High Performance Compute (HPC) Cluster

SCIAMA (pro: Shama) Quick Start

Author: G.Burton

Version: Version 1.0

Date: Apr 11

http://www.sciama.icg.port.ac.uk

INTRODUCTION

This document is intended as a quick start guide for Sciama. Detailed documentation on all subjects covered can be found on the Sciama website at:-

http://www.sciama.icg.port.ac.uk/

Function	Sciama Implementation
Application, compiler and library selection (Modules Environment)	➢ **module avail** ➢ **module add <package>** ➢ **module initadd <package>** ➢ **module list**
Text Editors	➢ **nedit** ➢ **gedit** ➢ **xemacs (under modules)**
File Managers	**Nautilus (on desktop)** **Midnight Commander (MC under modules)**
Compilers	**Gcc (under modules)** **Intel32 (under modules)** **Intel64 (under modules)**
MPI	**OpenMpi (under modules)** **IntelMpi (under modules)**
Job Submission and Queuing Mechanism (Maui / Torque)	➢ **qstat –u <username>** ➢ **qstat –tn1** ➢ **qsub <job script>** ➢ **qsub –IX** ➢ **qdel <jobid>**
File Systems	**$HOME is NFS** **$HOME/lustre is Lustre high performance parallel**
Data transfer to / from environment	➢ **scp** ➢ **rsysnc** ➢ **sftp**

WORKING IN THE ENVIRONMENT

It is assumed that the "Getting Started With Sciama" guide has been followed and you have now logged in. It is likely you will now want to do one of three things:-

- Run an application already loaded in the environment.

  You can see what applications are already installed using the "module avail" command. This is detailed in a later section. Once an application is selected via "module" no further setup is usually required (eg. All environmental variables PATH, LD_LIBRARY_PATH etc should be set).

- Get a new application installed so you can run it.

  If after issuing the "module avail" command you discover the application is not installed please contact sciama.support@port.ac.uk . Arrangements will be made to get the application installed (if possible). The installer will attempt to put the new application under the "module" command so others can use it.

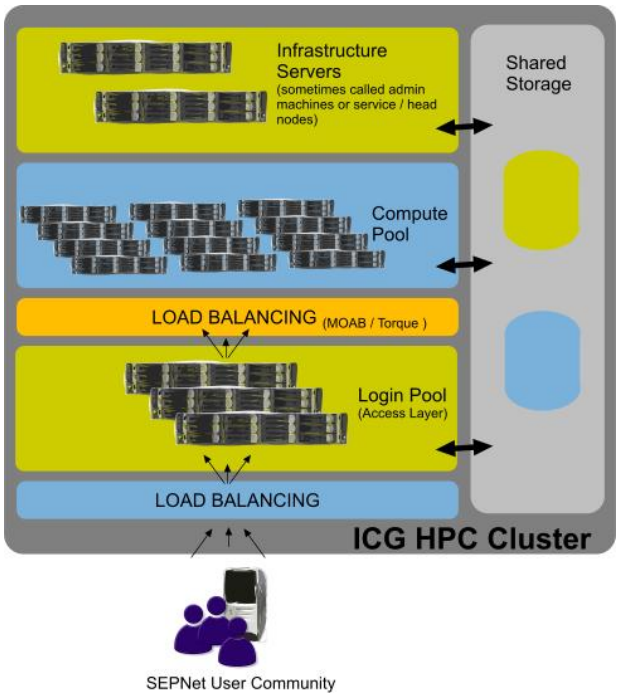
- Compile your own application from source code.

  You will need to select the compiler(s) and libraries using the "module" command described below. Once your code has been compiled it is likely you will need to write a small job submission script. This is also detailed below.

