Using Vera

John Urbanic
Parallel Computing Scientist
PSC / Physics Department

PITTSBURGH SUPERCOMPUTING CENTER

Carnegie Mellon University

Overview

- Apply for account
- Connecting
- System
- Account Management
- File Spaces
- Transferring Files
- Programming Environment
- Software Environment
- Running Jobs
- Support

Accounts

Getting a Vera account

- If you do not have an active account on any other PSC system you must first create a PSC username and password:
    - Create your PSC username by completing the form at vera-doc.psc.edu. You will receive an email message when your username has been created.
    - When your username is ready, create a PSC password (sometimes called "Kerberos password"). Go to the web-based PSC password change utility at apr.psc.edu to set your PSC password.
- If you are faculty, to request an allocation on Vera:
    - Complete the PSC account request form on vera-doc.psc.edu to request an allocation on Vera for your group. List the usernames for all the members of your group who should have access to this allocation in that form.
- If you are a student:
    - Have your advisor send email to grants@psc.edu asking to add you to their Vera account. Your advisor will need your PSC username in order to add you.

Changing your PSC/Vera password

There are two ways to change or reset your PSC password:

- Use the web-based PSC password change utility at apr.psc.edu
- Use the kpasswd command when logged into a PSC system. Do not use the passwd command.

When you change your PSC password, whether you do it via the online utility or via the kpasswd command on a PSC system, you change it on all PSC/Vera systems.

Connecting

- We take security very seriously! Be sure to read and comply with PSC policies on passwords, security guidelines, resource use, and privacy.

- When you connect to Vera, you are connecting to a Vera login node. The login nodes are used for managing files, submitting batch jobs and launching interactive sessions. *They are not suited for production computing.*

- Connect via ssh
    - Use an ssh client from your local machine to connect to hostname vera.psc.edu using the default port (22). You do not have to specify the port.

- Public-private keys
    - You can also use public-private key pairs to connect to Vera. To do so, you must first out the form at vera-doc.psc.edu to register your keys with PSC.

System Configuration

The Vera system consists of 28 compute nodes. They are identical, except that some nodes have 128GB of RAM and some have 256GB.

Node name	r001-r006, r009-020	r007-008, r021-028
RAM	256GB, DDR4-2133	128GB, DDR4-2133
CPUs	2 Intel Haswell (E5-2695 v3) CPUs; 14 cores/CPU; 2.3 - 3.3 GHz	
Cache	35MB LLC	
Node-local storage	2 HDDs, 4TB each	
Server	HPE Apollo 2000	

Carnegie Mellon University

PSC
PITTSBURGH SUPERCOMPUTING CENTER

File System

- The Vera filesystem is called verafs. You have a home directory and a shared group scratch directory.
- Your Vera home directory is /verafs/home/username, where username is your PSC username. Your home directory has a 5GB quota. Your home directory is backed up.
- Your group shares space available as /verafs/scratch/grantname. This scratch file space is NOT backed up.
- You can find your grant name by typing "id -gn".
- You can check your file usage using the command /opt/packages/allocations/my_quotas. Both your home directory and the scratch space available for your group are shown.
- There is a new 1 PB addition to the disk hardware mounted as /hildafs.   /hildafs/project/$grant/$username is not given to all users by default and will not be for some time. PIs can request it through the application form.

