University of Cape Town • iYunivesithi yaseKapa • Universiteit van Kaapstad •

SPES BONA

UCT
eResearch
ACCELERATING RESEARCH

Advanced HPC course

Agenda

1) Quick Overview of HPC (15 mins)
2) HPC Job Submission (60 min)

break (30 min)

3) Software Compile / Installs / Misc (60min)

Module 1:
Quick Overview of HPC

What is HPC ?

- HPC, or high-performance computing, refers to the application of supercomputers or clusters of computers to computational problems that typically arise through scientific inquiry.
- HPC is useful when a computational problem:
  - **<u>Is too large</u>** to solve on a conventional laptop or workstation (because it requires too much memory or disk space) or …
  - **<u>Would take too long</u>** (because the algorithm is complex, the dataset is large, or data access is slow) or …
  - **<u>Are too many</u>** – High Throughput Computing

Reasons to use UCT HPC ?

- You have a program that can be recompiled or reconfigured to use optimized numerical libraries that are available on HPC systems but not on your own system.
- You have a "parallel" problem, e.g. you have a single application that needs to be rerun many times with different parameters.
- You have an application that has already been designed with parallelism.
- To make use of the large memory available.
- Our facilities are reliable and regularly backed up.

When not to use HPC ?

- Cannot host databases on HPC, flat file databases are allowed but not Client\Server databases.

- Graphical User Interface (GUI) applications can be used but users are asked to be cautious. OpenOnDemand is coming…

Parallelism on HPC

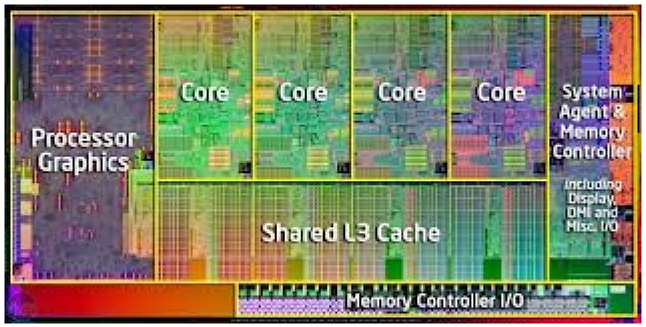
- Programs for HPC systems must be split up into many smaller “sub-programs“ which can be executed in parallel on different processors

- Writing <u>parallel software can be challenging</u>, and many existing software packages do not support parallelism & may require development.

<u>**NOTE: Some tasks cannot be parallelised**</u>

What does HPC consist of ?

- HPC is the aggregation of computing resources.
  - Cores (cpus / sockets)
  - RAM
  - Disk
  - Interconnect