In all cases you will probably want to transfer data to the environment. This is described here:-
http://www.sciama.icg.port.ac.uk/datatx.htm

ARCHITECTURAL CONCEPT AND BASIC COMMANDS

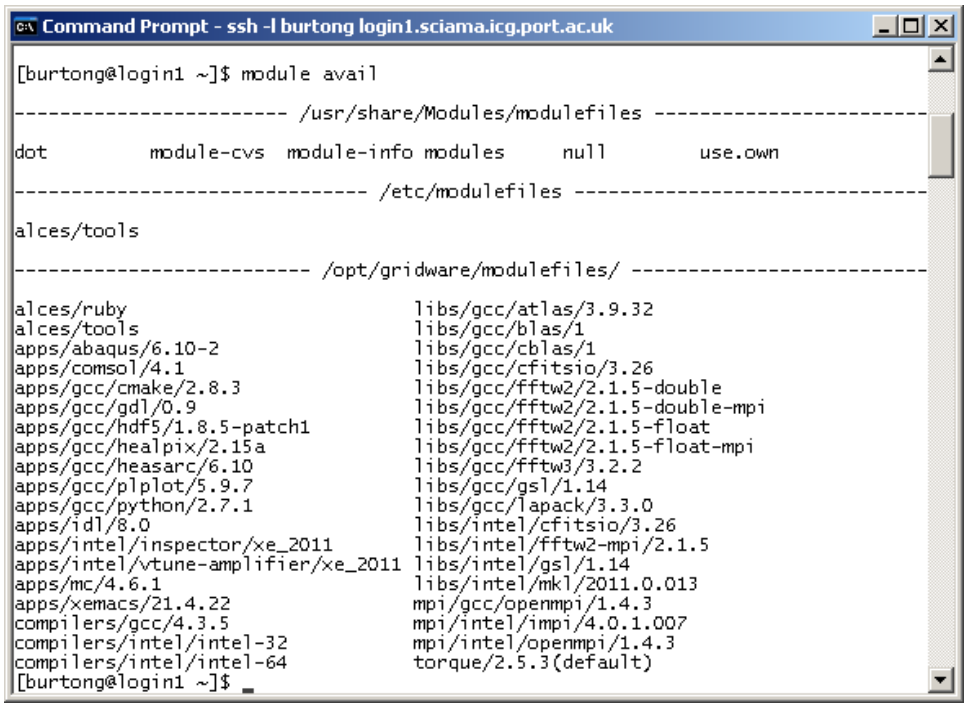
It is also assumed that the reader has a basic understanding of the High Performance Computing (HPC) concept.



1. Access is possible via a direct “ssh” connection or by the FreeNX remote desktop client.
2. Access to the environment is through one of the login servers (login1-4). Once in the environment any major applications (eg IDL, Abaqus) should be started on one of the compute nodes via the queuing system.
3. Your Home account resides on an NFS file system. Access to the Lustre file system is through $HOME/lustre. You will need to request an area on this file system. Data on either file system is **NOT** backed up.
4. You select the installed packages (compilers / libraries / application) using the “module” command. To see what has been installed type “module avail”.
5. The environment uses Maui / Torque to submit jobs to the compute pool. To view what is running type “qstat”.
6. To create an interactive shell on a compute server type “qsub –I” . To submit a batch job use “qsub <jobscript>” .
7. The environment supports both MPI and OpenMP.

.THE “MODULE” COMMAND

To see what has been installed on Sciama type “module avail” at the command prompt:-



You can either select modules for the current shell, eg:-

- module add apps/idl/8.0

or you can make the module available every time you create a new shell (inc. login):-

- module initadd apps/idl/8.0

In this example IDL version 8.0 has been selected. IDL can now be run from the command prompt simply by typing “idl” . Full details on running IDL can be found here:- http://www.sciama.icg.port.ac.uk/runningidl.pdf

To list what modules you have selected:-

- module list

To clear the list :-

- module clear

Full and detailed information on the “module” command can be found here:-

http://modules.sourceforge.net/

GNU COMPLIERS

The operating system on each node will have a bundled version of the GNU compilers (gcc, g++, gfortran) found under /usr/bin. For gcc this is version:-