Permissions

- chmod: as per normal Unix practice

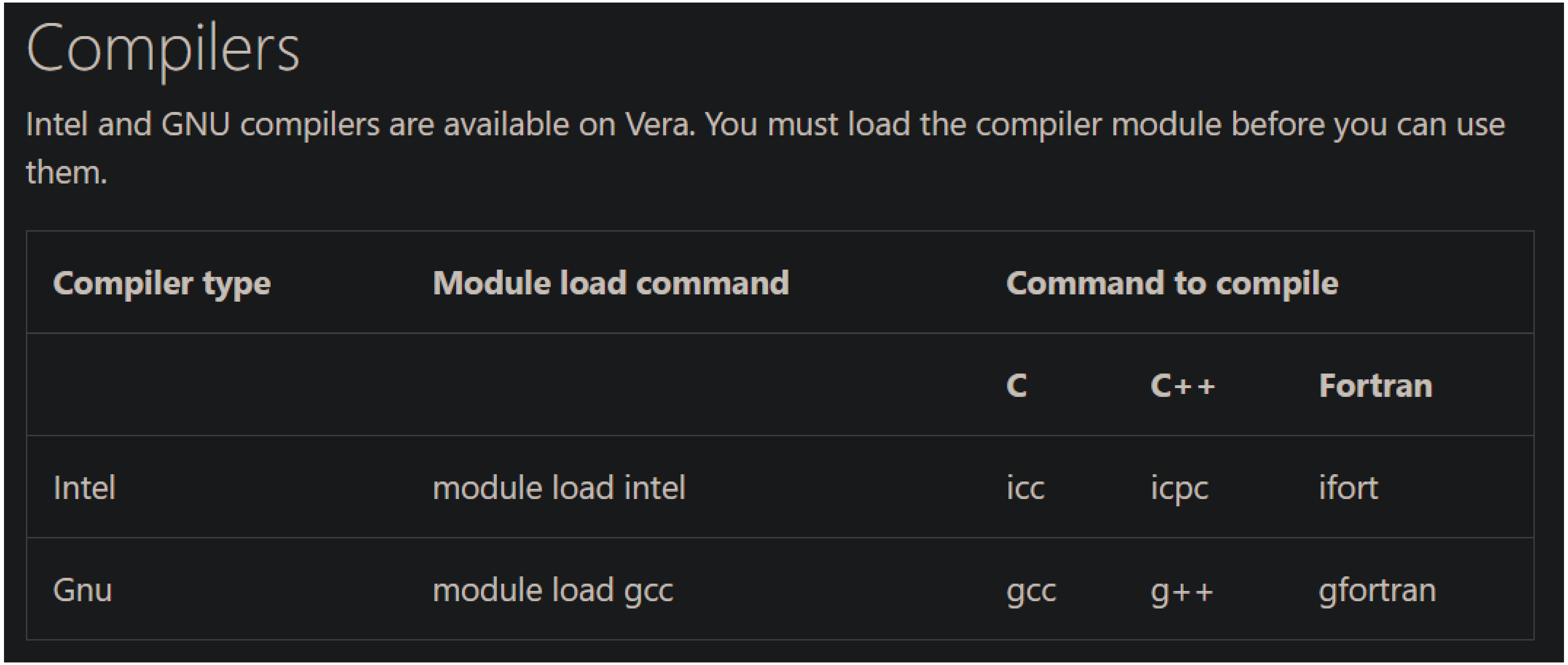
- ACLs: Finer grained control.
  - Ex: `setfacl -m user:janeuser:r filename`
  - Create any groups you want
  - man pages for setfacl and getfacl

Transferring Files

- You can use rync, scp or sftp to transfer files into and out of Vera.

- There are nodes dedicated to handling file transfers into Vera, named data.vera.psc.edu. Using these nodes will make file transfers more efficient.

