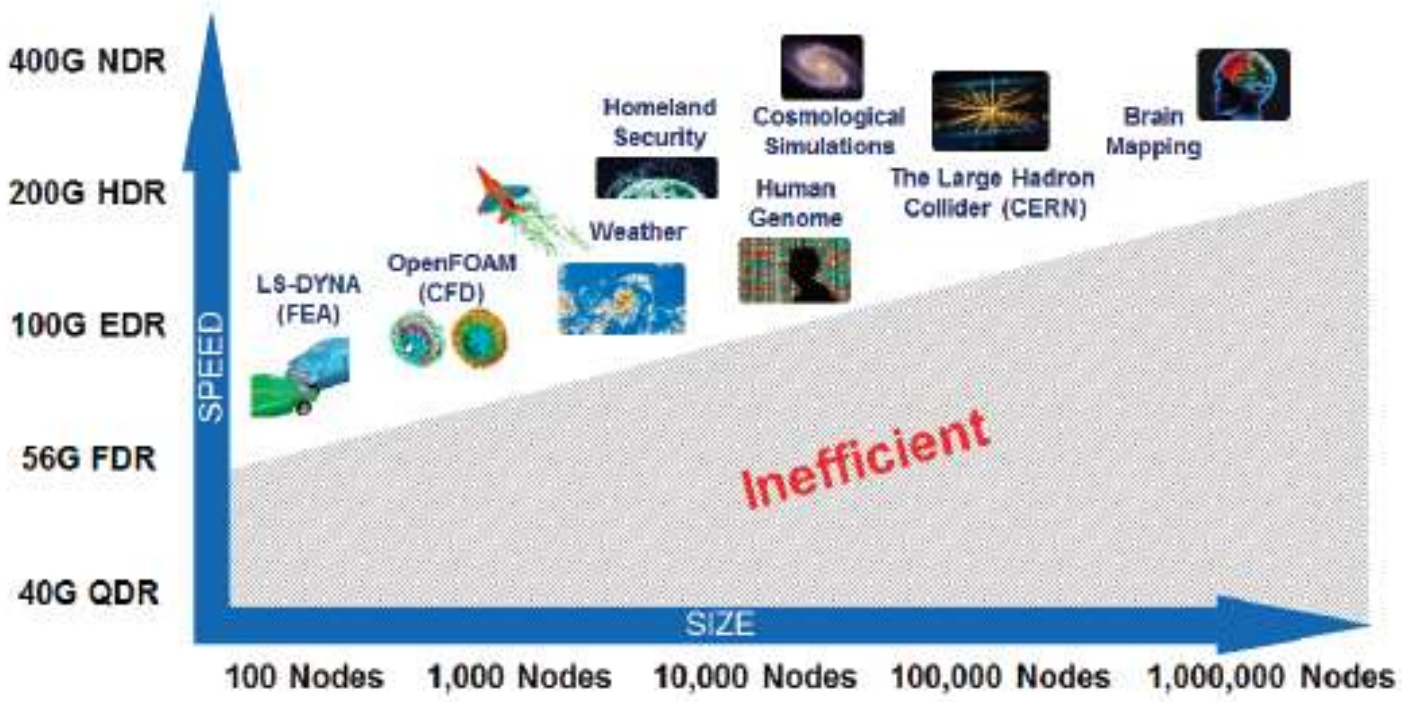
Announcing the GA release of Red Hat Enterprise Linux 8.6

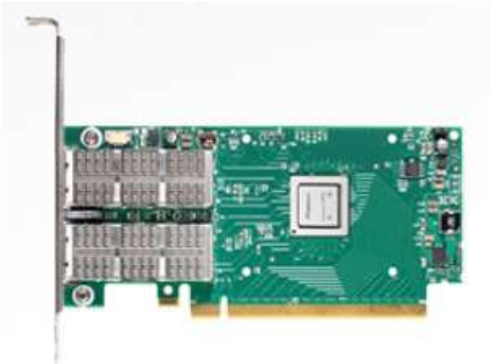
May 11 2022 at 1:42 PM

Prominent Components of Cluster Computers (III)

- **High Performance Networks/Switches**
  - Ethernet (10Mbps),
  - Fast Ethernet (100Mbps),
  - Gigabit Ethernet (1-10-100Gbps)
  - SCI (Scalable Coherent Interface- MPI- 12µsec latency)
  - ATM (Asynchronous Transfer Mode)
  - Myrinet (1.28Gbps)
  - QsNet (Quadrics Supercomputing World, 5µsec latency for MPI messages)
  - Digital Memory Channel
  - FDDI (fiber distributed data interface)
  - InfiniBand (40-100-200Gbps)

The demand for high speed network (Infiniband)





InfiniBand HCA 100 Gb 1/2 port EDR

Prominent Components of Cluster Computers (IV)

- **Cluster Middleware**
  - Single System Environment (SSE)
  - System Availability (SA) Infrastructure

- **Resource management and scheduling (RMS) software**
  - Sun SGE (Sun Grid Engine)
  - Platform LSF (Load Sharing Facility)
  - PBS (Portable Batch Scheduler)
  - Microsoft Cluster Compute Server (CCS)

Prominent Components of Cluster Computers (V)

- **File Systems**

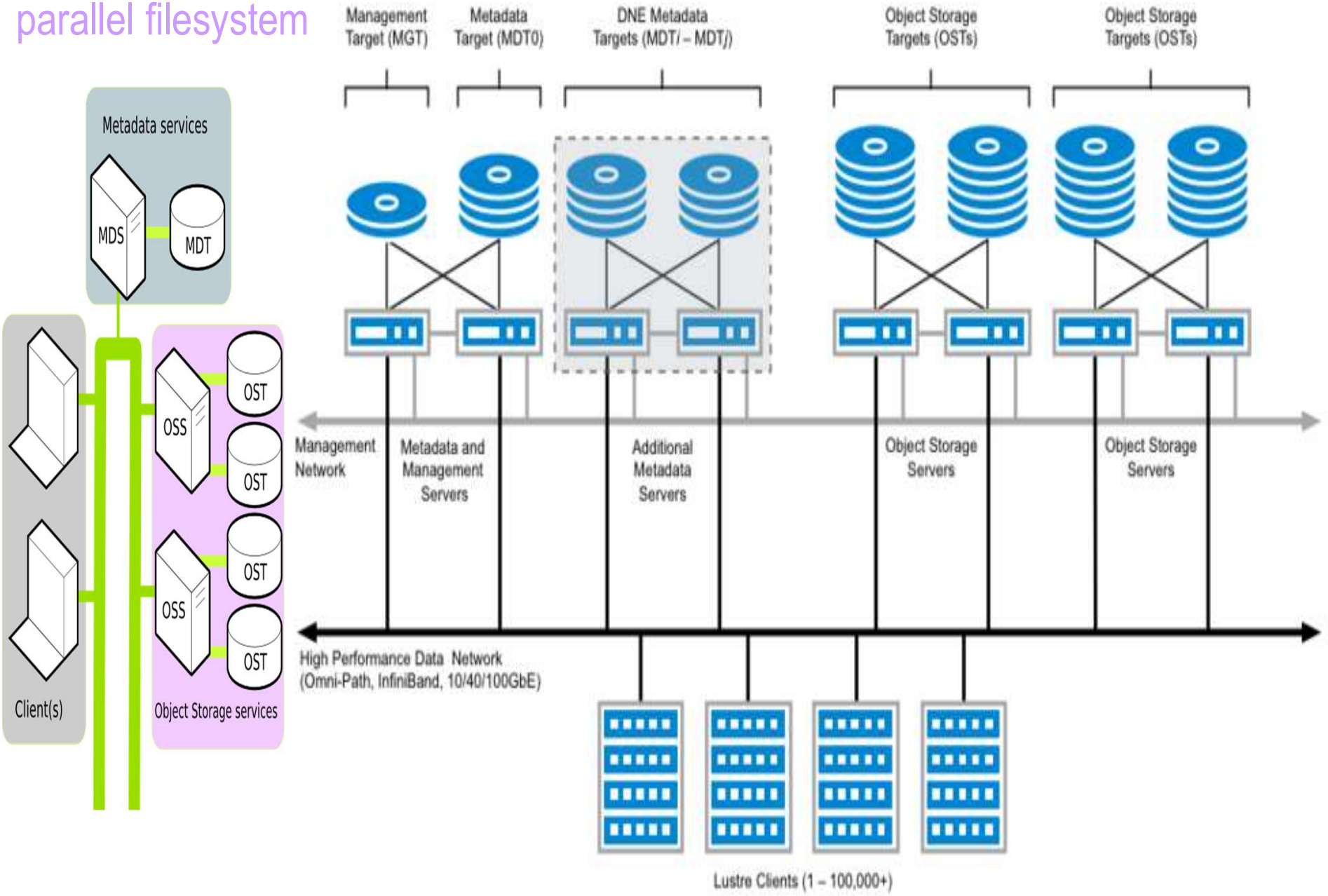
  -NFS (Network File Sytem)