[root@login1 stats]# gcc –v
Using built-in specs.
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --enable-shared --enable-threads=posix --enable-checking=release --with-system-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-libgcj-multifile --enable-languages=c,c++,objc,obj-c++,java,fortran,ada --enable-java-awt=gtk --disable-dssi --enable-plugin --with-java-home=/usr/lib/jvm/java-1.4.2-gcj-1.4.2.0/jre --with-cpu=generic --host=x86_64-redhat-linux
Thread model: posix

gcc version 4.1.2 20080704 (Red Hat 4.1.2-48)

Also in /usr/bin you will find gcc34 :-

[root@login1 stats]# gcc34 -v
Reading specs from /usr/lib/gcc/x86_64-redhat-linux/3.4.6/specs
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --enable-shared --enable-threads=posix --disable-checking --with-system-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-languages=c,c++,f77 --disable-libgcj --host=x86_64-redhat-linux
Thread model: posix

gcc version 3.4.6 20060404 (Red Hat 3.4.6-4)

You can select a later version of gcc using the Module command:-

[root@login1 stats]# module add compilers/gcc/4.3.5

[root@login1 stats]# gcc -v
Using built-in specs.
Target: x86_64-unknown-linux-gnu
Configured with: ../configure --prefix /opt/gridware/compilers/gcc/4.3.5/ --enable-shared --enable-threads=posix --enable-checking=release --enable-plugin --enable-languages=c,c++,objc,obj-c++,fortran
Thread model: posix

gcc version 4.3.5 (GCC)

GNU LIBRARIES

The following libraries have been compiled using gcc and are available with the “module” command:-

```
[burtong@login1]# module avail libs/gcc
---------------------------------- /opt/gridware/modulefiles/ ----------------------------------
libs/gcc/atlas/3.9.32          libs/gcc/fftw2/2.1.5              libs/gcc/fftw2/2.1.5-float-mpi
libs/gcc/blas/1                libs/gcc/fftw2/2.1.5-double       libs/gcc/fftw3/3.2.2
libs/gcc/cblas/1               libs/gcc/fftw2/2.1.5-double-mpi   libs/gcc/gsl/1.14
libs/gcc/cfitsio/3.26          libs/gcc/fftw2/2.1.5-float        libs/gcc/lapack/3.3.0
```

The mpi Libraries :-

```
mpi/gcc/openmpi/1.4.3
```

INTEL COMPILERS

Intel Cluster Studio version 12 is installed which includes the Intel compilers (icc, ifort). They are selectable using:-

> module add compilers/intel/intel-32

or

> module add compilers/intel/intel-64

[burtong@login1]$ icc -V
Intel(R) C Intel(R) 64 Compiler XE for applications running on Intel(R) 64, Version 12.0 Build 20101006
Copyright (C) 1985-2010 Intel Corporation. All rights reserved.

More details can be found here :-

http://www.sciama.icg.port.ac.uk/intelsw.htm

INTEL LIBRARIES

The following libraries have been compiled using the intel-64 compilers and are available with the "module" command:-

[burtong@login1]# module avail libs/intel

---------- /opt/gridware/modulefiles/ ----------

libs/intel/cfitsio/3.26 libs/intel/gsl/1.14

libs/intel/fftw2-mpi/2.1.5 libs/intel/mkl/2011.0.013

The mpi libraries:-

mpi/intel/impi/4.0.1.007

mpi/intel/openmpi/1.4.3

It should be noted that the openmpi version has a better Infiniband interface.

RUNNING JOBS IN THE SCIAMA ENVIRONMENT

You should never run any large processes on a login server. Login servers are shared by others so any intense processing will slow down the responsiveness others experience. All major applications / compilations should be done on a compute node. There are two ways to do this both using the "qsub" command to submit a job to a queue :-

- Create an interactive shell on a compute node:-

  qsub –IX # I=Interactive X=allow X server windows

  This will give you a command prompt in the current shell.

