Sherlock for IBIIS

William Law
Stanford Research Computing

Overview

- How we can help
- System overview
  - Tech specs
  - Signing on
- Batch submission
- Software environment
- Interactive jobs
- Next steps

We are here to help

We run a wide variety of systems

We can connect you to resources here and nationally

Today I’m going to focus on sherlock

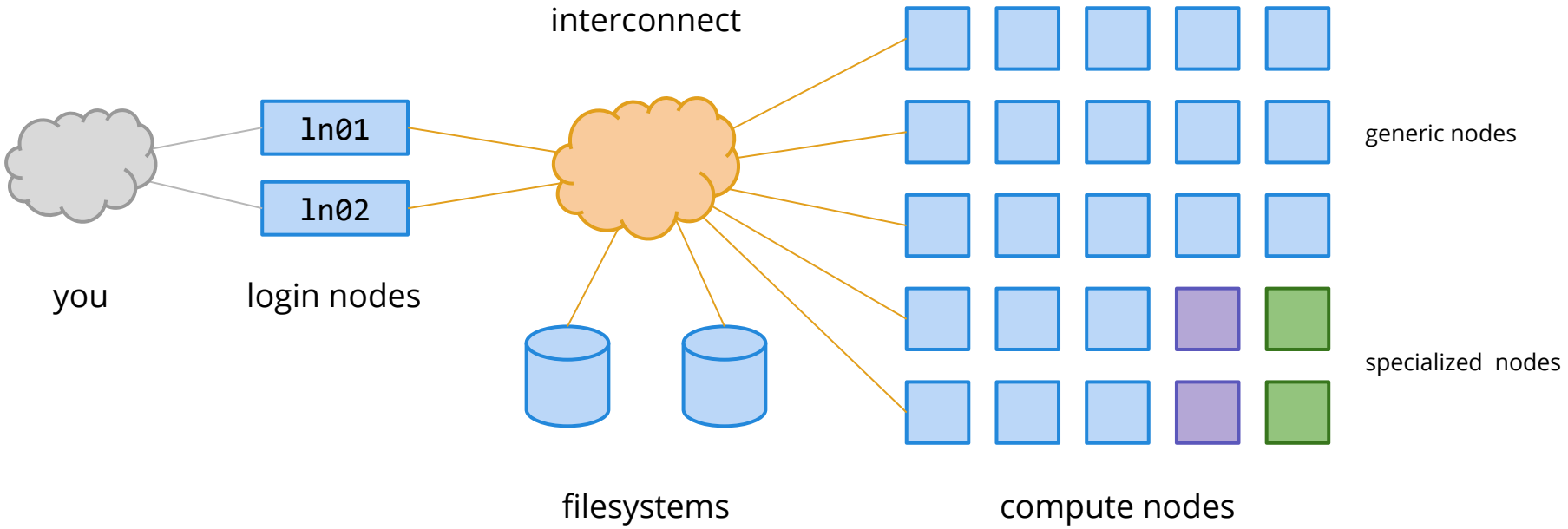
Why Sherlock?

Sherlock is a shared platform for research at Stanford

A single node of sherlock is more powerful than your laptop or desktop

Sherlock consists of many computers (nodes) which you can run your work on

System Overview



Login Nodes

Currently two machines

use: `sherlock.stanford.edu`

Login nodes = gateway to all other resources

Compute Nodes

Normal nodes

- 16 cores in two sockets
- 64GB RAM
- 100GB Solid State Disk
- Infiniband connection 56Gbps (2:1 oversubscribed)

GPU nodes (NVIDIA Kepler)

Big memory nodes (1.5TB RAM)

Storage

`/home: $HOME` (15GB), `$PI_HOME` (1TB)

small amount of space, backed up, small file friendly, accessible across all systems

`/scratch: $SCRATCH, $PI_SCRATCH`

large, high performance, not backed up, big file friendly, accessible across all systems

`/local_scratch: $LOCAL_SCRATCH`

SSD for duration of the job, local to a node

Storage Specs

/home

NFS
IOPS = ~16,200
Bandwidth = 125 MB/s based on connection

/scratch

Lustre
IOPS =~ 54,000
Bandwidth = 6.5 GB/s based on connection
total ~20 GB/s read; 13 GB/s write

/local_scratch

SSD
IOPS=100,000
Bandwidth = 500 MB/s read; 400 MB/s write

Sign On

Setup Kerberos

documented here:

http://sherlock.stanford.edu/mediawiki/index.php/SetupKerberos

Logon

2 steps: kinit then ssh

http://sherlock.stanford.edu/mediawiki/index.php/LogonCluster

Scheduling Jobs

Basic concept - tell the scheduler:

1. what you need
2. what it should do

Partitions

sets of physical machines with comparable hardware

QOS

sets of limitations (runtime, CPUs per user...)