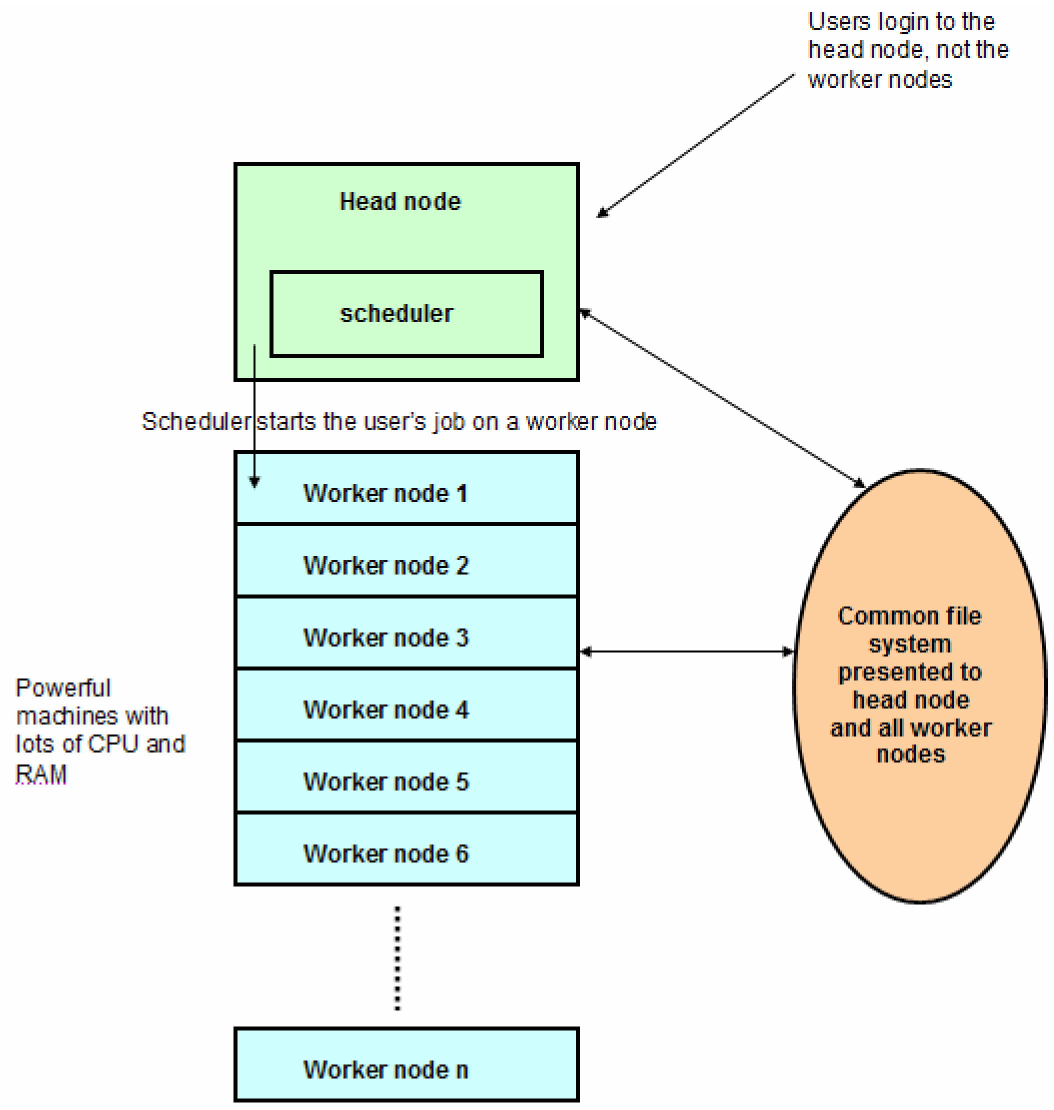
Cluster Architecture

- Operating system: Centos 7.4
- X86_64
- Scheduler: SLURM
- Worker nodes:
  - 34 Dell C6420 – Multi Core
  - 3 Dell C6145 – Many Core / dense array
  - 10 GPU servers (Tesla M2090/K40/K80/P100/A100)
  - 2 High Memory Machines – Dell R820 – 1TB RAM

BeeGFS Storage nodes:

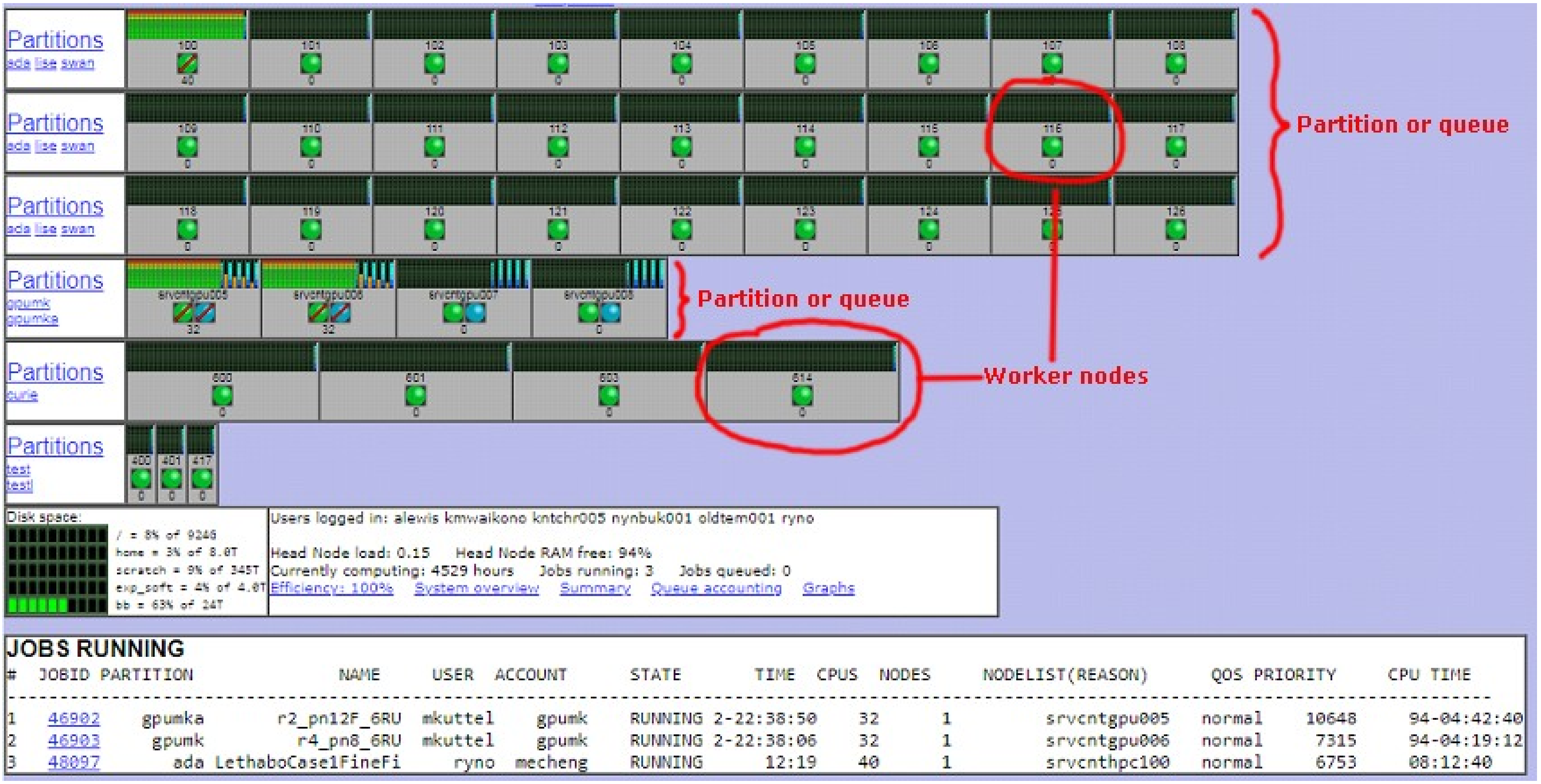
- 4 Dell R740s
- 360TB of scratch storage

Architecture


Users login to the head node, not the worker nodes
Head node
scheduler
Scheduler starts the user's job on a worker node
Worker node 1
Worker node 2
Worker node 3
Worker node 4
Worker node 5
Worker node 6
Worker node n
Powerful machines with lots of CPU and RAM
Common file system presented to head node and all worker nodes

The dashboard

- **To keep track of the cluster's status, workload and the jobs that are running go to: http://hpc.uct.ac.za/db**



Disk space:

```
/ = 8% of 924G
home = 3% of 8.0T
scratch = 9% of 345T
exp_soft = 4% of 4.0T
bb = 63% of 24T
```

Users logged in: alewis kmwaikono kntchr005 nynbuk001 oldtem001 ryno

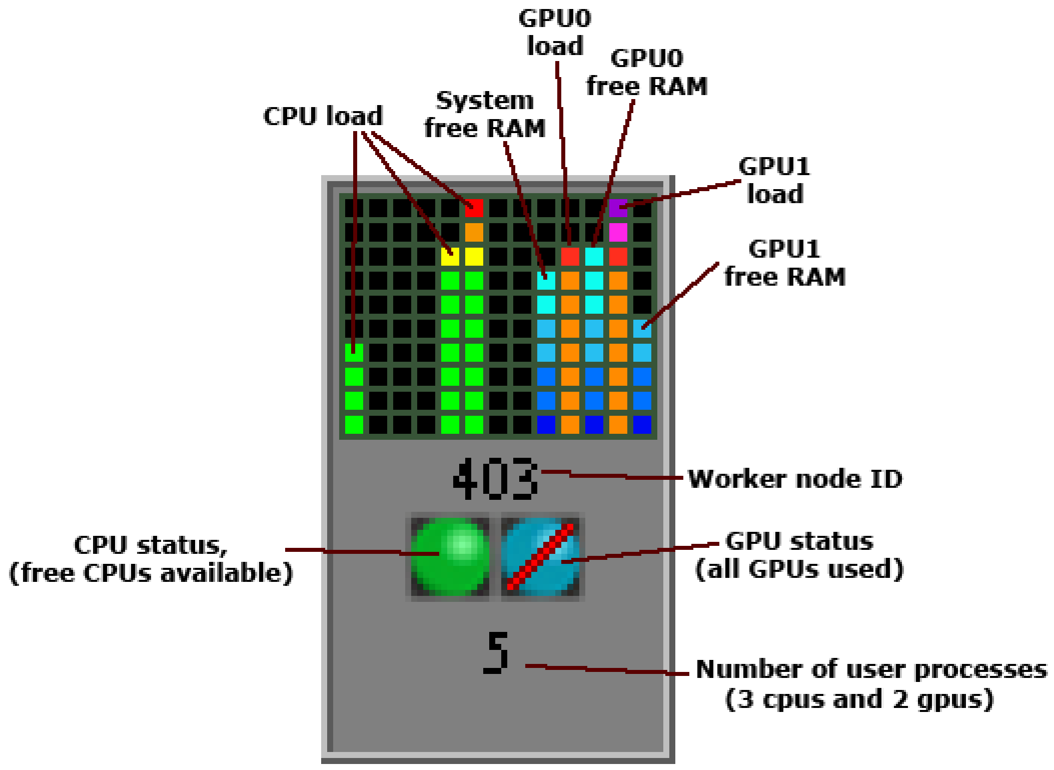
Head Node load: 0.15   Head Node RAM free: 94%
Currently computing: 4529 hours   Jobs running: 3   Jobs queued: 0
Efficiency: 100%   System overview   Summary   Queue accounting   Graphs