Programming Environment



Compilers

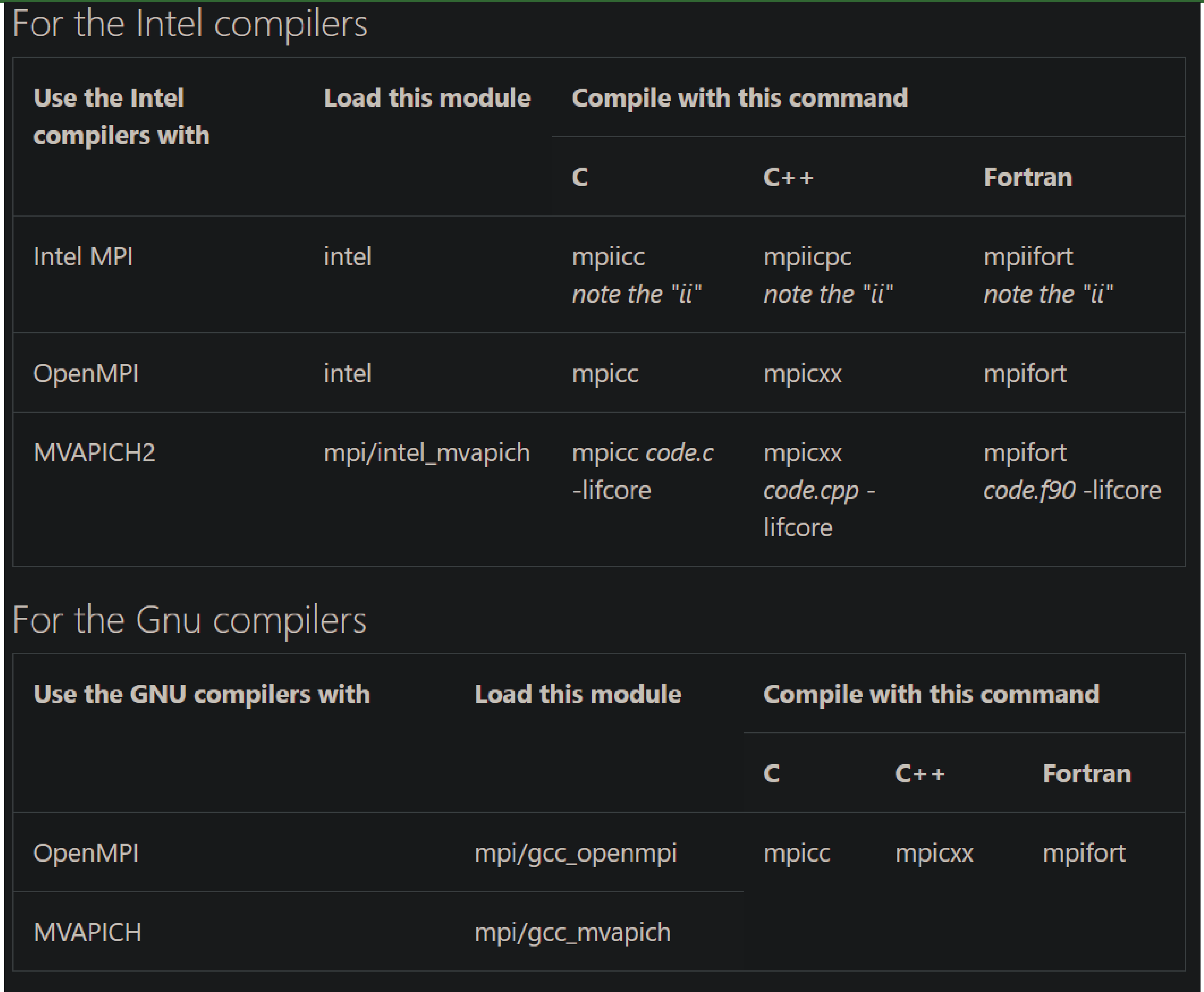
Intel and GNU compilers are available on Vera. You must load the compiler module before you can use them.

Compiler type	Module load command	Command to compile		
		C	C++	Fortran
Intel	module load intel	icc	icpc	ifort
Gnu	module load gcc	gcc	g++	gfortran