  -Intel DAOS (Distributed Asynchronous Object Storage)

  -Lustre (SUN)

  -PVFS (IBM)

Lustre
parallel filesystem



Prominent Components of Cluster Computers (VI)

- Parallel Programming Environments and Tools
  - Threads (PCs, SMPs, NOW..)
    - POSIX Threads
    - OpenMP
    - Java Threads
  - MPI (Message Passing Interface)
    - Linux, Windows, on many Supercomputers
  - Parametric Programming
  - Software DSMs (Shmem)
  - Compilers
    - C/C++/Python
    - Parallel programming with C++ (MIT Press book)
  - RAD (rapid application development) tools
    - GUI based tools for PP modeling
  - Debuggers
  - Performance Analysis Tools
  - Visualization Tools

Linux Cluster Compilers			
Compiler		**Serial Command**	**Parallel Commands**
Intel	C	icc	mpicc
	C++	icpc	mpicxx, mpic++
	Fortran	ifort	mpif77, mpif90, mpifort
GNU	C	gcc	mpicc
	C++	g++	mpicxx, mpic++
	Fortran	gfortran	mpif77, mpif90, mpifort
PGI	C	pgcc	mpicc
	C++	pgc++	mpicxx, mpic++
	Fortran	pgf77, pgf90, pgfortran	mpif77, mpif90, mpifort
LLVM/Clang	C	clang	mpicc
	C++	clang++	mpicxx, mpic++

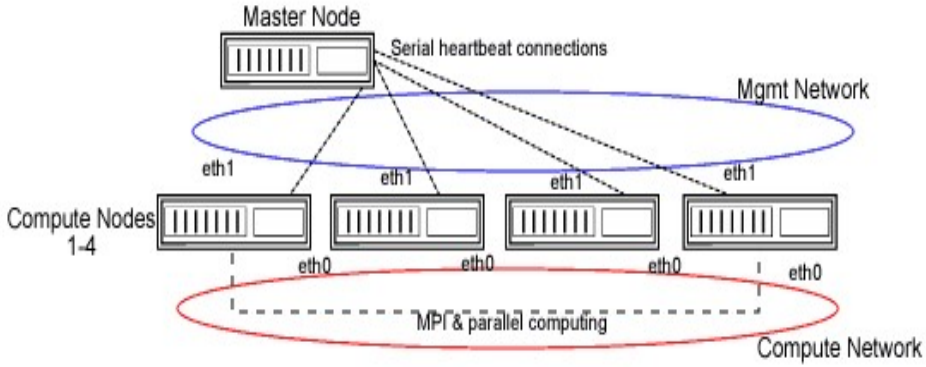
Prominent Components of Cluster Computers (VII)

- **Applications**
  - **Sequential**
  - **Parallel / Distributed (Cluster-aware app.)**
    - **Grand challenging applications**
      - **Weather Forecasting**
      - **Quantum Chemistry**
      - **Molecular Biology Modeling**
      - **Engineering Analysis (CAD/CAM)**
      - **.....**
    - **PDBs, web servers, data-mining**

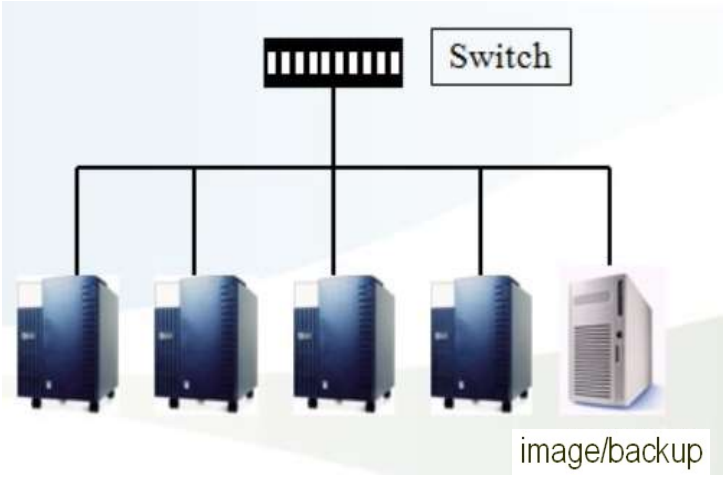
A Summary of HPC Cluster Components

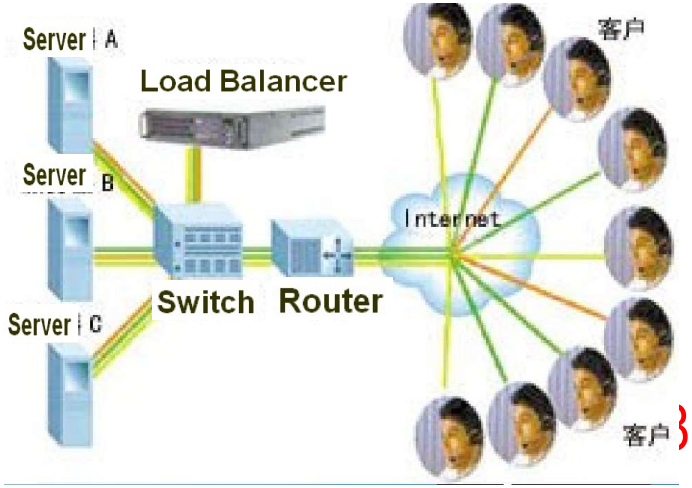
- **(6) Applications:**
  Scientific and engineering computing, CAD,CAE,CAM; BD & AI etc.
- **(5) Parallel Programming Environments and Tools:**
  -MPI/OpenMP, Math. Libraries and Compilers, Debuggers, etc.
  -Performance analysis and tuning tools: Vtune, Tau, Scalasca, perf, PAPI
- **(4) Cluster Middleware:**
  SSE, File System (e.g. NFS, Lustre, PVFS), SSH/rsh, NIS,
  Resource management and scheduling (RMS) software (e.g. LSF, PBS, Slurm, Condor)
- **(3) O.S. :**
  Linux, drivers, monitoring tool, etc.
- **(2) High Performance Networks/Switches:**
  Infiniband, Gigbit ethernet, etc.
- **(1) Hardware (nodes):**
  Multi-core/Many-core CPU, Mem, Disk/Disk Array, GPU

Clusters Classification (I)



- **Application Target**
  - **High Performance (HP) Clusters**
    - Grand Challenging Applications
  - **High Availability (HA) Clusters**
    - Mission Critical applications
  - **Load Balancer Cluster**
    - Web Server, Database, Email Server, VOD…





Clusters Classification (II)

- **Node Ownership**
  - **Dedicated Clusters**
  - **Non-dedicated clusters**
    - **Adaptive parallel computing**
    - **Communal multiprocessing**

Clusters Classification (III)

- Node Hardware
  - Clusters of PCs (CoPs)
    - Piles of PCs (PoPs)
  - Clusters of Workstations (COWs)
  - Clusters of SMPs (CLUMPs)
  - Clusters of (Blade) Servers

Clusters Classification (IV)

- Node Operating System
  - Linux Clusters (e.g., Beowulf)
  - Solaris Clusters (e.g., Berkeley NOW)
  - AIX Clusters (e.g., IBM SP2)
  - SCO/Compaq Clusters (Unixware)
  - Digital VMS Clusters
  - HP-UX clusters
  - Windows HPC clusters

Clusters Classification (V)

- **Node Configuration**
  - **Homogeneous Clusters**
    - **All nodes will have similar architectures and run the same OSs**
  - **Heterogeneous Clusters**
    - **All nodes will have different architectures and run different OSs**

Clusters Classification (VI)

- **Levels of Clustering**
  - Group Clusters (#nodes: 2-99)
    - Nodes are connected by SAN like Myrinet
  - Departmental Clusters (#nodes: 10s to 100s)
  - Organizational Clusters (#nodes: many 100s)
  - National Metacomputers (WAN/Internet-based)
  - International Metacomputers (Internet-based, #nodes: 1000s to many millions)
    - Grid Computing
    - Web-based Computing
    - Peer-to-Peer Computing

