TACC Linux User Environment
LSF/SGE Batch Schedulers

Karl W. Schulz

Texas Advanced Computing Center
The University of Texas at Austin

UT/Portugal Summer Institute Training
Coimbra, Portugal
July 14, 2008

TACC

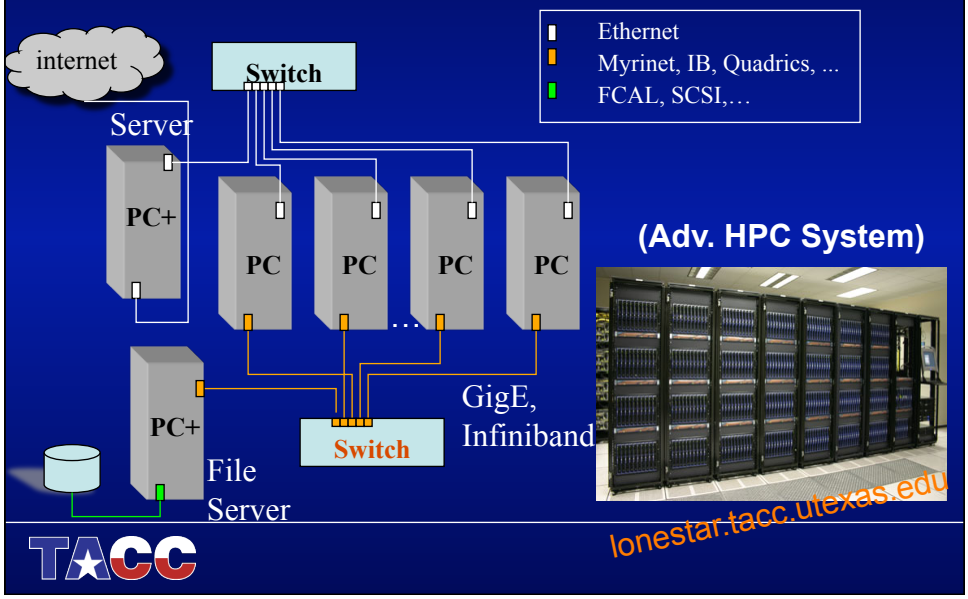
THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Outline

- Linux Clusters, HPC Software
- Initial Login
- Startup Scripts & Modules
- User Environment
- LSF Batch System
- SGE Batch System
- Intel Compilers
- Communication Switches and Libraries
- Libraries / Performance Tools
- System & Memory Information
- Moving Files Between Systems
- Little and Big Endians

TACC

Generic Cluster Architecture



HPC Software Components

- Major HPC Software Components
  - Batch System (production)
  - Interactive Utility (development)
  - High Speed Interconnects
  - Communication Library
  - Compilers
  - Advanced Math Libraries
  - Large Parallel File Systems
  - Monitoring Utilities
  - Multi-user Environment

TACC

Initial Login (Ranger)

- Login with SSH

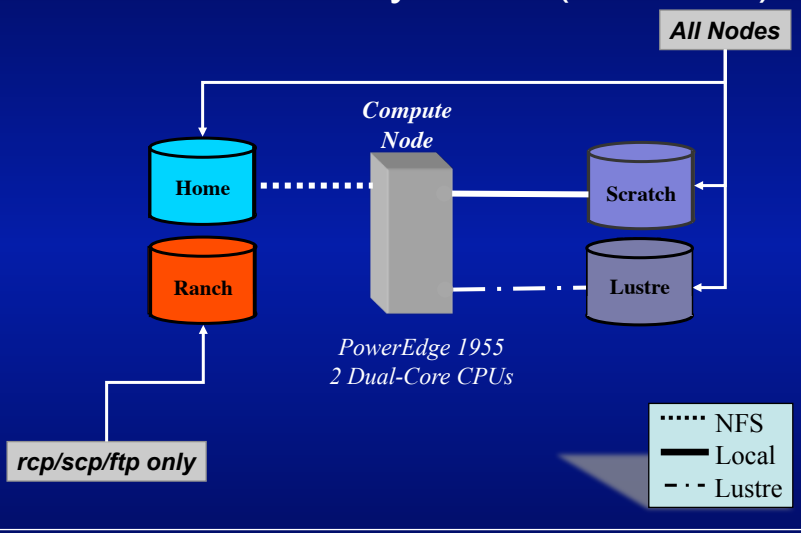
```
ssh ranger.tacc.utexas.edu
```

- Connects you to login3.ranger.tacc.utexas.edu or login4.ranger.tacc.utexas.edu

- Please don't overwrite ~/.ssh/authorized_keys
  - Feel free to add to it if you know what it's for
  - SSH used for job start up on the compute nodes, mistakes can prevent your jobs from running