- Run a batch job on a compute node(s) :-

  qsub <job-script>

  This will start / queue a job according to the directives specified in the job script.

EXAMPLE JOB SCRIPTS

Job scripts comprise of shell commands and PBS directives, which are commands to tell the queuing system your requirements. Here is a simple example:-

```
#!/bin/bash
# This is an example job script – these lines are comments
# PBS directives are shown below
# Configure the resources needed to run my job
#PBS -l mem=700mb,walltime=1:30:00
# Merge stdout and stderr
#PBS -j oe
echo "Job starting at `date`"
~/test/myapplication -i ~/data/intputfile4
```

Don't confuse a comment **"#"** with a queuing system directive **"#PBS"** .

In a more complex example (MPI or OpenMP) you may need to use multiple cores:-

```
#!/bin/bash
#PBS -l nodes=65:ppn=12
#PBS -l walltime=6:00:00
#PBS -N gadget
#PBS -o out_gadget_blind8001.o$PBS_JOBID
#PBS -e err_gadget_blind8001.e$PBS_JOBID
#PBS -m abe
#PBS -M gary.burton@port.ac.uk
#PBS -V
#PBS -q cluster.q

# Configure modules and load modules
cd $PBS_O_HOME
. /etc/profile.d/modules.sh
. $HOME/modules_gadget

cd $PBS_O_WORKDIR
mpirun -np 780 /users/burtong/Gadget2_grid2400  /users/burtong/Gadget_Oriana_Blind_ir8001.param
```

In general a queue name does not need to be specified.

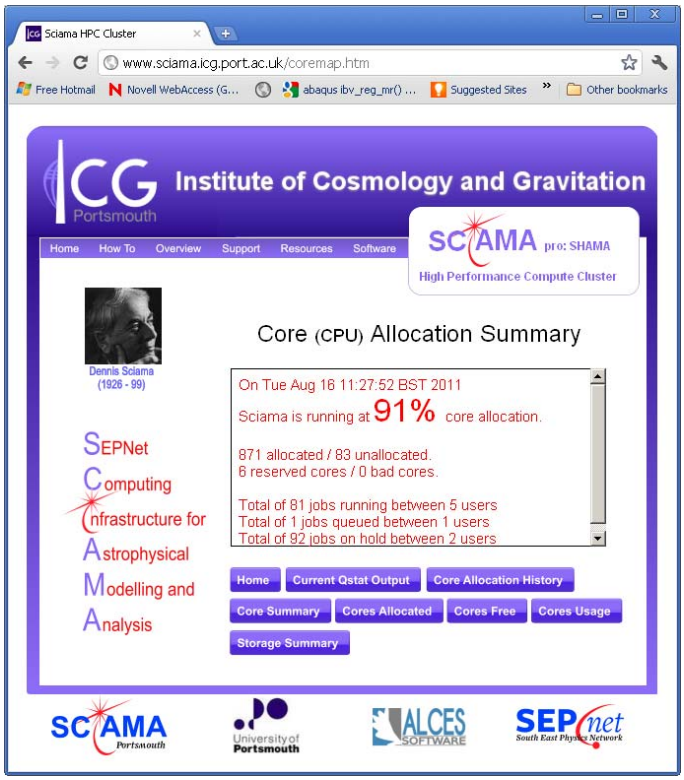
The default maximum runtime for a job is 48 hours. If a job needs to run for longer you need to specify the *#PBS -l walltime=60:00:00* directive (60 hours in this example).

Man pages are available for all queue commands:-

qdel qrls qhold qsig qstat qsub etc etc

JOB STATUS AND QUEUE INFORMATION

Up to date queue information can be obtained by logging into the environment. In addition a snapshot is taken every 20 minutes and displayed on the Sciama web site:-



http://www.sciama.icg.port.ac.uk/coremap.htm