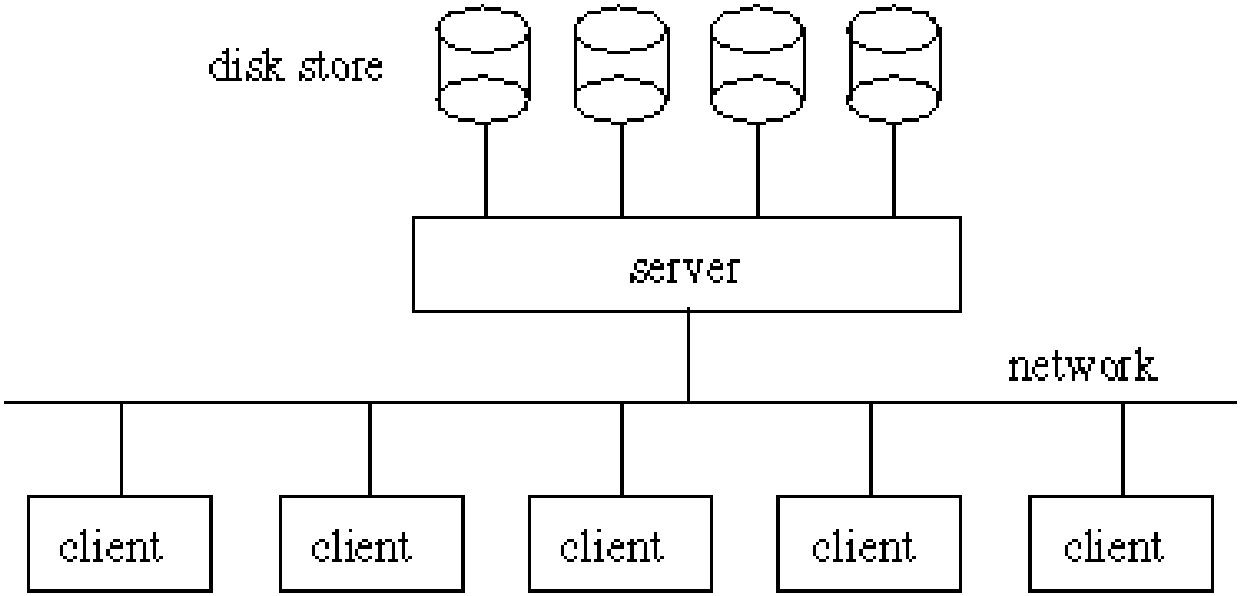
Berkeley Open Infrastructure for Network Computing

Linux Cluster

- Linux Cluster Introduction
- **NFS/NIS**
- Resource Management and Scheduling
- MPI/OpenMP

Network File System (NFS)

- NFS defines a method of sharing files in which files residing on one or more remote servers can be accessed on a local client system in a manner that makes them appear as local files and directories.



NFS versions

- NFS was originally developed by Sun Microsystems in 1984:

  NFSv2 released in 1985

  NFSv3 released in 1995

  NFSv4 released in Dec. 2003 for Sun and NetApp

  NFS v4.1 2010

  Linux in recently completed production stage (Univ. of Michigan)

- Detailed info about the NFS versions for Linux is available at http://nfs.sourceforge.net

- NFS2

  -its purpose is to achieve simple and fast server crash recovery

- NFS3

  - Support 64-bit file size and offset to handle files larger than 2 GB.
  - READDIRPLUS operation used to get the filehandle and attributes and file name when scanning the directory.
  - Support asynchronous writing on the server to improve writing performance.
  - Other file attributes in many responses to avoid the need to retrieve them.

- NFS4

  -a big advantage of NFSv4 is that only one UDP or TCP port 2049 is used to run the service, which simplifies the process of using protocols across firewalls

- NFS support should be enabled in Linux Kernel:

```
CONFIG_NFS_FS=m
CONFIG_NFS_V3=y
CONFIG_NFS_V4=y
CONFIG_NFS_DIRECTIO=y
CONFIG_NFSD=m
CONFIG_NFSD_V3=y
CONFIG_NFSD_V4=y
CONFIG_NFSD_TCP=y
```

- Verify NFS support at run time

```
lsmod | grep nfs
grep nfs /proc/filesystems
```

- Verify what NFS version is currently running

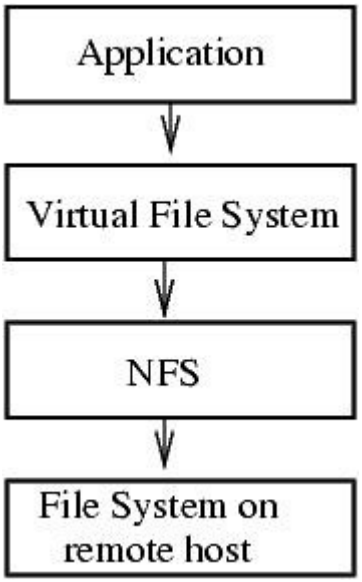
```
nfsstat
```

- Check the NFS version currently being used by the client

```
mount
```

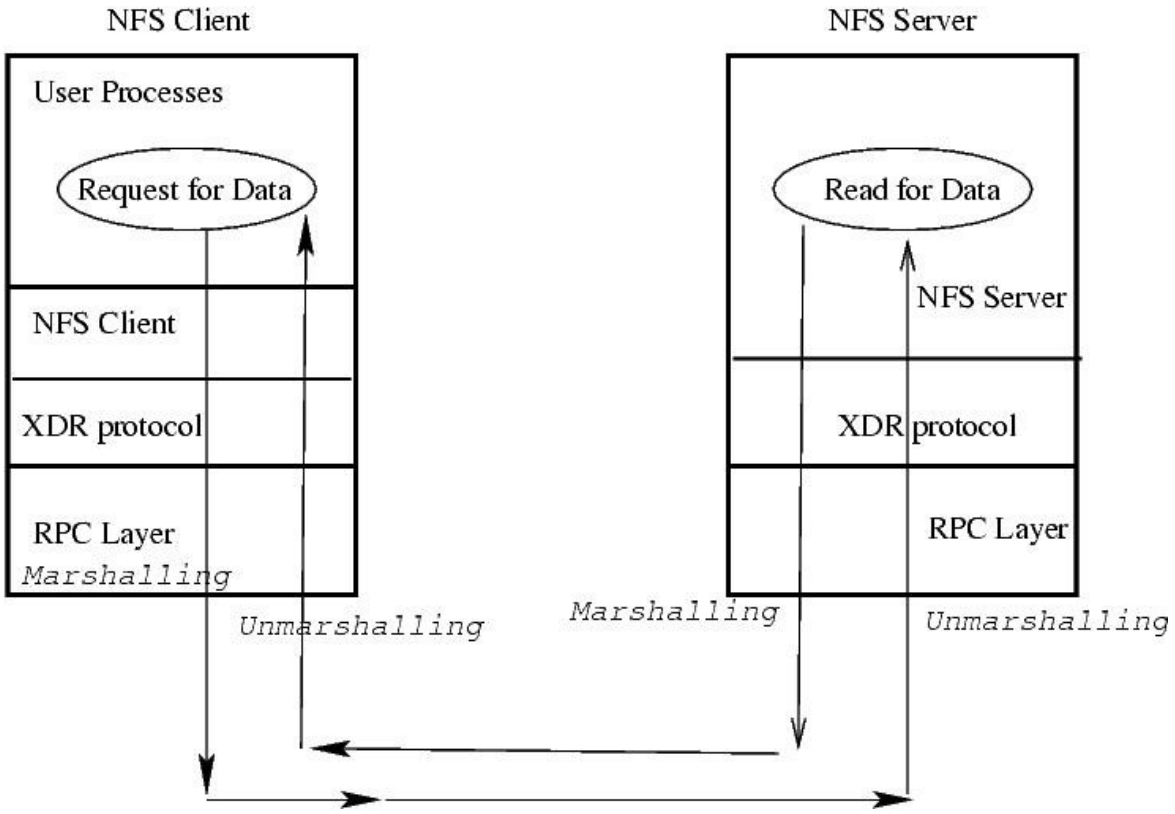
File System virtualization

- VFS is an abstraction of a local file system provided by Kernel for an application.
- Platform independent. Preserves or emulates Unix file system semantics.



NFS and RPC

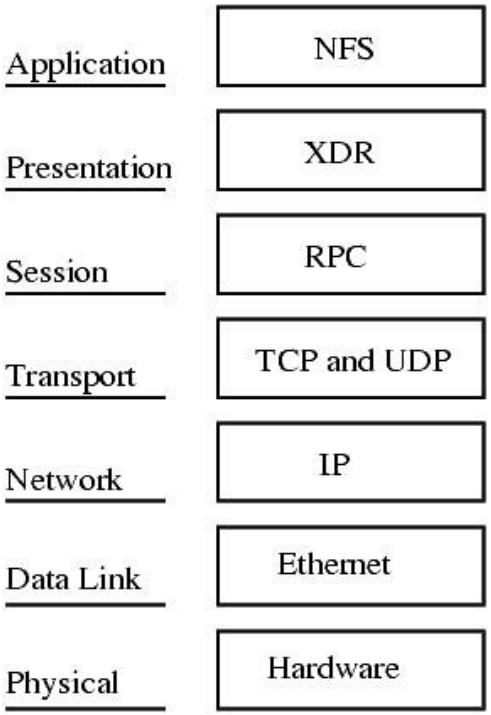
- NFS utilizes Remote Procedure Calls (RPC) layer for server -- client communications.



- Marshalling - Packaging arguments in XDR (eXternal Data Representation) format.
- XDR format is platform independent
- RPC allows applications on one host to call procedures (functions) on the other remote host
- RPC allows a server to respond to more than one version of a protocol at the same time (NFS 3 or NFS 2).