TACC

Available File Systems (Lonestar)



TACC

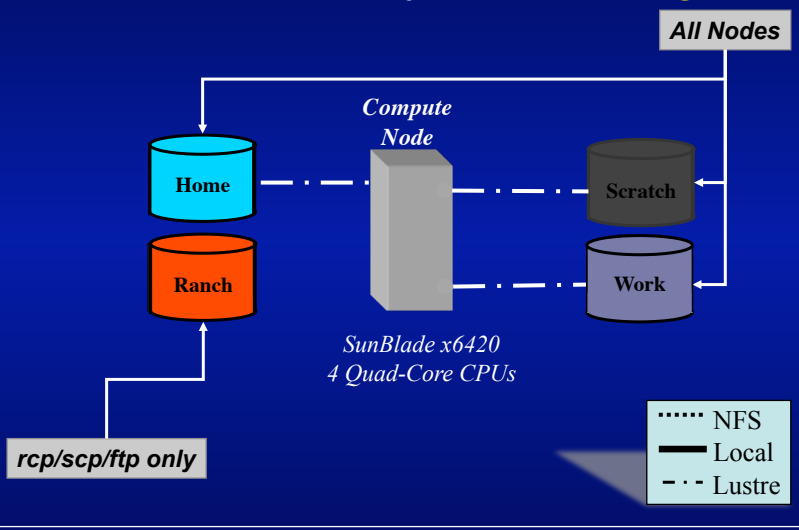
File System Access & Lifetime Table (lonestar)

Environment Variables	User Access Limit	Life Time
$HOME	200 MB quota	Project
$WORK	100TB/ no quota	10 Days
$ARCHIVE	Unlimited	Project
$SCRATCH	~56GB	Job Duration
SAN	Allocated	Project

(Use the aliases cd, and cdw to change directory to $HOME and $WORK respectively.)

TACC

Available File Systems (Ranger)



TACC

File System Access & Lifetime Table (Ranger)

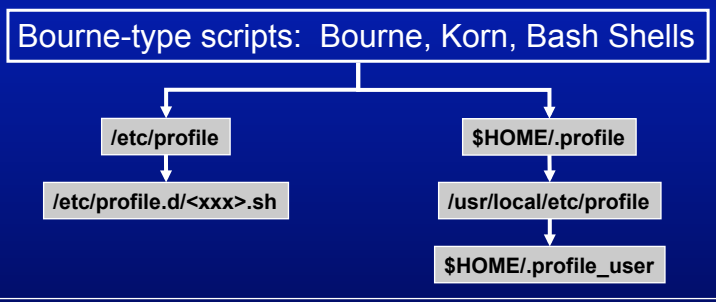
Environment Variables	User Access Limit	Life Time
$HOME	~2.5GB quota	Project
$WORK	~40GB quota	Project
$ARCHIVE	Unlimited	Project
$SCRATCH	~400TB	10 Days
SAN	Allocated	Project

(Use the aliases cd, cdw, and cds to change directory to $HOME, $WORK and $SCRATCH respectively.)

TACC

Startup Scripts & Modules

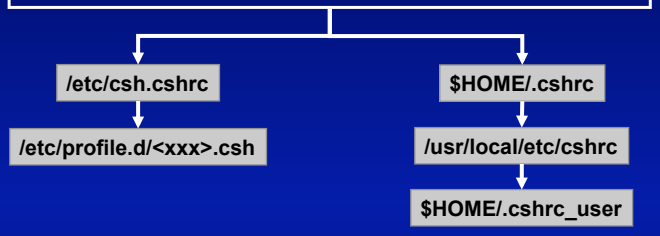
- Login shell is set with "chsh"
  - Takes some time to propagate (~1 hour)
- Each shell "sources" a set of scripts:



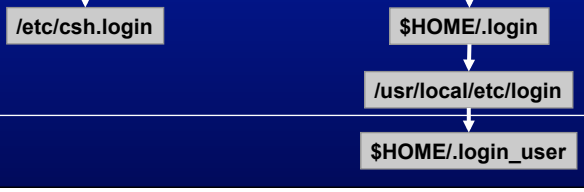
TACC

Startup Scripts & Modules

csh scripts for C-shell type shells: csh, tcsh



login scripts for C-shell type shells: csh, tcsh



TACC

Modules

- Modules are used to setup and remove various environment variables along with PATH, LD_LIBRARY_PATH declarations
- They are used to setup environments for packages & compilers.

```
lslogin1% module                                  {lists options}
lslogin1% module list                             {lists loaded modules}
lslogin1% module avail                            {lists available modules}
lslogin1% module del <module>                     {removes a module}
lslogin1% module add <module>                     {add a module}
lslogin1% module switch <mod1> <mod2>             {switch modules}
```