JOBS RUNNING

```
#  JOBID PARTITION              NAME     USER  ACCOUNT      STATE        TIME  CPUS  NODES      NODELIST(REASON)    QOS PRIORITY     CPU TIME
---------------------------------------------------------------------------------------------------------------------------------------------
1  46902    gpumka       r2_pn12F_6RU  mkuttel    gpumk    RUNNING 2-22:38:50    32      1          srvcntgpu005  normal    10648   94-04:42:40
2  46903     gpumk        r4_pn8_6RU  mkuttel    gpumk    RUNNING 2-22:38:06    32      1          srvcntgpu006  normal     7315   94-04:19:12
3  48097       ada LethaboCase1FineFi     ryno  mecheng    RUNNING      12:19    40      1          srvcnthpc100  normal     6753   08:12:40
```

The dashboard

Icon	Value	Description
🟢	Free CPUs	There are free CPUs, jobs may be submitted to this node.
⊘	Job-exclusive	All CPUs are busy, the node is running but no further jobs may be submitted.
🟡	Busy	Torque mom daemon or CPUs too busy to respond to further requests. Jobs are running but may be degraded.
🔴	Down	Node down or PBS mom daemon offline or not responding, no jobs may be submitted.
🔵	Free GPUs	There are free GPUs, jobs may be submitted.
⊘	Busy	All GPUs are busy, the node is running but no further jobs may be submitted.


GPU0 load
GPU0 free RAM
System free RAM
CPU load
GPU1 load
GPU1 free RAM
403
Worker node ID
CPU status, (free CPUs available)
GPU status (all GPUs used)
5
Number of user processes (3 cpus and 2 gpus)

BeeGFS Parallel Storage

- Pure software solution for scale-out parallel network-storage.