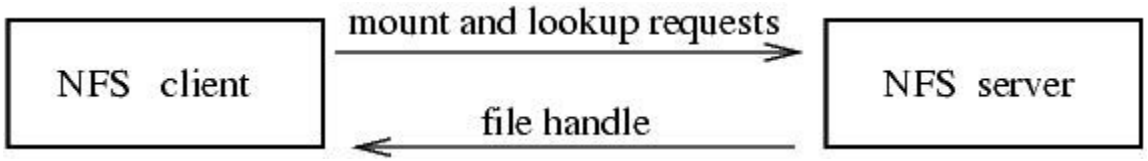
NFS in 7 layer OSI model

- NFS, XDR and RPC fit into the top 3 layes of the OSI model.
- XDR translates DATA into canonical (platform independent) format
- RPC provides remote procedure calls that appear as local processes.



NFSv2 and v3 server statelessness

- NFS server doesn't keep track with what clients access what files.
- NFS client keeps track with its RPC calls. If the server doesn‘t respond on a RPC call within defined time period, the client repeats the call.
- A NFS client receives *file handles* from NFS server when executes mount and lookup calls.



- The *file handles* on a client relate to the file pointers on an NFS server (inode number, disk device number, and inode generation number).
- If an NFS server crashes or reboots, NFS dependent applications on the client hang and then continue running after the server becomes available.
- If the file system on the server is changed or corrupted, the client gets a stale *file handle* error.

NFSv3 daemons

- NFS registers its processes (RPC services) with portmap.
- The portmap holds the port, service number and version numbers for each RPC service. If the portmap goes down, then all services need to be restarted after the portmap is restarted.

- **nfs** – The nfs service will start the server and the RPC processes necessary for accepting shared systems.
- **nfslock** – The nfslock service starts the RPC processes and allows NFC clients to lock files.
- **portmap** – You can take port reservations from local services with this one. portmap will respond to messages stating that certain ports are available for file access.

NFS server daemons

portmap	handles RPC requests and registers ports for RPC services
rpc.mountd	handles the initial mount requests
nfsd or [nfsd]	handles data streaming
rpc.rquotad	handles user file quotas on exported volumes
[lockd]	handles file locking to make sure several processes don't write into the same file
rpc.statd	lock monitor daemon works with [lockd]

Note, [nfsd] and [lockd] above mean they run by the Kernel threads and don't need to be started by a user.

NFS client daemons

```
portmap
[lockd]
rpc.statd
```

To verify that the services have started and registeresd with portmap, run command

```
rpcinfo -p
```

NFS Mount Request Illustrated

mount	--- RPC request --->	portmapper
mount	<-- port number ---	portmapper
mount	--- RPC request --->	rpc.mountd
mount	<-- respond --------	rpc.mountd

- I/O (Read/Write) access

write()	---- I/O ---------->	[nfsd]
write()	<-- ACK upon complete---	[nfsd]

An Example of Configuration for NFS

- **NFS server:**
  - **nfs-utils , NFS service**

/sbin/chkconfig nfs on
/sbin/chkconfig nfslock on
/etc/init.d/nfslock restart
/etc/init.d/nfs restart

  - **/etc/exports :**

/home *(rw,async,no_root_squash)
exportfs –a or reboot

- **NFS Client:**

mkdir /home
/etc/fstab:
server:/home /home nfs rw,bg,intr,soft,retrans=100 0 0

Network Information Service (NIS)

- NIS (originally called Yellow Pages or YP)
- a client–server directory service protocol for distributing system configuration data such as user and host names between computers on a computer network
- Sun Microsystems developed the NIS; the technology is licensed to virtually all other Unix vendors

```
Use "ethers"    for map "ethers.byname"
Use "aliases"   for map "mail.aliases"
Use "services"  for map "services.byname"
Use "protocols" for map "protocols.bynumber"
Use "hosts"     for map "hosts.byname"
Use "networks"  for map "networks.byaddr"
Use "group"     for map "group.byname"
Use "passwd"    for map "passwd.byname"
```

NIS Example

- NIS Server:

  /etc/sysconfig/network (Domain Name)

  ypserv,ypbind

  cd /var/yp && make

  /usr/lib/yp/ypinit -m

- NIS Client:

  /etc/yp.conf

  ypbind

- Both:

  /etc/nsswitch.conf (nisplus→nis files)

ssh/rsh

- **ssh**

  ssh-keygen –t rsa (–q –N “”)

  desk@cwxhost： ~> cd /home/desk/.ssh

  desk@cwxhost： ~/.ssh> cat id_rsa.pub >> authorized_keys

- **rsh**

  chkconfig --list rsh/rlogin/xinetd (on)

  /etc/init.d/xinetd restart

  Firewall off

  /etc/hosts.equiv or .rhosts

- **rsh for root**

  /etc/securetty (rsh,rlogin,rexec)

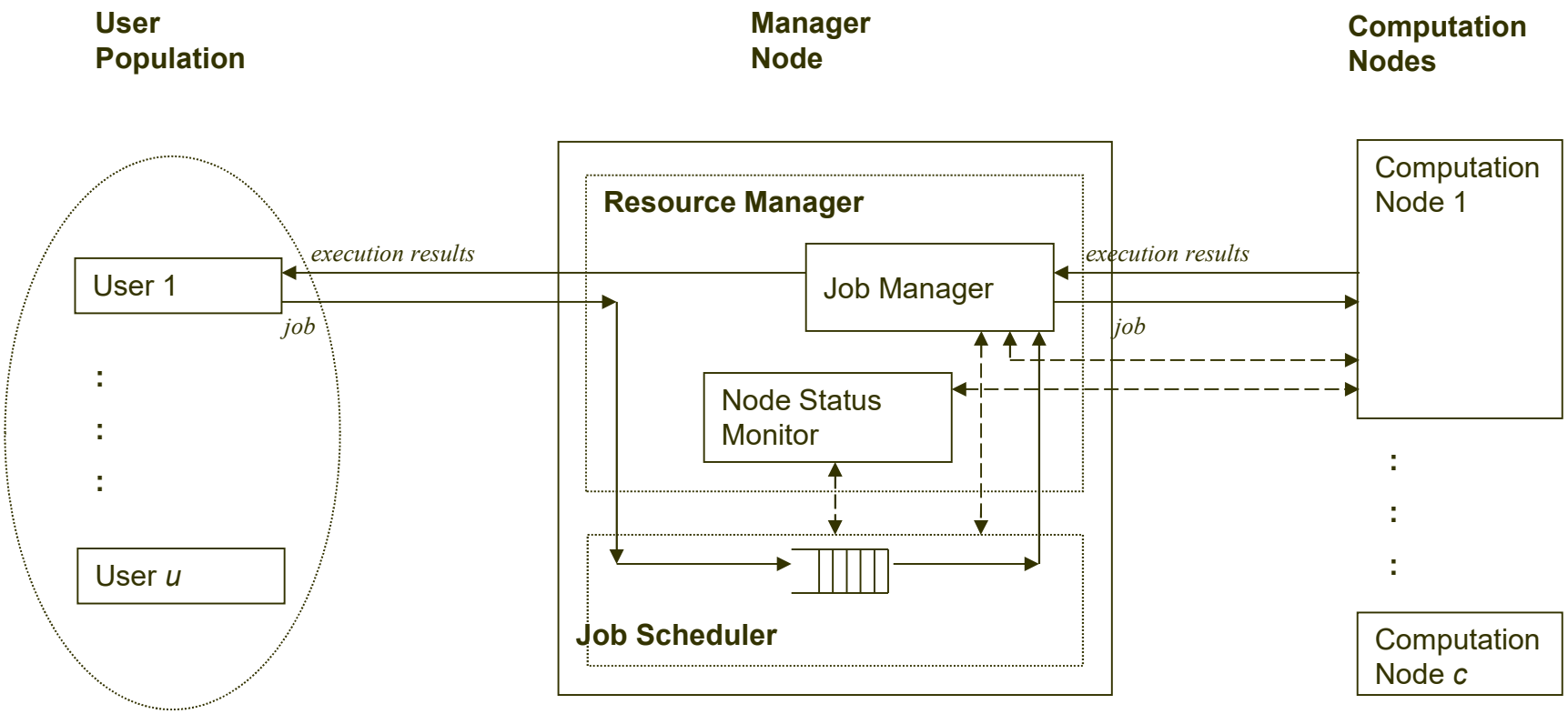
Linux Cluster

- Linux Cluster Introduction
- NFS/NIS
- **Resource Management and Scheduling**
- MPI/OpenMP

Resource Management and Scheduling (RMS)

- **RMS system is responsible for distributing applications among cluster nodes.**
- **It enables the effective and efficient utilization of the resources available**
- **Software components**
  - Resource manager
    - Locating and allocating computational resource, authentication, process creation and migration
  - Resource scheduler
    - Queuing applications, resource location and assignment. It instructs resource manager what to do when (policy)
- **Reasons for using RMS**
  - Provide an increased, and reliable, throughput of user applications on the systems
  - Load balancing
  - Utilizing spare CPU cycles
  - Providing fault tolerant systems
  - Manage access to powerful system, etc
- **Basic architecture of RMS: client-server system**

Cluster RMS Architecture



Services provided by RMS

- **Process Migration**
  - Computational resource has become too heavily loaded
  - Fault tolerant concern
- **Checkpointing**
- **Scavenging Idle Cycles**
  - 70% to 90% of the time most workstations are idle
- **Fault Tolerance**
- **Minimization of Impact on Users**
- **Load Balancing**
- **Multiple Application Queues**

Some Popular Resource Management Systems

Project	Commercial Systems - URL
LSF	http://www.platform.com/
SGE	http://www.sun.com/grid/
NQE	http://www.cray.com/
LL	http://www.ibm.com/systems/clusters/software/loadleveler/
PBS	http://www.pbsgridworks.com/

	Public Domain System - URL
Slurm	https://slurm.schedmd.com/
Condor	http://www.cs.wisc.edu/condor/
GNQS	http://www.gnqs.org/

Distributed Load Sharing System--LSF

- **Introduction**
  - LSF (Load Sharing Facility): is developed by Platform Computing Corporation ,Canada
  - Its CTO is the first doctor of computer science school of NWPU. Its products have formed a series of load sharing suite, occupying maximum market quotient, which is sold bundled with products of famous IT corporations.
  - Now LSF is the factual industry standard of load sharing.