Currently available modules on Lonestar: `lslogin1$ module avail`

```
----------------------- /opt/intel9/modulefiles ------------------
mvapich-gen2/0.9.8   petsc/2.3.1-cxx      petsc/2.3.1-debug
petsc/2.3.1          petsc/2.3.1-cxxdebug

----------------------- /opt/modulefiles -------------------------
Linux                gromacs/3.3.1        netcdf/3.6.1
TACC                 hdf4/2r1             papi/3.2.1
amber/8              hdf5/1.6.5           plapack/3.0
amgr/2.0             intel/9.1            pmetis/3.1
binutils/2.17        java/1.4.2           scalapack/1.7
cluster              kojak/2.1.1          sprng/2.0
ddt/1.10             launcher/1.1         tacc-binutils/2.17
fftw/2.1.5           metis/4.0            tau/2.15.3
fftw/3.1.1           mkl/8.1              gamess/02_2006
gotoblas/1.02
```

TACC

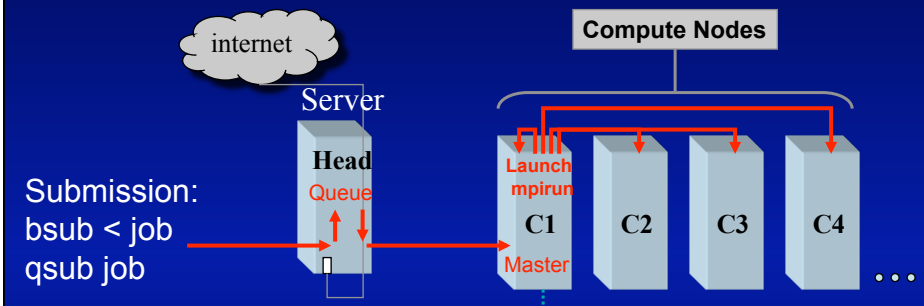
Modules

- Modules often define environment variables for convenient access to binaries, libraries, include files, and documentation
- See individual module's help for more information (and suggestions on linking against 3rd party libraries). For example: `lslogin1$ module help mkl`

```
----------- Module Specific Help for 'mkl/8.1' --------------------

The MKL module file defines the following environment variables:
TACC_MKL_DIR, TACC_MKL_DOC, TACC_MKL_LIB, and TACC_MKL_INC for
the location of the Intel MKL distribution, documentation,
libraries, and include files, respectively.

To use the MKL library, compile the source code with the option:
          -I$TACC_MKL_INC
and add the following options to the link step:
          -Wl,-rpath,$TACC_MKL_LIB -L$TACC_MKL_LIB -lmkl_em64t
The -Wl,-rpath,$TACC_MKL_LIB option is not required, however,
if it is used, then this module will not have to be loaded
to run the program during future login sessions.

Here is an example command to compile mkl_test.c:
mpicc -Wl,-rpath,$TACC_MKL_LIB -I$TACC_MKL_INC mkl_test.c
       -L$TACC_MKL_LIB -lmkl_em64t
```

TACC

Batch Submission Process



Queue: Job Script waits for resources on Server
Master: Compute Node that executes the job script, launches ALL MPI processes

Launch: ssh to each compute node to start executable (e.g. a.out)

ibrun ./a.out

mpirun –np # ./a.out

TACC

Batch Systems

- *Lonestar* uses Platform LSF for both the batch queuing system and scheduling mechanism (provides similar functionality to PBS)
  - LSF includes *global fairshare*, a mechanism for ensuring no one user monopolizes the computing resources
- *Ranger* uses Sun GridEngine (SGE) for both the batch queuing system and scheduling mechanism
- Batch jobs are submitted on the front end and are subsequently executed on compute nodes as resources become available
- Order of job execution depends on a variety of parameters:
  - Submission Time
  - Queue Priority: some queues have higher priorities than others
  - Backfill Opportunities: small jobs may be back-filled while waiting for bigger jobs to complete
  - Fairshare Priority: users who have recently used a lot of compute resources will have a lower priority than those who are submitting new jobs
  - Advanced Reservations: jobs my be blocked in order to accommodate advanced reservations (for example, during maintenance windows)
  - Number of Actively Scheduled Jobs: there are limits on the maximum number of concurrent processors used by each user