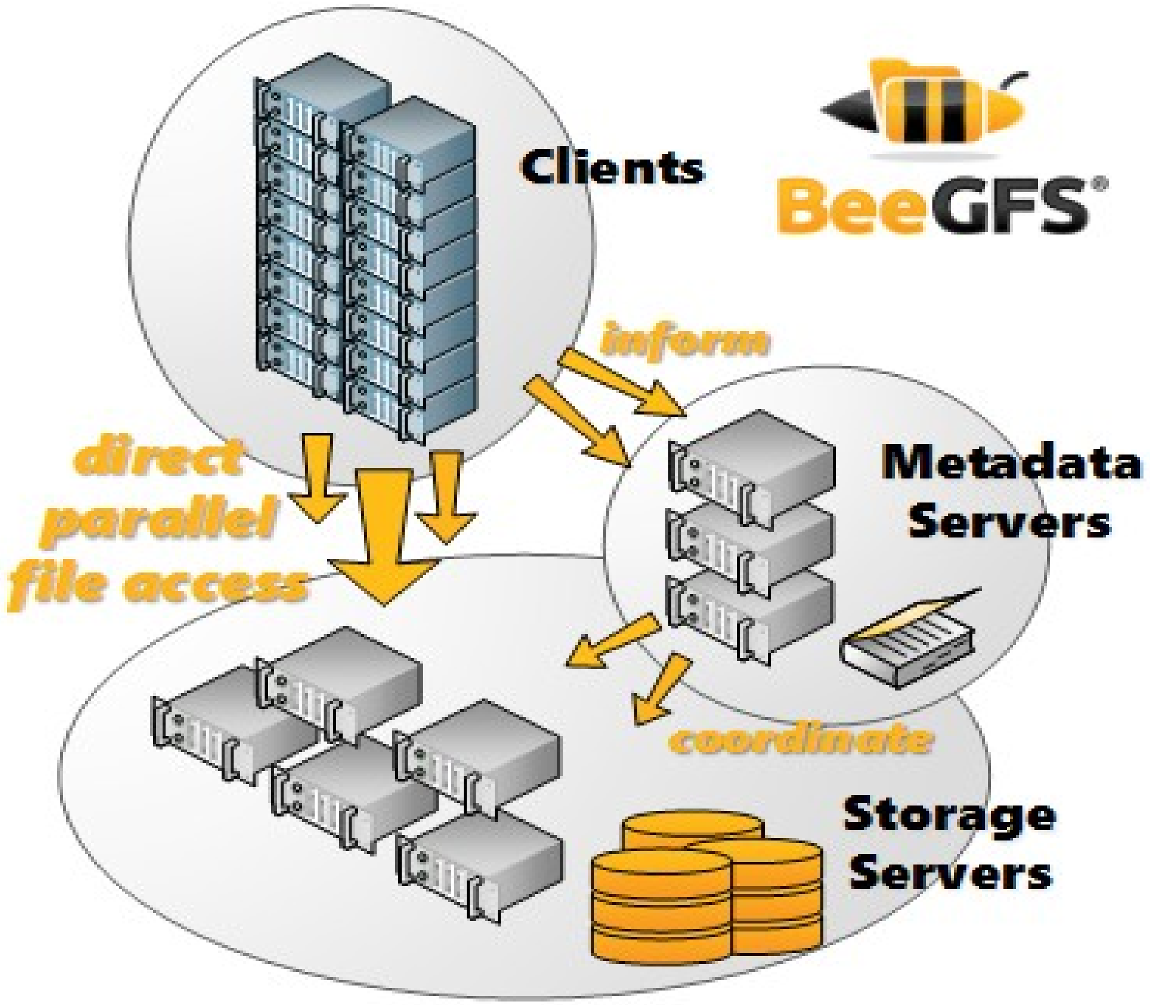
- Each HPC node is connected with IB cables to the IB switch. The BeeGFS store is connected to the same switch.

  - /scratch

- Advantages : Very fast storage

- Disadvantages: No backups, “volatile” area, scrubbing policy available on the HPC website

BeeGFS Architecture



BeeGFS connected to HPC

- Parallel storage is connected via Infiniband (RDMA only). The only TCP connection which exists is for Admon / MGMT services.
- TCP is the backup protocol should RDMA (IB switch) fail.
- Headnode is identical to worker nodes and also has IB.
- Once your job executes on a worker node, traffic to the storage service is ~100Gb/sec.
- Parallel software can also communicate over IB if compiled with OpenMPI and PMIx.

Module 2:
Various Job Submission Methods – Interactive

Software Required

Use your web browser to download Putty and PuttySCP from: http://www.putty.org

- Click on the “Download Putty” link and download:
  - putty.exe (a Telnet and SSH client)
  - WinSCP (GUI-BASED SCP)