Carnegie Mellon University

PSC
PITTSBURGH SUPERCOMPUTING CENTER

MPI



For the Intel compilers

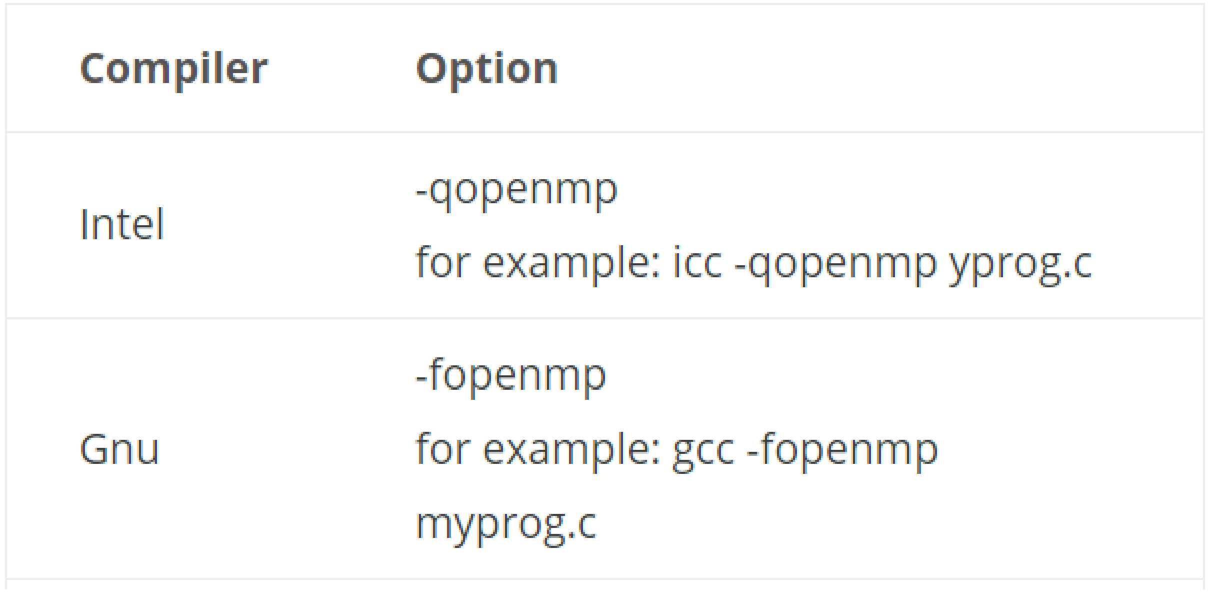
Use the Intel compilers with	Load this module	Compile with this command		
		C	C++	Fortran
Intel MPI	intel	mpiicc *note the "ii"*	mpiicpc *note the "ii"*	mpiifort *note the "ii"*
OpenMPI	intel	mpicc	mpicxx	mpifort
MVAPICH2	mpi/intel_mvapich	mpicc *code.c* -lifcore	mpicxx *code.cpp* -lifcore	mpifort *code.f90* -lifcore

For the Gnu compilers

Use the GNU compilers with	Load this module	Compile with this command		
		C	C++	Fortran
OpenMPI	mpi/gcc_openmpi	mpicc	mpicxx	mpifort
MVAPICH	mpi/gcc_mvapich			

OpenMP

- Make sure you have loaded the compiler module you desire.