TACC

Lonestar Queue Definitions

Queue Name	Max Runtime	Min/Max Procs	SU Charge Rate	Use
normal	48 hours	2/512	1.0	Normal usage
high	48hours	2/512	1.8	Higher priority usage
development	30 min	1/16	1.0	Debugging and development Allows *interactive* jobs
hero	24 hours	>512	1.0	Large job submission Requires special permission
serial	12 hours	1/1	1.0	For serial jobs. No more than 4 jobs/ user
request				Special Requests
spruce				Debugging & development, special priority, urgent comp. env.
systest				System Use (*TACC Staff only*)

TACC

Ranger Queue Definitions

Queue	Max Runtime	Min/Max Procs	SU Charge Rate	Purpose
normal	24 hours	16/4096	1.0	Normal usage
large	24 hours	16/12288	1.0	Large job submission
development	2 hours	2/256	1.0	Debugging and development
serial	2 hours	1/1	1.0	Uniprocessor jobs
request		> 12K		Big, big jobs
systest				TACC system queue

TACC

Fairshare

- A global fairshare mechanism is implemented on Lonestar/ Ranger to provide fair access to its substantial compute resources
- Fairshare computes a dynamic priority for each user and uses this priority in making scheduling decisions
- Dynamic priority is based on the following criteria
  - Number of shares assigned
  - Resources used by jobs belonging to the user:
    - Number of job slots reserved
    - Run time of running jobs
    - Cumulative actual CPU time (not normalized), adjusted so that recently used CPU time is weighted more heavily than CPU time used in the distant past

TACC

LSF Fairshare

- **bhpart**: Command to see current fairshare priority. For example:

```
lslogin1--> bhpart -r
HOST_PARTITION_NAME:  GlobalPartition
HOSTS:  all

SHARE_INFO_FOR: GlobalPartition/
USER/GROUP   SHARES  PRIORITY  STARTED  RESERVED  CPU_TIME  RUN_TIME
avijit          1      0.333        0         0       0.0         0
chona           1      0.333        0         0       0.0         0
ewalker         1      0.333        0         0       0.0         0
minyard         1      0.333        0         0       0.0         0
phaa406         1      0.333        0         0       0.0         0
bbarth          1      0.333        0         0       0.0         0
milfeld         1      0.333        0         0       2.9         0
karl            1      0.077        0         0   51203.4         0
vmcalo          1      0.000      320         0  2816754.8   7194752
```

Priority

Commonly Used LSF Commands

bhosts	Displays configured compute nodes and their static and dynamic resources (including job slot limits)
lsload	Displays dynamic load information for compute nodes (avg CPU usage, memory usage, available /tmp space)
bsub	submits a batch job to LSF
bqueues	displays information about available queues
bjobs	displays information about running and queued jobs
bhist	displays historical information about jobs
bstop	suspends unfinished jobs
bresume	resumes one or more suspended jobs
bkill	Sends signal to kill, suspend, or resume unfinished jobs
bhpart	Displays global fairshare priority
lshosts	Displays hosts and their static resource configuration
lsuser	Shows user job information

Note: most of these commands support a "-l" argument for long listings. Consult the man pages for each of these commands for more information.

LSF Batch System

- LSF Defined Environment Variables:

LSB_ERRORFILE	name of the error file
LSB_JOBID	batch job id
LS_JOBPID	process id of the job
LSB_HOSTS	list of hosts assigned to the job. Multi-cpu hosts will appear more than once (may get truncated)
LSB_QUEUE	batch queue to which job was submitted
LSB_JOBNAME	name user assigned to the job
LS_SUBCWD	directory of submission, i.e. this variable is set equal to $cwd when the job is submitted
LSB_INTERACTIVE	set to 'y' when the –I option is used with bsub

TACC

SGE Batch System: Env. Variables

Variable	Purpose
JOB_ID	Batch job id
JOB_NAME	User-assigned (-J) name of the job
NSLOTS	Number of slots/processes for a parallel job
QUEUE	Name of the queue the job is running in
PE	Parallel environment used by the job
SGE_STDOUT_PATH SGE_STDERR_PATH	Location of the file where standard output/error is being written

TACC

LSF/SGE Batch Systems

- Comparison of LSF, SGE and Loadleveler commands that provide similar functionality