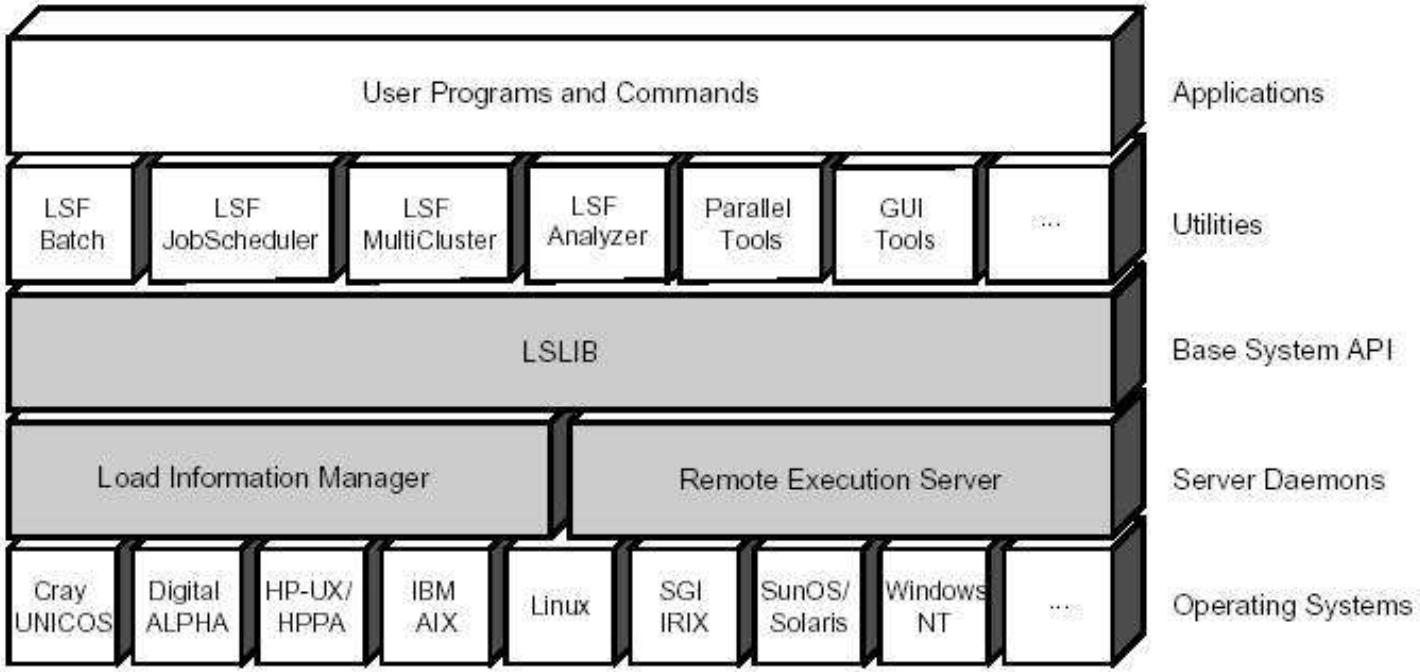
Performance and Feature

- **Goal:** provide reliable, extendable advance methods fully utilized network resource
- **Function:** provide various abilities for load management by scheduling, analyzing and supervising distributed application load
- **Mode:** provide multiple OS functions in network layer, manage all the distributed computing resource, make the whole network look like a uniform virtual host or a super computer
- **Framework:** six relatively independent components, support mainstream Operating Systems