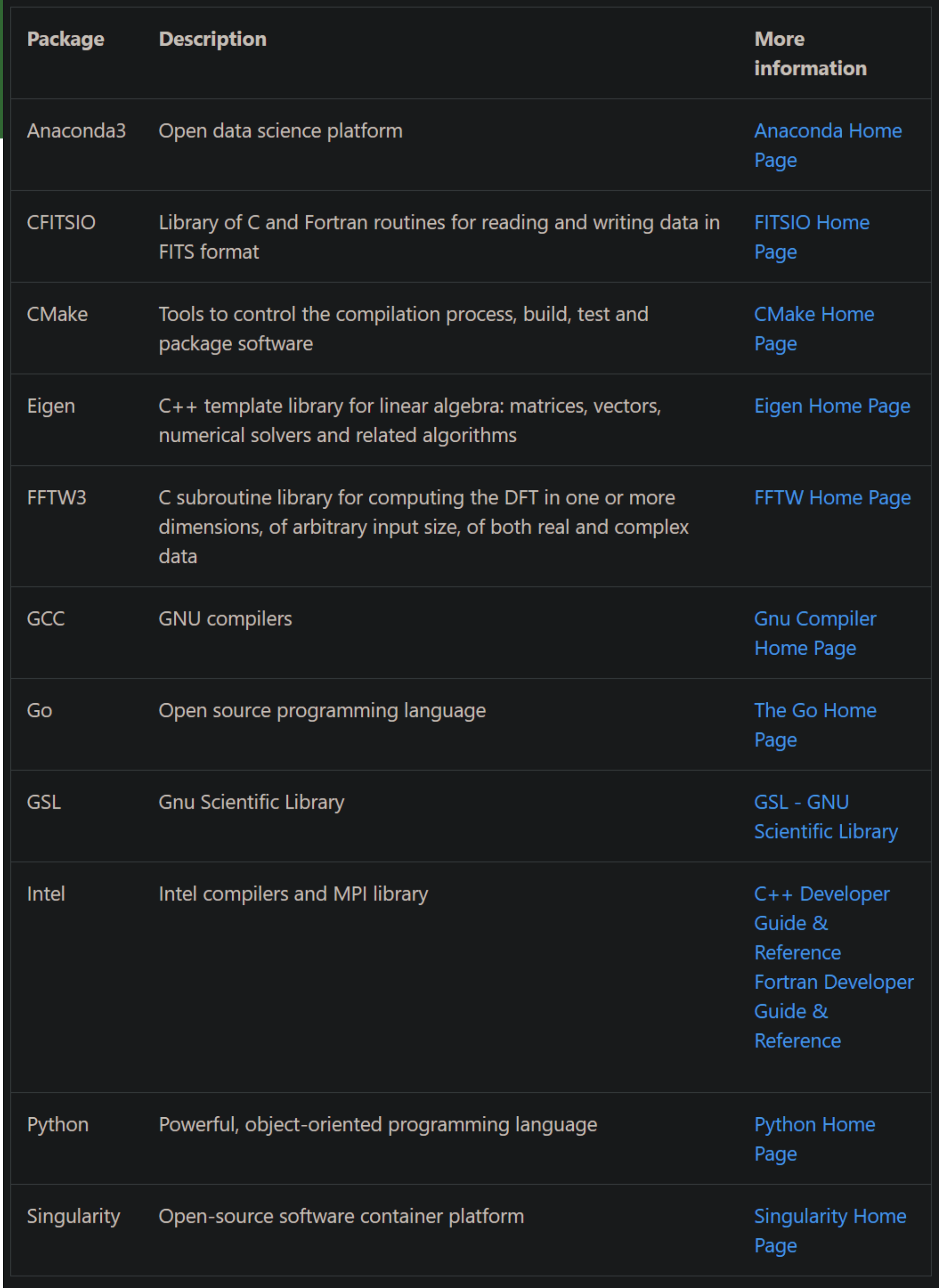


Compiler	Option
Intel	-qopenmp for example: icc -qopenmp yprog.c
Gnu	-fopenmp for example: gcc -fopenmp myprog.c

Software

Vera has a growing collection of applications installed.

- See the official list at https://vera-doc.psc.edu/#software
- There are also useful user instructions there.
- Most of these use the *module* command to enable them.



Package	Description	More information
Anaconda3	Open data science platform	Anaconda Home Page
CFITSIO	Library of C and Fortran routines for reading and writing data in FITS format	FITSIO Home Page
CMake	Tools to control the compilation process, build, test and package software	CMake Home Page
Eigen	C++ template library for linear algebra: matrices, vectors, numerical solvers and related algorithms	Eigen Home Page
FFTW3	C subroutine library for computing the DFT in one or more dimensions, of arbitrary input size, of both real and complex data	FFTW Home Page
GCC	GNU compilers	Gnu Compiler Home Page
Go	Open source programming language	The Go Home Page
GSL	Gnu Scientific Library	GSL - GNU Scientific Library
Intel	Intel compilers and MPI library	C++ Developer Guide & Reference Fortran Developer Guide & Reference
Python	Powerful, object-oriented programming language	Python Home Page
Singularity	Open-source software container platform	Singularity Home Page

Carnegie Mellon University

Modules

- The environment management package *module* is essential for running software on most PSC systems.
- Check if there is a module for the software you want to use by typing *module avail software-name.*
- To load the environment for a software package type *module load software-name*.
- module has a lot of other options. Try "module help" if you think you care.

Slurm is for you, not against you.

Slurm exists to aid your throughput - not just add an additional step to getting your work done. We structure the queues to facilitate that objective, and tune them based on your feedback. If we do that well, you just submit and don't notice.

If we didn't have Slurm, you would be at each others throats. *Maybe* an exaggeration, but I have seen HPC anarchy, and it isn't pretty. Even with Slurm, you can still be a jerk.