LSF	SGE	Loadleveler
bresume	qhold -r	llhold -r
bsub	qsub	llsubmit
bqueues	qstat	llclass
bjobs	qstat	llq
bstop	qhold	llhold
bkill	qdel	llcancel

TACC

Batch System Concepts

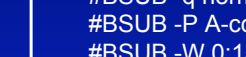
- Submission (need to know)
  - Required Resources
  - Run-time Environment
  - Directory of Submission
  - Directory of Execution
  - Files for stdout/stderr Return
  - Email Notification
- Job Monitoring
- Job Deletion
  - Queued Jobs
  - Running Jobs

TACC

LSF: Basic MPI Job Script

```
#!/bin/csh
#BSUB -n 32                  → Total number of processes
#BSUB -J hello               → Job name
#BSUB -o %J.out              → Stdout Output file name (%J = jobID)
#BSUB -e %J.err              → Stderr Output file name
#BSUB -q normal              → Submission queue
#BSUB -P A-ccsc              → Your Project Name
#BSUB -W 0:15                → Max Run Time (15 minutes)

echo "Master Host = "`hostname`        } Echo pertinent
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"      } environment info
echo "PWD_DIR: "`pwd`

ibrun ./hello                } Execution command
```

Parallel application manager and mpirun wrapper script — *executable*



LSF: Extended MPI Job Script

```
#!/bin/csh
#BSUB -n 32                  → Total number of processes
#BSUB -J hello               → Job name
#BSUB -o %J.out              → Stdout Output file name (%J = jobID)
#BSUB -e %J.err              → Stderr Output file name
#BSUB -q normal              → Submission queue
#BSUB -P A-ccsc              → Your Project Name
#BSUB -W 0:15                → Max Run Time (15 minutes)
#BSUB -w 'ended(1123)'       → Dependency on Job <1123>
#BSUB -u karl@tacc.utexas.edu → Email address
#BSUB -B                     → Email when job begins execution
#BSUB -N                     → Email job report information upon completion

echo "Master Host = "`hostname`
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"

ibrun ./hello
```

SGE: Basic MPI Job Script

```
#$ -S /bin/csh
#$ -pe 16way 32          } Total number of processes
#$ -N hello              } Job name
#$ -o ${JOB_ID}J.out     } Stdout Output file name
#$ -e ${JOB_ID}J.err     } Stderr Output file name
#$ -q normal             } Submission queue
#$ -A A-ccsc             } Your Project Name
#$ -l h_rt=00:15:00      } Max Run Time (15 minutes)

echo "Master Host = "`hostname`
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"
echo "PWD_DIR: "`pwd`          } Echo pertinent environment info

ibrun ./hello                 } Execution command
```

Parallel application manager and mpirun wrapper script *executable*

Job Sizing with SGE

- You must always put a multiple of 16 next to the name of the parallel environment

```
#$ -pe 16way 64       {64 tasks, 4 nodes}
#$ -pe  8way 64       {32 tasks, 4 nodes}
```

- SGE doesn't automatically handle the case where the number of tasks you want is not a multiple of 16
- If you want a non-multiple of 16, you may set

```
#$ -pe 16way 32
...
export MY_NSLOTS=23
...
ibrun ./mycode
```

SGE: Extended MPI Job Script

```
#$ -S /bin/csh
#$ -pe 16way 32
#$ -N hello
#$ -o ${JOB_ID}.out
#$ -e ${JOB_ID}.err
#$ -q normal
#$ -A A-ccsc
#$ -l h_rt=00:15:00
#$ -hold_jid 1123
#$ -M name@domain
#$ -m b,e

echo "Master Host = "`hostname`
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"

ibrun ./hello
```

Total number of processes
Job name
Stdout Output file name (%J = jobID)
Stderr Output file name
Submission queue
Your Project Name
Max Run Time (15 minutes)
Dependency on Job <1123>
Email address
Email when job begins execution and email job report information upon completion

LSF: Job Script Submission

- When submitting jobs to LSF using a job script, a redirection is required for bsub to read the commands. Consider the following script:

```
lslogin1> cat job.script
#!/bin/csh
#BSUB -n 32
#BSUB -J hello
#BSUB -o %J.out
#BSUB -e %J.err
#BSUB -q normal
#BSUB -W 0:15
echo "Master Host = "`hostname`
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"
echo "PWD_DIR: "`pwd`

ibrun ./hello
```

- To submit the job:

```
lslogin1% bsub < job
```

Re-direction is required!