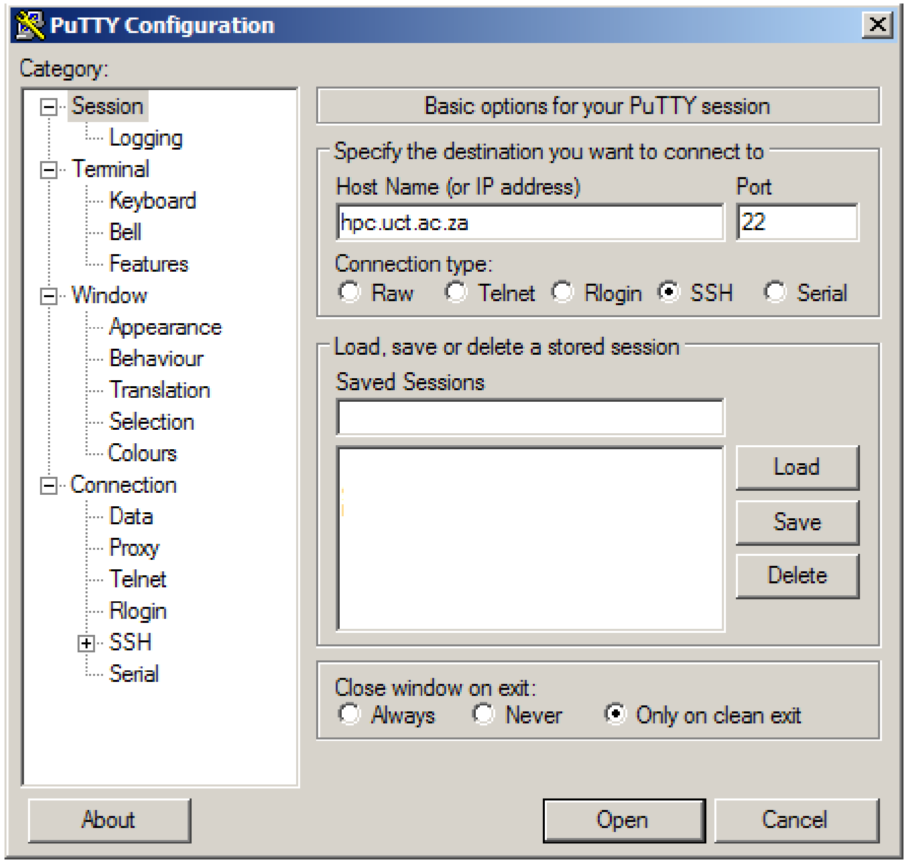
- Double click to install on your PC.
- MacOS users may launch a terminal

- Xming for Windows (**Tick NoACL**) / MacOSX users may use Quartz

Course Credentials

- Start the putty telnet/ssh client by double clicking on putty.exe and connect to the HPC Machine
  - Host: **`hpc.uct.ac.za`**
  - Connection Type: **ssh**
  - Port: **22**

```
1. Select SSH, X11, Enable X11
2. Click on session, top left.
3. Saved sessions: hpc
4. Click the save button.
```


PuTTY Configuration

Category:

Session: Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): hpc.uct.ac.za

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

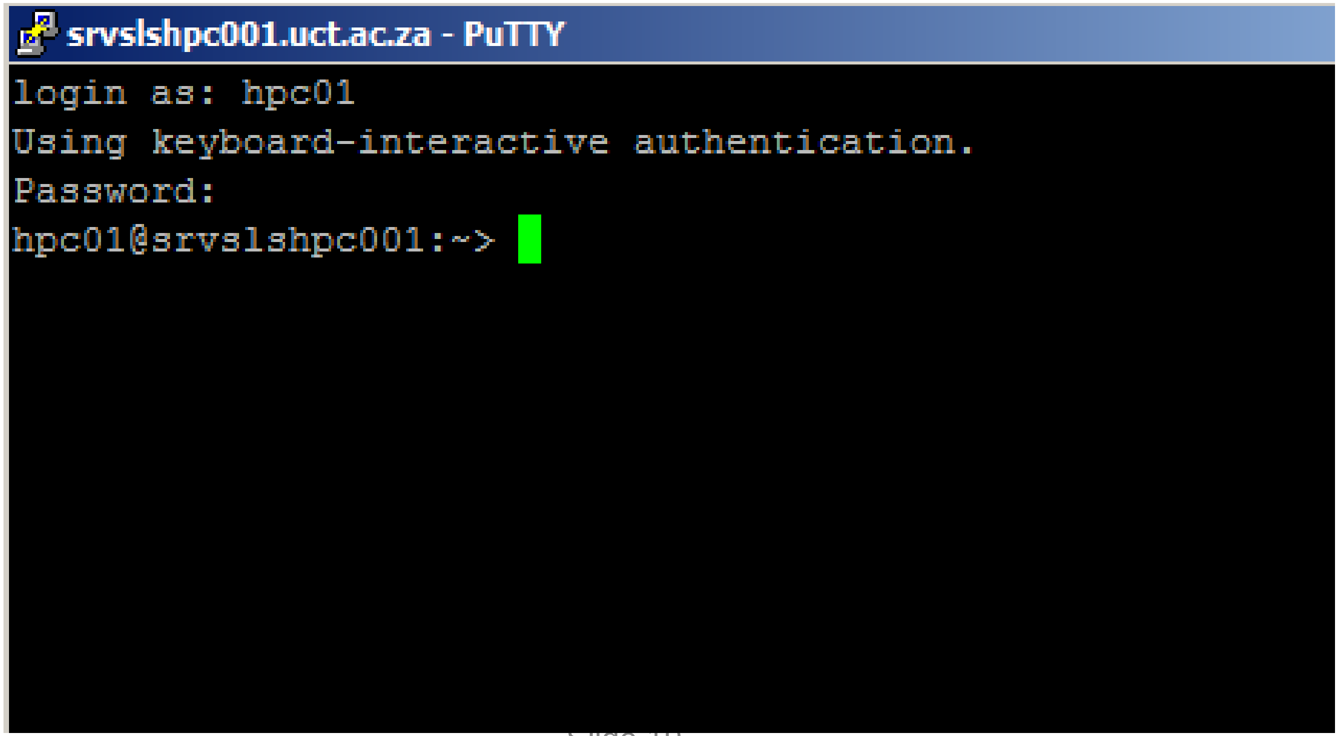
Load, save or delete a stored session

Saved Sessions

Close window on exit: Always Never [✓ Only on clean exit]

Course Credentials

- Log into the training HPC system using the Test Account allocated to you, e.g.
  - **Account Name:** hpc0(n)
  - **Password:** **will not be displayed as you type**



```
login as: hpc01
Using keyboard-interactive authentication.
Password:
hpc01@srvslshpc001:~>
```

Download training material

```
wget http://hpc.uct.ac.za:/db/training-
material.tgz

tar zxvf training-material.tgz

cd scripts
```

Modules

- Switching between different versions of the same application.
- Use Case: Job requires functionality from one or more applications, e.g. more modern R or Python than the standard installation.
- Sets up Library / Include / Bin / Custom Paths
- module avail - Lists all modules available
- module load <module> - Loads a specific module

The Environment

- These are variables set in your shell.
- Modules also forms part of and can control your environment.
- All environment variables exported to jobs by default.
- These can be controlled via the SBATCH directive
  - **--export**=<*environment variables [ALL] | NONE*>
- You can override the environment by placing commands in .bashrc or using –export
- But beware, you can mess up your shell\job.

The Environment

- Type env
- Now type env | grep PATH
- Now type echo $PATH
- The environment is a set of variables you can control
- Run an interactive job and type env | grep SLURM
- ***Your HPC job is just an SSH session full of environment variables!***

Standard Job Submission

```
#!/bin/bash
#SBATCH --account icts
#SBATCH --partition=ada
#SBATCH --nodes=1 --ntasks=1
#SBATCH --time=10:10:00

pwd
date
hostname
```

Rules:
No space before #SBATCH
No space between # and SBATCH

Exercise 1 (a)

Add #SBATCH --mail-user=user@email
Add #SBATCH –mail-type=ALL
sbatch standard-job.sh

Jobs Arrays

- Use Case: Lots of input files, not possible to submit manually.

- Common PBS Environment Variables
    - SLURM_ARRAY_TASK_ID=1
    - SLURM_ARRAY_TASK_COUNT=3
    - SLURM_ARRAY_TASK_MAX=3
    - SLURM_ARRAY_TASK_MIN=1

Exercise 1 (b)

sbatch --array=1-6 array-job.sh
sbatch --array=2-6:2 array-job.sh (Step array – process every 2nd job)
sbatch --array=2,4,6 array-job.sh

Interactive Jobs

- salloc
- Dangerous as you’re still on the head node.
- srun to run on cluster
- Type exit when done!!!

Exercise 1 (c)

```
salloc
hostname
srun hostname
```

Interactive Jobs

Exercise 1 (d)

Your previous allocation is still running, so clean it up…

squeue | grep <myuser> **(or just type qstat)**
exit

Now, remember that a SLURM job is just an ssh session and environment variables…?

```
salloc --ntasks=2
env
srun env
```

What do you see?