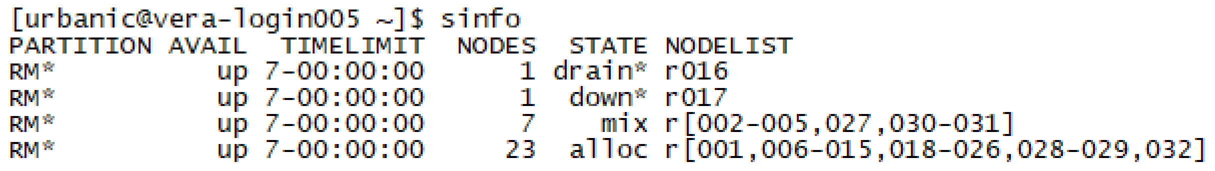
Carnegie Mellon University

PSC
PITTSBURGH SUPERCOMPUTING CENTER

Getting along.

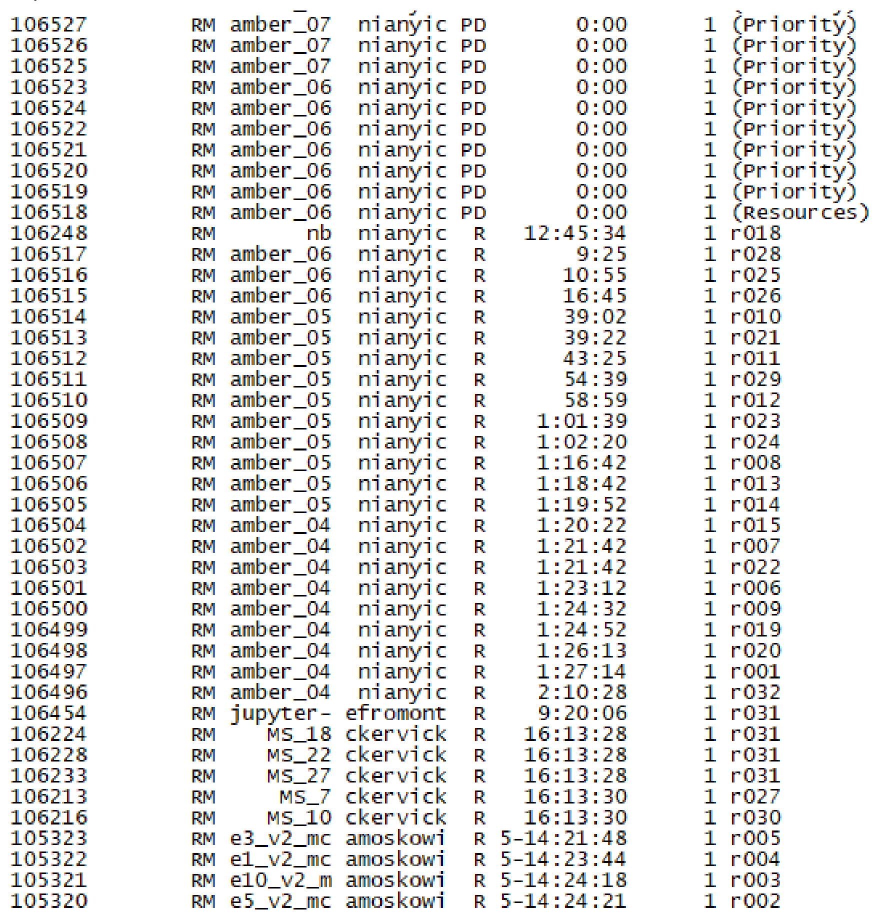
We will try to set, and adjust, queue policies so that everyone gets a fair share. But that is a subjective goal.