SGE: Job Script Submission

- When submitting jobs to SGE using a job script, the script file should be the first argument to qsub. Consider the following script:

```
login3$ cat job.script
#$ -S /bin/csh
#$ -pe 16way 32
#$ -N hello
#$ -o ${JOB_ID}.out
#$ -e ${JOB_ID}.err
#$ -q normal
#$ -l h_rt=00:15:00
echo "Master Host = "`hostname`
echo "LSF_SUBMIT_DIR: $LS_SUBCWD"
echo "PWD_DIR: "`pwd`

ibrun ./hello
```

- To submit the job:

```
login3$ qsub job.script
```

LSF: Interactive Execution

- Several ways to run interactively
  - Submit entire command to bsub directly:

```
> bsub -q development -I -n 2 -W 0:15 ibrun ./hello

Your job is being routed to the development queue
Job <11822> is submitted to queue <development>.
<<Waiting for dispatch ...>>
<<Starting on compute-1-0>>
 Hello, world!
   --> Process #   1 of   2 is alive. ->compute-1-0
   --> Process #   0 of   2 is alive. ->compute-1-0
```

  - Submit using normal job script and include additional `-I` directive:

```
> bsub -I < job.script
```

SGE: Memory Limits

- Per process memory limits are enforced to ensure that physical memory is not over-allocated.
- Default parallel job submission allocates all 16 compute cores per node
- If you need more memory per MPI task, you can request fewer cores per node with the SGE parallel environment:

PE	Meaning
16way	16 MPI tasks per node, 1.92 GB memory/ task
8way	8 MPI tasks per node 3.84 GB memory/ task
4way	4 MPI tasks per node 7.68 GB memory/ task
2way	2 MPI tasks per node 15.36 GB memory/ task

- Please note that accounting charges are based on the node usage (not the core usage). A job using 4way will incur an SU charge four times larger than a default run using 16way (and requesting the same number of tasks)

TACC

Batch Script Suggestions

- Echo issuing commands
  - ("set -x" and "set echo" for ksh and csh).
- Avoid absolute pathnames
  - Use relative path names or environment variables ($HOME, $WORK)
- Abort job when a critical command fails.
- Print environment
  - Include the "env" command if your batch job doesn't execute the same as in an interactive execution.
- Use "./" prefix for executing commands in the current directory
  - The dot means to look for commands in the present working directory. Not all systems include "." in your $PATH variable. (usage: ./a.out).
- Track your CPU time

TACC

LSF/SGE Job Monitoring (*showq* utility)

```
lslogin1% showq
ACTIVE JOBS--------------------
JOBID     JOBNAME    USERNAME      STATE  PROC  REMAINING       STARTTIME

11318  1024_90_96x6    vmcalo    Running    64   18:09:19  Fri Jan  9 10:43:53
11352           naf   phaa406    Running    16   17:51:15  Fri Jan  9 10:25:49
11357           24N   phaa406    Running    16   18:19:12  Fri Jan  9 10:53:46
 23 Active jobs     504 of 556 Processors Active (90.65%)

IDLE JOBS----------------------
JOBID     JOBNAME    USERNAME      STATE  PROC    WCLIMIT       QUEUETIME

11169        poroe8      xgai       Idle   128   10:00:00  Thu Jan  8 10:17:06
11645   meshconv019    bbarth       Idle    16   24:00:00  Fri Jan  9 16:24:18
 3 Idle jobs

BLOCKED JOBS-------------------
JOBID     JOBNAME    USERNAME      STATE  PROC    WCLIMIT       QUEUETIME

11319  1024_90_96x6    vmcalo   Deferred    64   24:00:00  Thu Jan  8 18:09:11
11320  1024_90_96x6    vmcalo   Deferred    64   24:00:00  Thu Jan  8 18:09:11
 17 Blocked jobs

Total Jobs: 43    Active Jobs: 23    Idle Jobs: 3     Blocked Jobs: 17
```

LSF Job Monitoring (*bjobs* command)

```
lslogin1% bjobs
JOBID   USER    STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
11635   bbarth  RUN   normal     lonestar    2*compute-8 *shconv009 Jan  9 16:24
                                             2*compute-9-22
                                             2*compute-3-25
                                             2*compute-8-30
                                             2*compute-1-27
                                             2*compute-4-2
                                             2*compute-3-9
                                             2*compute-6-13
11640   bbarth  RUN   normal     lonestar    2*compute-3 *shconv014 Jan  9 16:24
                                             2*compute-6-2
                                             2*compute-6-5
                                             2*compute-3-12
                                             2*compute-4-27
                                             2*compute-7-28
                                             2*compute-3-5
                                             2*compute-7-5
11657   bbarth  PEND  normal     lonestar                *shconv028 Jan  9 16:38
11658   bbarth  PEND  normal     lonestar                *shconv029 Jan  9 16:38
11662   bbarth  PEND  normal     lonestar                *shconv033 Jan  9 16:38
11663   bbarth  PEND  normal     lonestar                *shconv034 Jan  9 16:38
11667   bbarth  PEND  normal     lonestar                *shconv038 Jan  9 16:38
11668   bbarth  PEND  normal     lonestar                *shconv039 Jan  9 16:38
```

Note: Use "`bjobs -u all`" to see jobs from all users.