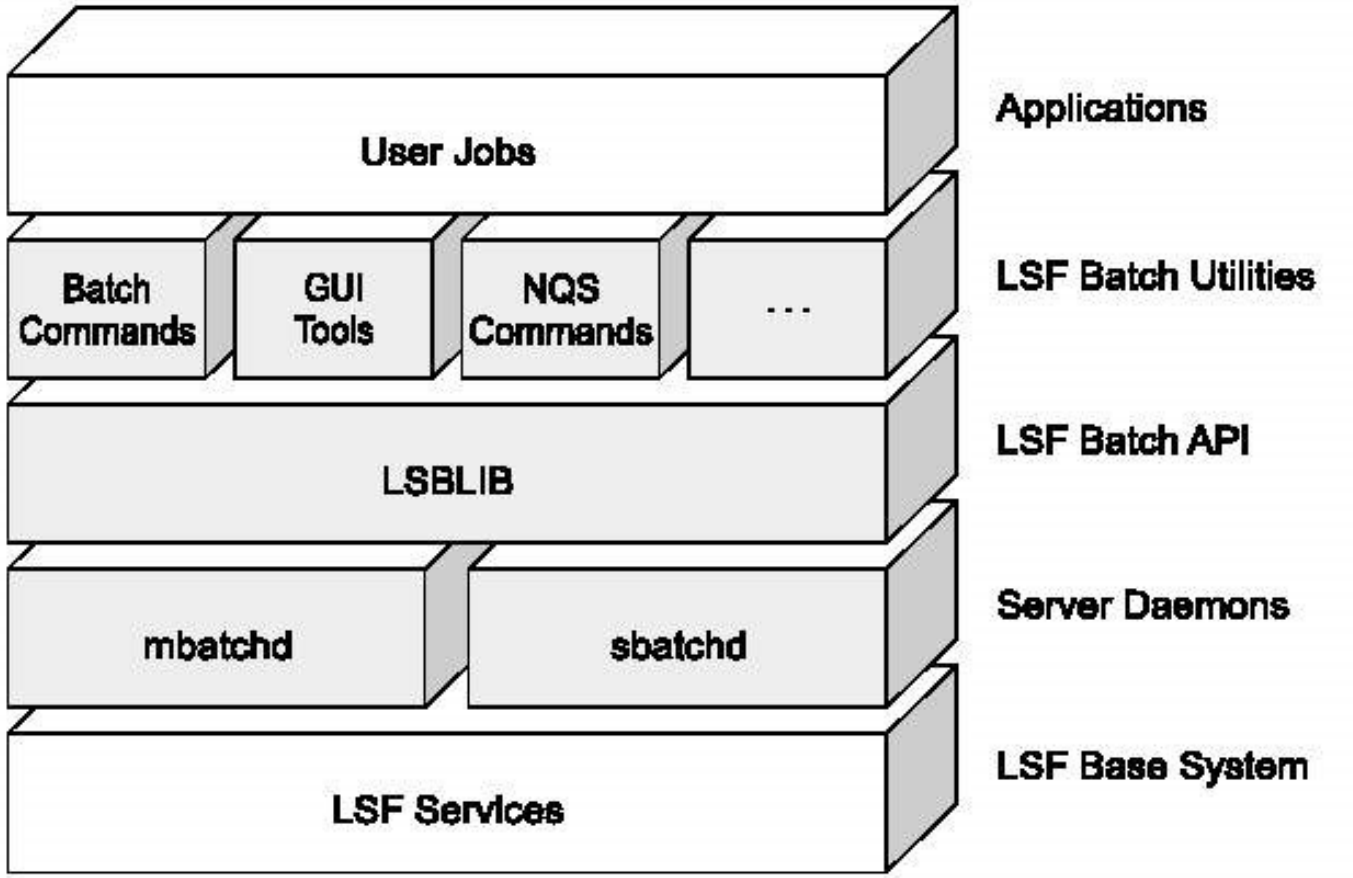
Structure of LSF Base

Structure of LSF Base

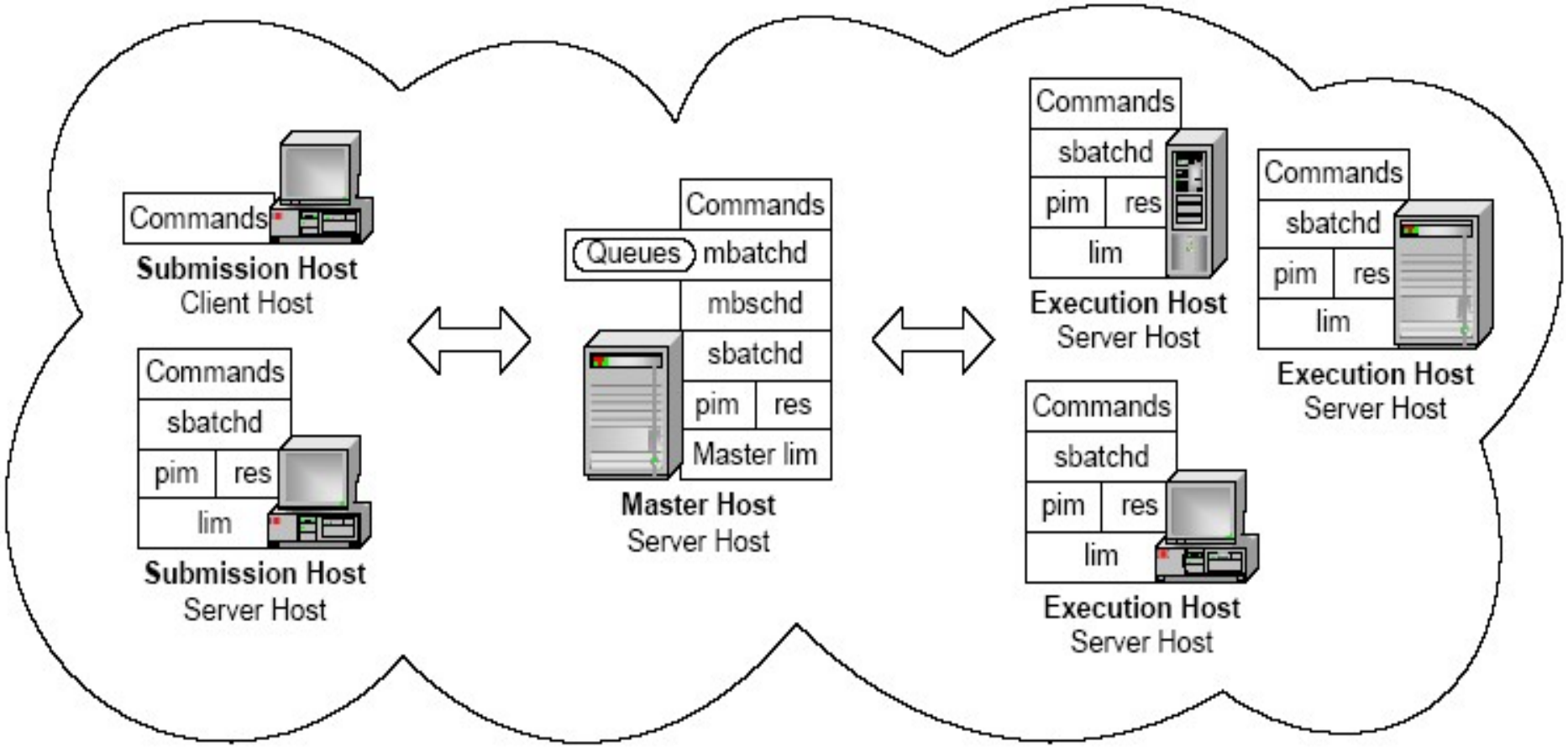
The software modules that make up LSF Base are shaded.



Structure of LSF Batch



LSF Cluster Concepts



Related commands

- Lsload, lshosts, bhosts, bqueues;
- Bsub, bjobs, bkill, bpeek, bhist

HOST_NAME	status	r15s	r1m	r15m	ut	pg	ls	it	tmp	swp	mem
c06b16	ok	0.0	0.0	0.0	0%	0.0	0	172	150G	62G	61G
su01	ok	0.0	0.6	0.4	1%	0.0	14	1	162G	62G	27G
mu02	ok	0.3	0.0	0.2	1%	0.0	0	8657	126G	31G	29G
su02	ok	0.3	1.9	2.2	3%	0.0	2	12	93G	62G	28G
c05b16	ok	3.4	4.1	4.1	24%	0.0	0	14176	116G	62G	18G
c09b06	ok	6.0	5.9	3.9	37%	0.0	0	39904	179G	62G	61G
c09b12	ok	6.6	6.4	7.1	40%	0.0	0	19024	119G	62G	61G
c03b03	ok	6.7	7.0	7.0	44%	0.0	0	28752	193G	62G	54G
c01b12	ok	7.0	7.0	7.0	44%	0.0	0	39936	192G	62G	60G
c02b05	ok	7.0	7.0	7.0	44%	0.0	0	3914	193G	62G	61G
c03b07	ok	7.0	7.1	7.0	44%	0.0	0	28752	184G	62G	60G
c04b12	ok	7.0	7.0	7.1	44%	0.0	0	3992	192G	62G	60G
c04b13	ok	7.1	7.1	6.5	45%	0.0	0	28752	193G	62G	60G
c02b04	ok	7.1	7.2	7.1	44%	0.0	0	39936	192G	62G	60G
c09b15	ok	7.4	7.5	6.6	46%	0.0	0	39904	187G	62G	60G
c05b01	ok	7.6	6.3	7.3	41%	0.0	0	28752	193G	62G	60G
c05b05	ok	7.7	8.0	7.9	49%	0.0	0	28752	179G	62G	18G
c03b06	ok	8.0	8.0	8.0	49%	0.0	0	28752	189G	62G	18G
c04b14	ok	8.0	7.9	7.7	49%	0.0	0	28752	193G	62G	60G
c06b12	ok	8.0	8.2	8.0	50%	0.0	0	39936	192G	62G	60G
c09b01	ok	8.0	8.2	7.8	50%	0.0	0	39904	187G	62G	60G
c04b02	ok	8.0	8.0	7.9	50%	0.0	0	28752	193G	62G	60G
c04b04	ok	8.0	8.0	8.0	50%	0.0	0	28720	186G	62G	51G
c06b06	ok	8.2	7.9	7.9	50%	0.0	0	39936	174G	62G	60G
c05b11	ok	8.2	8.0	8.0	50%	0.0	0	28752	193G	62G	48G
c09b04	ok	9.0	9.0	8.9	56%	0.0	0	39904	189G	62G	58G
c02b11	ok	10.8	12.2	12.5	76%	124.8	0	39936	193G	47G	16G
c03b16	ok	12.0	12.0	12.6	75%	0.0	0	28752	189G	62G	61G
c03b09	ok	12.0	12.0	12.0	75%	0.0	0	28752	165G	62G	49G
c06b01	ok	12.0	12.0	12.0	75%	0.0	0	39936	189G	62G	28G
c09b07	ok	12.0	12.1	12.1	76%	0.0	0	39872	189G	62G	48G
c09b09	ok	12.0	12.0	12.0	75%	0.0	0	39904	189G	62G	60G
c05b07	ok	12.1	11.8	12.1	75%	0.0	0	28752	186G	62G	60G
c04b06	ok	12.1	12.1	12.0	76%	0.0	0	28752	193G	62G	44G
c04b11	ok	13.0	13.0	12.7	81%	0.0	0	28752	166G	62G	33G
c01b10	ok	13.1	15.1	15.1	93%	786.0	0	39936	193G	42G	10G
c09b10	ok	13.3	14.7	15.5	91%	0.0	0	39904	186G	62G	60G
c05b03	ok	14.0	14.0	14.0	88%	0.0	0	28752	86G	62G	53G
c09b14	ok	15.0	15.0	15.0	94%	0.0	0	39904	187G	62G	25G
c05b02	ok	15.4	15.1	15.7	94%	0.0	0	28752	192G	62G	59G
c04b01	ok	15.7	16.0	15.9	100%	0.0	0	28752	192G	62G	52G
c05b12	ok	15.8	15.6	15.7	98%	0.0	0	28752	192G	62G	51G
c08b12	ok	16.0	16.0	16.0	100%	0.0	0	39936	189G	62G	53G
c01b03	ok	16.0	16.1	16.1	100%	0.0	0	812	192G	62G	54G
c01b05	ok	16.0	16.0	15.6	100%	0.0	0	39936	150G	62G	54G
c01b09	ok	16.0	16.2	16.3	100%	0.0	0	34240	189G	62G	48G
c01b15	ok	16.0	16.0	16.0	100%	0.0	0	39936	192G	62G	25G
c01b16	ok	16.0	16.0	15.9	100%	0.0	0	39936	192G	62G	50G

```
[houzx@console ms]$ cat run.sh
#cd `pwd`
DIR=`pwd`
echo $DIR
cd "$DIR"
/export/home/ms6.1/Accelrys/MaterialsStudio6.1/etc/Discover/bin/RunDiscover.sh -np 6 Sketch
```