If you want to be a good citizen, you can be aware of the current use, and backlog, on the machine with the *squeue* and *sinfo* commands:



```
[urbanic@vera-login005 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
RM*          up 7-00:00:00      1 drain* r016
RM*          up 7-00:00:00      1  down* r017
RM*          up 7-00:00:00      7    mix r[002-005,027,030-031]
RM*          up 7-00:00:00     23  alloc r[001,006-015,018-026,028-029,032]
```

squeue

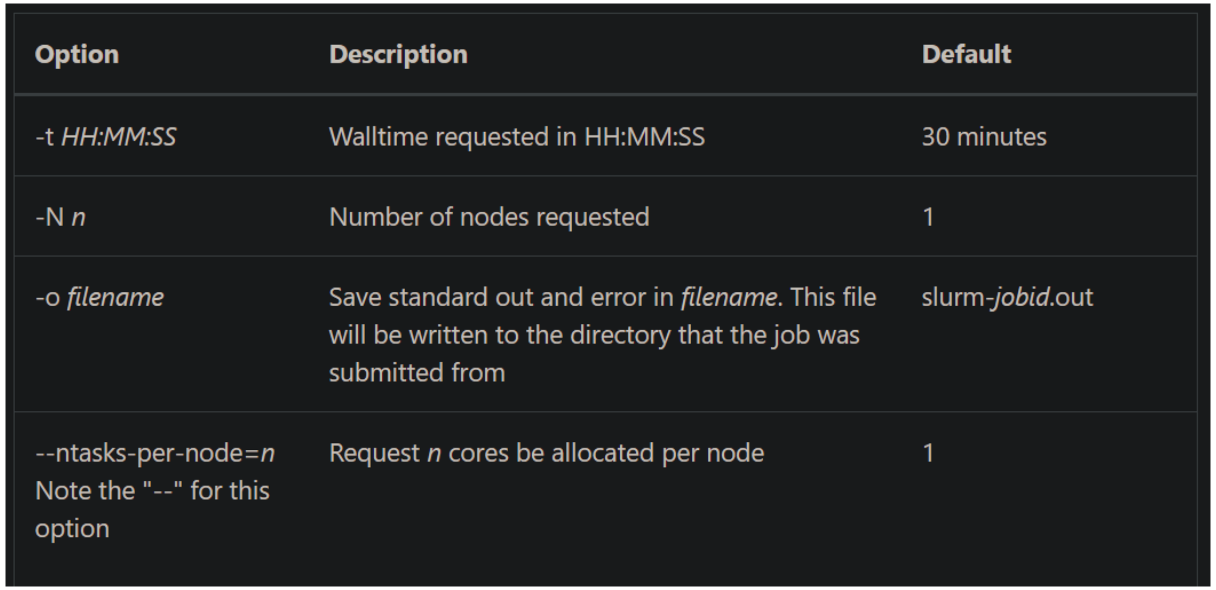


```
106527         RM amber_07  nianyic PD       0:00      1 (Priority)
106526         RM amber_07  nianyic PD       0:00      1 (Priority)
106525         RM amber_07  nianyic PD       0:00      1 (Priority)
106523         RM amber_06  nianyic PD       0:00      1 (Priority)
106524         RM amber_06  nianyic PD       0:00      1 (Priority)
106522         RM amber_06  nianyic PD       0:00      1 (Priority)
106521         RM amber_06  nianyic PD       0:00      1 (Priority)
106520         RM amber_06  nianyic PD       0:00      1 (Priority)
106519         RM amber_06  nianyic PD       0:00      1 (Priority)
106518         RM amber_06  nianyic PD       0:00      1 (Resources)
106248         RM       nb  nianyic  R   12:45:34      1 r018
106517         RM amber_06  nianyic  R       9:25      1 r028
106516         RM amber_06  nianyic  R      10:55      1 r025
106515         RM amber_06  nianyic  R      16:45      1 r026
106514         RM amber_05  nianyic  R      39:02      1 r010
106513         RM amber_05  nianyic  R      39:22      1 r021
106512         RM amber_05  nianyic  R      43:25      1 r011
106511         RM amber_05  nianyic  R      54:39      1 r029
106510         RM amber_05  nianyic  R      58:59      1 r012
106509         RM amber_05  nianyic  R    1:01:39      1 r023
106508         RM amber_05  nianyic  R    1:02:20      1 r024
106507         RM amber_05  nianyic  R    1:16:42      1 r008
106506         RM amber_05  nianyic  R    1:18:42      1 r013
106505         RM amber_05  nianyic  R    1:19:52      1 r014
106504         RM amber_04  nianyic  R    1:20:22      1 r015
106502         RM amber_04  nianyic  R    1:21:42      1 r007
106503         RM amber_04  nianyic  R    1:21:42      1 r022
106501         RM amber_04  nianyic  R    1:23:12      1 r006
106500         RM amber_04  nianyic  R    1:24:32      1 r009
106499         RM amber_04  nianyic  R    1:24:52      1 r019
106498         RM amber_04  nianyic  R    1:26:13      1 r020
106497         RM amber_04  nianyic  R    1:27:14      1 r001
106496         RM amber_04  nianyic  R    2:10:28      1 r032
106454         RM jupyter- efromont  R    9:20:06      1 r031
106224         RM    MS_18 ckervick  R   16:13:28      1 r031
106228         RM    MS_22 ckervick  R   16:13:28      1 r031
106233         RM    MS_27 ckervick  R   16:13:28      1 r031
106213         RM     MS_7 ckervick  R   16:13:30      1 r027
106216         RM    MS_10 ckervick  R   16:13:30      1 r030
105323         RM e3_v2_mc amoskowi  R 5-14:21:48      1 r005
105322         RM e1_v2_mc amoskowi  R 5-14:23:44      1 r004
105321         RM e10_v2_m amoskowi  R 5-14:24:18      1 r003
105320         RM e5_v2_mc amoskowi  R 5-14:24:21      1 r002
```