SGE Job Monitoring (*qstat* command)

```
login4$ qstat -s a
job-ID  prior   name       user         state submit/start at     queue              slots
-----------------------------------------------------------------------------------------
  16414 0.12347 NAMD       user001      r     01/09/2008 15:13:58 normal@i101-302...   512
  15907 0.13287 tf7M.8     user001      r     01/09/2008 13:36:20 normal@i105-410...   512
  15906 0.13288 f7aM.7     user001      r     01/09/2008 13:33:47 normal@i171-401...   512
  16293 0.06248 ch.r32     user001      r     01/09/2008 14:56:58 normal@i175-309...   256
  16407 0.12352 NAMD       user001      qw    01/09/2008 12:23:21                      512
  16171 0.00000 f7aM.8     user001      hqw   01/09/2008 10:03:43                      512
  16192 0.00000 tf7M.9     user001      hqw   01/09/2008 10:06:17                      512
```

Basic *qstat* options:

-s {p|r|e|a|…} Display jobs with the specified status (a for all)
Default setting shows running jobs only

-u username Display jobs belonging to specified user (* for all users)

-t Display detailed information about controlled subtasks

-r Display extended job information

-g {c|d|t} Display grouping information according to cluster, job arrays or parallel jobs (c most commonly used)

SGE Job Manipulation/Monitoring

- To kill a running or queued job (takes ~30 seconds to complete):
  `qdel <jobID>`
  `qdel -f <jobID>` (Use when qdel alone won't delete the job)
- To suspend a queued job:
  `qhold <jobId>`
- To resume a suspended job:
  `qhold -r <jobID>`
- To see more information on why a job is pending:
  `qstat –r -j <jobID>`
- To see a historical summary of a job:
  `qacct –j <jobID>`

```
login4$ qacct -j 16110
==============================================================
qname        normal
hostname     i172-208.ranger.tacc.utexas.edu
...
qsub_time    Wed Dec 31 18:00:00 1969
start_time   Wed Jan  9 08:33:41 2008
end_time     Wed Jan  9 08:33:51 2008
...
```

Serial/Threaded Compilers (Intel/PGI)

Compiler	Program	Type Suffix	Example
icc/pgcc	C	.c	**icc [options] prog.c**
icpc/pgCC	C++	.C, .cc, .cpp, .cxx	**icpc [options] prog.cpp**
ifort/pgf77	F77	.f, .for, .ftn	**ifort -Vaxlib [options] prog.f**
Ifort/pgf90	F90	.f90, .fpp	**ifort -Vaxlib [options] prog.f90**

C	**icc -o prog [options] prog.c [linker options]**
F90	**ifort -o prog -Vaxlib [options] prog.f90 [linker options]**

MPI Compilation (*what you really want*)

Compiler	Program	Type Suffix	Example
mpicc	c	.c	**mpicc prog.c**
mpiCC	C++	.C, .cc, .cpp, .cxx	**mpiCC prog.cc**
mpif77	F77	.f, .for, .ftn	**mpif77 -Vaxlib prog.f**
mpif90	F90	.f90, .fpp	**mpif90 -Vaxlib prog.f90**

C	**mpicc -o prog [options] prog.c [linker options]**
F90	**mpif90 -o prog -Vaxlib [options] prog.f90 [linker options]**

Useful Compiler Options (Ranger)

PGI	Intel 10	Intel 9	Description
-O3	-O3	-O3	Aggressive serial optimizations
-ipa=fast,inline	-ipo / -ip	-ipo / -ip	Interprocedural optimization
-mp	-openmp	-openmp	Enable generation of OpenMP code
-tp barcelona-64	-xO	-xW -xT (Lonestar)	Enable generation of SSE instructions
-g –gopt	-g	-g	Include debugging symbols
-help	-help	-help	List help information

TACC

Math Libraries (Intel)

- MKL (Math Kernel Library)
  - LAPACK, BLAS, and extended BLAS (sparse), FFTs (single- and double-precision, real and complex data types).
  - APIs for both Fortran and C
  - www.intel.com/software/products/mkl/
    Example: mpicc -Wl,-rpath,$TACC_MKL_LIB -I$TACC_MKL_INC mkl_test.c -L$TACC_MKL_LIB -lmkl_em64t
- VML (Vector Math Library) [equivalent to libmfastv]
  - Vectorized transcendental functions.
  - Optimized for Pentium III, 4, Xeon, and Itanium processors.