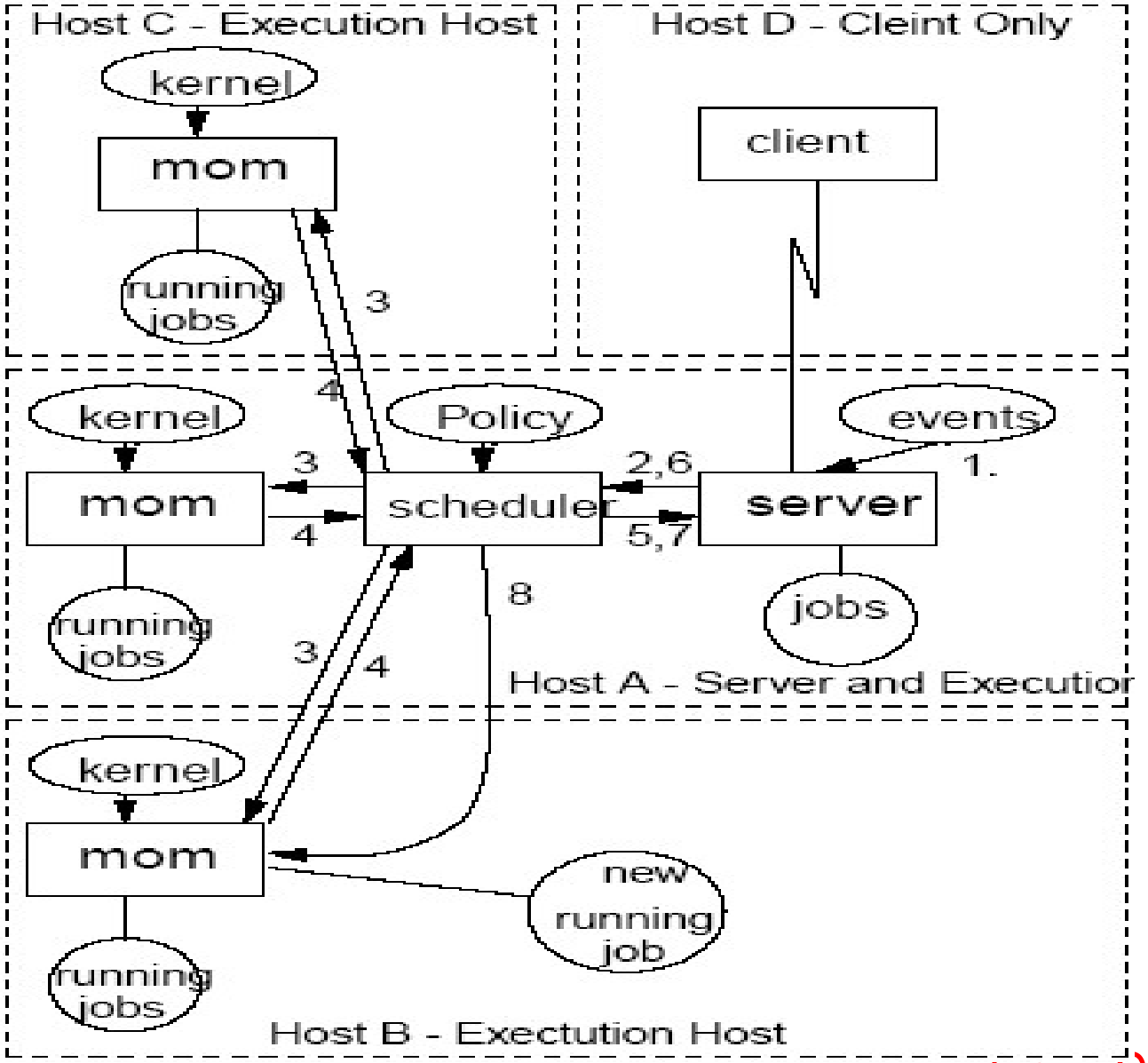
- **bsub ./run.sh**
- **bjobs**

```
[houzx@console ms]$ bjobs -u all
JOBID   USER    STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
374997  zengqf  RUN   normal     console     c0131       WSY3       Sep 17 20:59
                                             c0131
                                             c0131
                                             c0131
375004  zengqf  RUN   normal     console     c0131       WSY3       Sep 17 20:59
                                             c0131
                                             c0131
                                             c0131
374998  zengqf  RUN   normal     console     c0128       WSY3       Sep 17 20:59
                                             c0128
                                             c0128
                                             c0128
375006  zengqf  RUN   normal     console     c0128       WSY3       Sep 17 20:59
                                             c0128
                                             c0128
                                             c0128
374999  zengqf  RUN   normal     console     c0129       WSY3       Sep 17 20:59
                                             c0129
                                             c0129
                                             c0129
```

What is PBS?

- The Portable Batch System, PBS, is a batch job and computer system resource management package. It is open source software.
- It was developed with the intent to be conformant with the POSIX 1003.2d Batch Environment Standard.
- PBS may be installed and configured to support jobs run on a single system, or many systems grouped together. Because of the flexibility of PBS, the systems may be grouped in many fashions.
- New PBS—TORQUE ;Add new function.

OPEN PBS Batch Scheduling on Multiple Hosts



Components of PBS

- Commands
  - •command line commands that are POSIX 1003.2d conforming a graphical interface.
  - •These are used to submit, monitor, modify, and delete jobs.
  - •The commands can be installed independently
  - •There are three classifications of commands: user commands ;operator commands, and manager commands. Operator and manager commands require different access privileges.
- Job Server

The Job Server is the central of PBS, The Server's main function is to provide the basic batch services such as receiving/creating a batch job, modifying the job, protecting the job against system crashes, and running the job

Components of PBS

Job Executor

The job executor is the daemon which actually places the job into execution. This daemon, *pbs_mom* , is informally called *Mom* as it is the mother of all executing jobs.
Mom places a job into execution when it receives a copy of the job from a Server. Mom creates a new session as identical to a user login .

Job Scheduler

- The Job Scheduler is another daemon which contains the site's policy controlling which job is run and where and when it is run.
- PBS allows each site to create its own Scheduler.
- the Scheduler can communicate with the various Moms to learn about the state of system resources and with the Server to learn about the availability of jobs to execute.
- The interface to the Server is through the same API as the commands. In fact, the Scheduler just appears as a batch Manager to the Server.

Components of PBS

- **API**

In addition to the above major pieces, PBS also provides a Application Program Interface, API, which is used by the commands to communicate with the Server. This API is described in the section 3 man pages finished with PBS. A site may make use of the API to implement new commands if so desired