Slurm

Slurm is quite powerful, and it benefits you to take a look through the official documentation. However many of you will get by with just submitting jobs with *sbatch*, monitoring with *squeue* and occasionally canceling one with *scancel*.

Job arrays may be very interesting to those of you doing lots of parameter searches.



Option	Description	Default
-t *HH:MM:SS*	Walltime requested in HH:MM:SS	30 minutes
-N *n*	Number of nodes requested	1
-o *filename*	Save standard out and error in *filename*. This file will be written to the directory that the job was submitted from	slurm-*jobid*.out
--ntasks-per-node=*n* Note the "--" for this option	Request *n* cores be allocated per node	1

Likewise, many of you will be happy just using the basic functionality for your jobscripts. There are many more options.

Slurm sample serial job

What you probably really want is a sample jobscript to use as a template. Here you go, but you may also use the Vera User Guide for your source as we will shortly have more examples there, with detailed documentation.

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 5:00:00

#This job will ask for 1 core for 5 hours

#echo commands to stdout
set -x

#If you are using a package, you may want to load a module here

#run pre-compiled program which is sitting in the submission directory
./a.out
```

Slurm sample MPI job

```
#!/bin/bash
#SBATCH -N 2
#SBATCH -t 5:00:00
#SBATCH --ntasks-per-node=28

#this job will ask for 2 full nodes(56 cores) for 5 hours

#make sure to use the same module you used to compile
module load mpi/gcc_openmpi

#run pre-compiled MPI program which is sitting in the submission directory
mpirun -np $SLURM_NTASKS ./a.out
```

Interactive Slurm use

This is very convenient and I use it for much of my debugging and development. Be aware that you are occupying any requested resources whether you are active or not. So by polite and don't launch a long interactive session and walk away.

```
[urbanic@vera-login005 ~]$ srun --nodes=1 --ntasks-per-node=1 --time=01:00:00 --pty bash -i
srun: job 106722 queued and waiting for resources
srun: job 106722 has been allocated resources
[urbanic@r007 ~]$ mpirun -n 1 a.out
Hello from Process 0 running on r007.opa.vera.psc.edu
[urbanic@r007 ~]$
```

If you are using modules, don't forget to load them in your interactive environment as well.

Support

To report a problem on Vera, please email help@psc.edu. Please report only one problem per email. Be sure to include:

- an informative subject line
- your username
- if the question concerns a particular job, include these in addition:
- the JobID
- any error messages you received
- the date and time the job ran
- link to job scripts, output and data files
- the software being used, and versions when appropriate
- a screenshot of the error or the output file showing the error, if possible