Contents

- 1 Back to the Parent Page
- 2 BERTINI FRONT-END MACHINES
- 3 CLUSTER SPECIFICATION
- 4 MODULES
- 5 JOB SCRIPTS
- 6 PARALLEL ENVIRONMENTS
- 7 JOB SUBMISSION AND MONITORING
  - 7.1 Well-formed Job Scripts
  - 7.2 Job Deletion
  - 7.3 Job Resource Monitoring

Back to the Parent Page

BERTINI FRONT-END MACHINES

**hostnames:** bertini1.crc.nd.edu, bertini2.crc.nd.edu, and bertini3.crc.nd.edu

You can securely log into the front-end BERTINI machine using a ssh client e.g.

> ssh ?Y netid@bertini1.crc.nd.edu

CLUSTER SPECIFICATION

Sommese Skoll (Newer Xeon)

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell PowerEdge R410 Servers</td>
<td>@sommese_dqcneh</td>
<td>9</td>
<td>8</td>
<td>Dual Quad-core</td>
<td>12GB</td>
</tr>
</tbody>
</table>

Sommese Skoll (Older Xeon)

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Western Scientific ASUS DSBF-DE Servers</td>
<td>@sommese_dqcxeon</td>
<td>28</td>
<td>8</td>
<td>Dual Quad-core</td>
<td>8GB</td>
</tr>
</tbody>
</table>

Sommese Large Memory (New)
## BertiniHardware

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell R815 AMD &quot;Abu Dahbi&quot; Servers</td>
<td>@sommese_q16copt</td>
<td>4</td>
<td>64</td>
<td>Quad 16-core</td>
<td>128GB</td>
</tr>
</tbody>
</table>

### Sommese Large Memory (Older)

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP ProLiant SL165z G7 Servers</td>
<td>@sommese_d8copt_96GB</td>
<td>2</td>
<td>16</td>
<td>Dual Eight-Core</td>
<td>96GB</td>
</tr>
</tbody>
</table>

### Hauenstein (New)

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell R815 AMD &quot;Abu Dahbi&quot; Servers</td>
<td>@hauenstein_q16copt</td>
<td>4</td>
<td>64</td>
<td>Quad 16-core</td>
<td>128GB</td>
</tr>
</tbody>
</table>

### Hauenstein (From NCState)

<table>
<thead>
<tr>
<th>Description</th>
<th>Host Group Name</th>
<th>Nodes</th>
<th>Cores/Node</th>
<th>Node Specification</th>
<th>Memory/Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dell R815 AMD &quot;Abu Dahbi&quot; Servers</td>
<td>@hauenstein_q16copt</td>
<td>12</td>
<td>64</td>
<td>Quad 16-core</td>
<td>128GB</td>
</tr>
</tbody>
</table>

## MODULES

The software environment on Bertini cluster is managed with *modules*. You can easily modify your programming and/or application environment by simply loading and removing the required modules. The most useful module commands are:

- `? module avail` (view the complete module list)
- `? module load xyz` (load the module xyz)
- `? module unload xyz` (unload the module xyz)
- `? module list` (list currently loaded modules)
JOB SCRIPTS

Jobs are submitted to the compute nodes via the Univa Grid Engine (UGE) *batch submission system*. Basic UGE batch scripts should conform to the following template:

```csh
!/bin/csh
#$ -M netid@nd.edu       # Email address for job notification
#$ -m bea                # Send mail when job begins, ends and aborts
#$ -pe mpi-64 128        # Specify parallel environment and legal core size
#$ -q *@@bertini         # Specify queue
#$ -N job_name           # Specify job name
module load bertini              # Required modules
mpiexec -np $NSLOTS bertini input_file # Application to execute
```

PARALLEL ENVIRONMENTS

Parallel job scripts must request a *parallel environment* for execution:

1. *smp* (parallel jobs running within a single machine)
2. *mpi-64* (parallel jobs running across multiple 64-core machines)
3. *mpi-16* (parallel jobs running across multiple 16-core machines)
4. *mpi-8* (parallel jobs running across multiple 8-core machines)

*Note:* If no parallel environment is requested (i.e. you do not specify a *?pe* parameter), then the default execution environment is a single-core serial job.

When requesting a parallel environment you must also specify a valid core size. Legal core sizes for the parallel environments are:

- *smp* (1, 2, 3, …, 64)
- *mpi-64* (a multiple of 64)
- *mpi-16* (a multiple of 16)
- *mpi-8* (a multiple of 8)

JOB SUBMISSION AND MONITORING

Job scripts can be submitted to the UGE batch submission system using the *qsub* command:

```bash
> qsub job.script
```

Once your job script is submitted, you will receive a numerical *job id* from the batch submission system, which you can use to monitor the progress of the job.

Well-formed Job Scripts

Job scripts that are determined by UGE to have made valid resource requests will enter the queuing system with a *queue-waiting*(qw) status (once the requested resources become available, the job will enter the
Bertini_Hardware

Running (r) status. Job scripts that are determined not to be valid will enter the queuing system with an error queue-waiting (Eqw) status.

To see the running/queued status of your job submissions, invoke the qstat command with your username (netid) and observe the 'status column:

> qstat ?u username

For a complete overview of your job submission, invoke the qstat command with the job id:

> qstat ?j job_id

Note: The main reasons for invalid job scripts (i.e. having Eqw status) typically are:

1. Illegal specification of parallel environments and/or core size requests
2. Illegal queue specification
3. Copying job scripts from a Windows OS environment to the Linux OS environment on the front-end machines (invisible Windows control codes are not parsed correctly by UGE). This can be fixed by running the script through the dos2unix command

Job Deletion

To delete a running or queued (e.g. submissions with Eqw status) job, use the following command:

> qdel ?j job_id

Job Resource Monitoring

To better understand the resource usage (e.g. memory, cpu and I/O utilization) of your running jobs, you can monitor the runtime behavior of your job's tasks as they execute on the compute nodes.

To determine the nodes on which your tasks are running, enter the following qstat command along with your username. Record the machine names (e.g. q16copt003.crc.nd.edu) associated with each task (both MASTER and SLAVE):

> qstat -u username -g t