Related commands

- pbsnodes,

- Qsub,qstat,qdel,qstop,qrerun

- **qsub MPI.pbs**

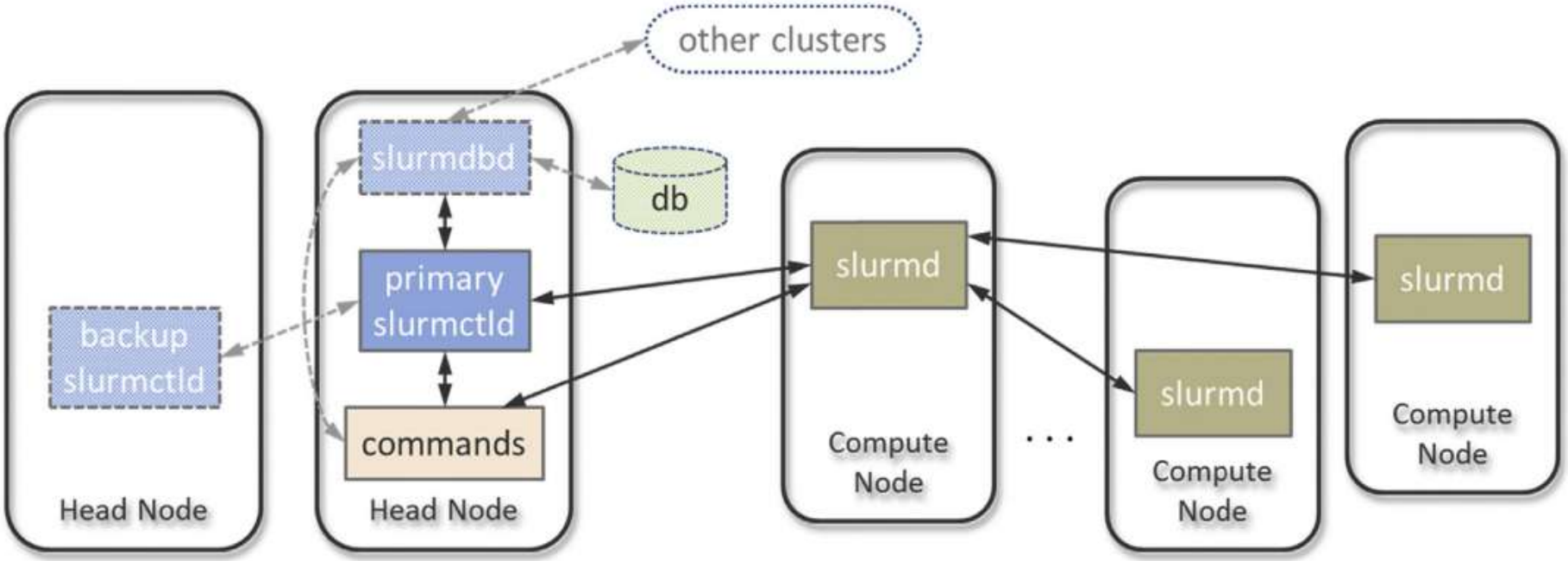
```
[houzx@su02 ~]$ cat /export/env/MPI.pbs
#!/bin/sh
#####################################################################
#PBS -N MPI
#PBS -l nodes=3:ppn=16
#PBS -l walltime=4:00:00
#PBS -q batch
#PBS -V
#PBS -S /bin/bash
#####################################################################

#
source /export/compiler/intel/impi/4.1.0.024/mpienvSet.sh

###To use mpiifort for compilation###

##############To be modified by users###########

ARGS="test-c"

################################################

NP=`wc -l < $PBS_NODEFILE`

cd ${PBS_O_WORKDIR}

EXEC=${PBS_O_WORKDIR}/$ARGS.exe

#$MPI_HOME/bin/mpirun -machinefile $PBS_NODEFILE -np $NP $EXEC

cat $PBS_NODEFILE > /tmp/nodefile.$$
sort $PBS_NODEFILE |uniq > ./mpd.hosts
NN=`wc -l < ./mpd.hosts`
sed -e 's\c\ibc\g' -i /tmp/nodefile.$$

mpdboot -n $NN -f mpd.hosts
mpdtrace
mpiexec -genv I_MPI_DEVICE ssm -machinefile /tmp/nodefile.$$ -np $NP $EXEC
mpdallexit

rm -rf /tmp/nodefile.$$ ./mpd.hosts
```

```
[houzx@su02 huyt]$ cat vasp5.3.pbs
#!/bin/sh
#####################################################################
#PBS -N vasp5.3
#PBS -l nodes=2:ppn=16
#PBS -l walltime=12:00:00
#PBS -q batch
#PBS -V
#PBS -S /bin/bash
#####################################################################

####For Intel MPI#####
source /export/env/bashrc
source /export/opensource/vasp5.3/vaspenvSet.sh
EXEC=/export/opensource/vasp5.3/vasp_intelmpi


####For Open MPI#####
#export LD_LIBRARY_PATH=/export/compiler/intel/mkl/lib/intel64:/export/opensource/openmpi_1.8.4/lib:$LD_LIBRARY_PATH
#EXE=/export/opensource/vasp5.3/vasp_openmpi
#MPI_HOME=/export/opensource/openmpi_1.8.4

NP=`wc -l < $PBS_NODEFILE`

cd ${PBS_O_WORKDIR}

## 512 MB
#export P4_GLOBMEMSIZE=536870912

cat $PBS_NODEFILE > /tmp/nodefile.$$
sort $PBS_NODEFILE |uniq > ./mpd.hosts
NN=`wc -l < ./mpd.hosts`
sed -e 's\c\ibc\g' -i /tmp/nodefile.$$

mpdboot -n $NN -f mpd.hosts
mpdtrace
mpiexec -genv I_MPI_DEVICE ssm -machinefile /tmp/nodefile.$$ -np $NP $EXEC > out
mpdallexit

rm -rf /tmp/nodefile.$$ ./mpd.hosts
```

```
[houzx@su02 huyt]$ qsub vasp5.3.pbs
6489263.su02
[houzx@su02 huyt]$ qstat -an

su02:
                                                                  Req'd  Req'd   Elap
Job ID               Username    Queue    Jobname          SessID NDS   TSK    Memory Time  S Time
-------------------- ----------- -------- ---------------- ------ ----- ------ ------ ----- - -----
6489263.su02         houzx       batch    vasp5.3            --      2     32    --  12:00 Q   --
    --
```

```
[houzx@su02 ~]$ qstat -q

server: su02

Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
gpu                --       --       --    --    0   0 --   E R
batch              --       --       --    --  146  32 --   E R
grid               --       --       --    --    0   0 --   E R
exclusive          --       --       --    --    0   0 --   E R
mic                --       --       --    --    0   0 --   E R
                                               ----- -----
                                                 146    32
```

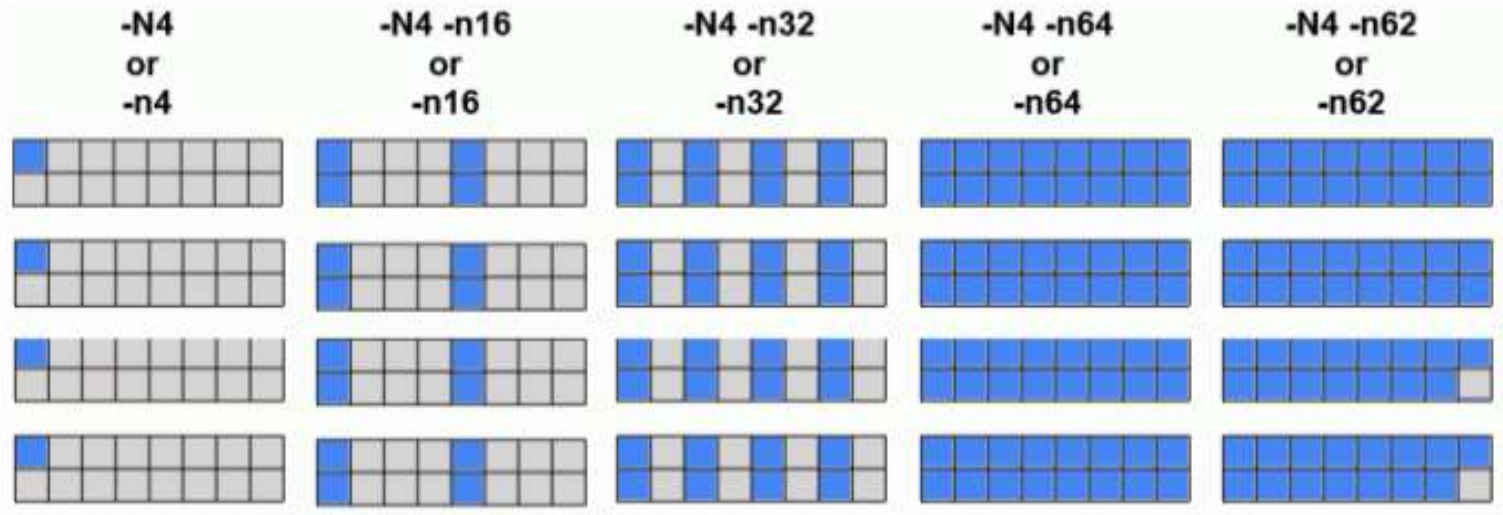
Simple Linux Utility for Resource Management (SLURM)



- **Simplifified architecture of SLURM. Components framed by dashed lines are optional.**

Srun

- srun, salloc, sbatch, squeue, scancel, sacct, sinfo
- Behavior of **`srun` -N** and **-n** flags - using 4 nodes in batch (#MSUB -l nodes=4), each of which has 16 cores



srun diagram

- Process/Thread Binding to Cores

```
% srun -l -n2 numactl -s | grep physc | sort
0: physcpubind: 0 1 2 3 4 5 6 7
1: physcpubind: 8 9 10 11 12 13 14 15

% srun -l -n4 numactl -s | grep physc | sort
0: physcpubind: 0 1 2 3
1: physcpubind: 4 5 6 7
2: physcpubind: 8 9 10 11
3: physcpubind: 12 13 14 15

% srun -l -n8 numactl -s | grep physc |sort
0: physcpubind: 0 1
1: physcpubind: 2 3
2: physcpubind: 4 5
3: physcpubind: 6 7
4: physcpubind: 8 9
5: physcpubind: 10 11
6: physcpubind: 12 13
7: physcpubind: 14 15
```

Slurm

```
% sbatch myjobscript

Submitted batch job 645133
```

```
#!/bin/tcsh
##### These lines are for Slurm
#SBATCH -N 16
#SBATCH -J parSolve34
#SBATCH -t 2:00:00
#SBATCH -p pbatch
#SBATCH --mail-type=ALL
#SBATCH -A myAccount
#SBATCH -o /p/lustre1/joeuser/par_solve/myjob.out

##### These are shell commands
date
cd /p/lustre1/joeuser/par_solve
##### Launch parallel job using srun
srun -n128 a.out
echo 'Done'
```

Linux Cluster

- Linux Cluster Introduction
- NFS/NIS
- Resource Management and Scheduling
- **MPI/OpenMP (Next)**

Homework_1

- **Experiments for a Linux cluster**

  Linux/VM;

  user account;

  /etc/hosts;

  SSH, rsh, NFS, NIS;

  gcc / intel compiler;

  MPI (mpich, openmpi, or MVAPICH)

  (Due: Dec. 30th, 2022)

Thank you!