No need to specify QOS or partition in most cases

A bit on scheduling

Sherlock currently uses fairshare scheduling

- balances out usage across users and PIs

Uses backfill scheduling

- short jobs can run before larger pending jobs

Accurate requirements help the scheduler

- and you! (shorter wait times)

See different partitions/QOS at http://sherlock.stanford.edu

Condo model

- General partition
  - everybody can run there
- Owners' partitions
  - owners have immediate access to their own nodes
  - they can run on other owners' nodes, but could be preempted by the rightful owner's jobs

Slurm tools

Run

`salloc`	reserve nodes
`srun`	run a scheduled command
`sbatch`	submit a batch script
`scancel`	cancel a job

Check

`squeue`	check job status and queue
`sstat/sacct`	check job stats while/after completion
`sprio`	view job priority factors
`sshare`	view shares (fairshare scheduling)

Sample batch file

```
#!/bin/bash
#SBATCH --job-name=R-hemisphere
#SBATCH --output=hemisphere.out
#SBATCH --error=hemisphere.err
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<sunetid>@stanford.edu
#SBATCH --time=20:00
#SBATCH --nodes=1
#SBATCH --mem=4000
#SBATCH --ntasks-per-node=1

module load R

srun R --no-save < hemisphere.R
```

Environment Modules

Allows to manage software versions

`module avail` shows what is available

`module load` loads a package

```
------------------------------------------------------------ /share/sw/modules/Core ------------------------------------------------------------
   R/3.0.2                     freesurfer/5.3.0  (D)    intelmpi/4.1.3.048         openmpi/1.6.5/intel13sp1up1 (D)    python/3.3.2      (D)
   allinea/4.2                 fsl/5.0.6-test           julia/gcc/0.3.0-pre        openmpi/1.7.4/gcc                  rstudio/0.98.501
   ansys/icemcfd/15.0          fsl/5.0.6         (D)    matlab/R2012b              openmpi/1.7.4/intel13sp1up1 (D)    rstudio/0.98.983  (D)
   comsol/43b                  gcc/4.8.1                matlab/R2013b              paraview/4.1.0                     sas/9.4
   comsol/44             (D)   gcta/1.24.2              matlab/R2014a        (D)   plink/1.07                         stata/12
   cuda/5.5                    intel/13sp1up1           nag/m24                    plink/1.9_b1                (D)    stata/13          (D)
   freesurfer/5.3.0-test       intel/13sp1.2.144 (D)    openmpi/1.6.5/gcc          python/2.7.5                       tabix/0.2.6
```

Interactive Jobs for starting

Any resource can be interactive

- A custom tool (`sdev`) makes interactive jobs easier
- The dev partition allows jobs to start reasonably quickly
- You might not be able to *run* for that long

To get a quick debug shell

```
$ sdev
```

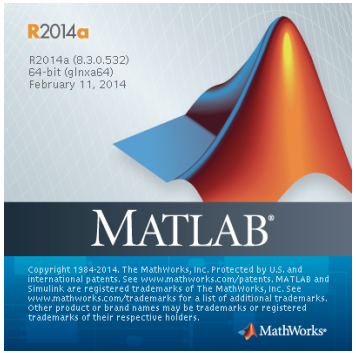
To run interactively (not just for debugging)

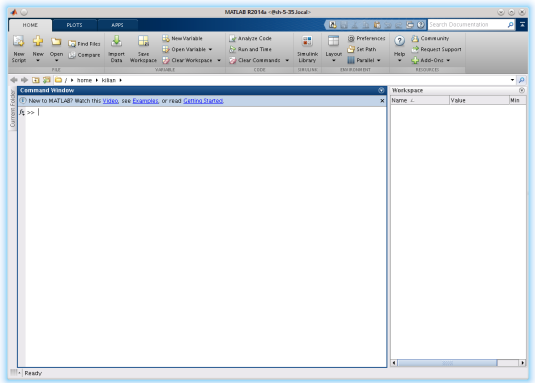
```
$ srun --pty bash
```

Graphical interfaces

Example with Matlab

```
sherlock-ln01$ srun  -p dev --qos dev  --pty --x11 --time=1:0:0 --mem=8GB bash
sh-5-2$ ml load matlab/R2014a
sh-5-2$ matlab
```





Next Steps

Request an account - email research-computing-support@stanford.edu, cc your PI

Get started, try submitting some jobs

- interactive jobs, batch scripts

Documentation (WiP)

- http://sherlock.stanford.edu