TACC

Math Libraries (AMD)

- ACML (AMD Core Math Library)
  - LAPACK, BLAS, and extended BLAS (sparse), FFTs (single- and double-precision, real and complex data types).
  - APIs for both Fortran and C
  - http://developer.amd.com/acml.jsp
    Example: mpicc -Wl,-rpath,$TACC_ACML_LIB -I $TACC_ACML_INC acml_test.c -L$TACC_ACML_LIB –lacml_mp

TACC

Performance Libraries

- papi (NCSA Tools Hardware Performance Monitor)
  - Events, floats, instruction, data access, cache access, TLB misses (*4 counters available on Barcelona, 2 on Intel Woodcrest*)
  - http://www.ncsa.uiuc.edu/UserInfo/Resources/Software/Tools/PAPI
- TAU (Tuning and Analysis Utilities)
  - Portable profiling and tracing toolkit for performance analysis of parallel programs
  - www.cs.uoregon.edu/research/paracomp/tau/
  - Fortran 77/90, C, C++, Java
  - OpenMP, pthreads, MPI, mixed mode

TACC

Little vs Big Endian

- A byte is the lowest addressable storage unit on many machines.
- A “word”, often refers to a group of bytes.
- There are two different ways to store a word on disk and memory: Big Endian and Little Endian.
- Intel Pentium and AMD Opteron machines are Little Endian Machines.
- Most “big iron” machines are Big Endian: (Crays, IBMs, & SGIs. Macs (Motorola processors) are Big Endian machines.

TACC

Little vs Big Endian Storage

Word (least significant digits in B1) | B4 | B3 | B2 | B1 |

Little Ending

Little Endian means the bytes are stored from the least significant byte to the highest, beginning at the lowest address. The word is stored “little end first”

	B1	B2	B3	B4
Base Address	**+0**	**+1**	**+2**	**+3**

Big Ending

Big Endian means the bytes are stored from the most significant byte to the lowest, beginning at the lowest address. The word is stored “big end first”

	B4	B3	B2	B1
Base Address	**+0**	**+1**	**+2**	**+3**

TACC

Little vs Big Endian Conversion

- C code uses shift and “and” macro.
  - For 4-byte words: Shift all bytes to correct position and zero out everything else, then “or” components:

```
#define SWAP32(x) \
 x = ((((x) & 0xff000000) >> 24) | (((x) & 0x00ff0000) >>  8) | \
      (((x) & 0x0000ff00) <<  8 ) | (((x) & 0x000000ff) << 24))
```

  - For 8-byte words:
    - use a 4-byte pointer (a) to the data,
    - SWAP on both 4 byte groups and then exchange the first 4 bytes and last 8 bytes:

```
SWAP32(a[j]); SWAP32(a[j+1]);
atmp   = a[j];
a[j]     = a[j+1];
a[j+1]  = atmp;
```

Platform Independent Binary Files

- XDR
  - Developed by SUN as part of NSF
  - Using XDR API gives platform independent binary files
  - `man xdr_vector`
  - `man xdr_string`
- NETCDF – Unidata Network Common Data Form
  - Common format used in Climate/Weather/Ocean applications
  - http://my.unidata.ucar.edu/content/software/netcdf/docs.html
  - `module load netcdf; man netcdf`
- HDF - Hierarchical Data Format developed by NCSA
  - http://hdf.ncsa.uiuc.edu/

References

- www.tacc.utexas.edu/ {click on User Guides}
- www.redhat.com
- www-unix.mcs.anl.gov/mpi/
- www.tacc.utexas.edu/resources/user_guides/ssh_intro/
- www.rocksclusters.org/Rocks/
- www.pbspro.com/openpbs.html
- www.tacc.utexas.edu/resources/user_guides/intel/
- www.tacc.utexas.edu/resources/user_guides/mkl/
- MKL/VML /opt/intel/mkl8.1/doc (pdf & html)
- www.tacc.utexas.edu/resources/user_guides/modules/
- www-unix.mcs.anl.gov/romio/papers.html