Interactive Jobs

- srun -- account=icts --time=10:00:00 --partition ada --ntasks=5 --pty bash -l

  There is a shortened wrapper script:
  sintx

- Use Case: Compiling, Debug Application / Testing,

- Advantage: Work directly on a worker node

- Disadvantage: CPU expensive. Get done and exit

Exercise 1 (e)

- **sintx --ntasks=5**
- **hostname**
- **exit**

X support

- Must run putty with X forwarding
- Display variable must be set up and exported to job
- All variables exported in example below
- Hint, Start\Run\cmd\ipconfig

Exercise 1 (f)

- **export DISPLAY=137.158.X.Y:0.0**
- **sbatch X-job.sh**
- **qstat**
- **Kill eyes window**
- **qstat**

MPI Jobs

- Message Passing Interface (MPI) is used for communication among the nodes running a parallel program on a distributed memory system.
- Compile mpitest.c - “mpicc -o mpitest mpitest.c”
- “sbatch mpi-job.sh“
- Important to use srun from the same openmpi version.

MPI Jobs

Exercise 1 (g)

- module load compilers/gcc820 mpi/openmpi-4.0.1
- Compile mpitest.c - “mpicc mpi.c -o mpitest”
- “sbatch mpi-job.sh“
- Here we are running 4 cores on 1 node
- The compile command can be part of the job.

MPI Jobs

Exercise 1 (h)

- We could also use salloc:

```
salloc  --ntasks=4 --nodes=2
module load mpi/openmpi-4.0.1
hostname     (we’re still on the head node)
srun mpitest     (this runs on cluster nodes)
exit      (please don’t forget to relinquish resources!!!)
```

- Why don’t we get an even spread of node resources, ie why are we given 3 cores on 1 node and 1 core on another?
- …to preserve as many low usage nodes as possible.
- To insist on an even spread add --ntasks-per-node=2

Modules Exercise 1(i)

module avail	Shows all modules available
module load python/anaconda-python-2.7	Environment modified for application
which python	Location for which binary
module unload python/anaconda-python-2.7	Unload the module

```
#SBATCH --job_name “Tea Time”
#SBATCH --time=00:30:00
```

Module 3:
Software Compile / Installs / Misc

The HPC software repository does not contain my software

- All software resides in /opt/exp_soft and shared between the HPC worker nodes. Please do not install on /scratch.

- Problem: I have a RPM file but cannot install because I do not have root privileges. Solution:

  ```
  rpm --prefix=/home/username/install-dir -i app.rpm
  ```

- Problem: I have the source but its such a mission to compile. Solution: (1)Make a list of dependencies, (2)download install, (3) Compile and view logs

- Package your application into a Singularity container and ship to HPC.

Let us establish a HPC interactive session

- Please do not run software installs from the head node!
- Start an interactive job:

```
sint --account=icts --time=10:00:00 --partition ada --ntasks=5
```

PEAR - Paired-End reAd mergeR

- Software for merging raw illumina paired-end reads
- One of many Open Source tools in the Bio-Informatics software catalogue.
- Quick and simple to compile.
- Compiling applications from scratch allows for extensive customization and integration.

Let’s compile some software

- Initiate a interactive job session using “sintx --ntasks=2”
- mkdir ~/pear-install
- Change directory into ~/training-material/software-src/
- Uncompress with “ tar xfvz pear-0.9.6-src.tar.gz “
- Change directory into pear-0.9.6-src
- “./configure -- help “ for a list of features and tuning parameters
- “./configure --prefix=/home/username/pear-install/”
- “make -j 2” - Compile the application. “ –j 2 “ ??
- “make install” - Install the compiled binary / lib / include

Working remotely with screen

- Allows you create additional virtual terminals inside a single process called “ Screen “
- Use Cases:
  - Works great for unrealible internet connections
  - Long running compilations / file copies
- Execute the command called “ screen “
- “ctrl + a +c “ - Create additional terminals
- “ctrl +a + n or p” - Move back / forward between tty
- “ctrl +a +d “ - Detach from a screen session
- “screen -r -d “ - detach and re-attach
- “screen -x “ - reattach but keep my remote sys active

Note: By default, screen sessions does not scroll.

- “ termcapinfo xterm|xterms|xs|rxvt ti@:te@ “

Road Map for UCT HPC 2023

- Capability to allow departments to purchase their own hardware to intergrate into HPC.
- HPC Operating System upgrades.
- Introduction of a new /scratch storage array.
- OpenOnDemand WebUI for HPC application usage.

Thank You