Qlogin for Interactive Sessions

You can use qlogin on the HPC cluster system to get an interactive SGE session. Qlogin will use the SGE scheduler to select the next available host on the cluster and mark the slots as active as in any other job. It can select hosts based on complexes, etc. just like the qsub command can, including selecting parallel environments. This can be useful for development and prototyping to get an environment on identical hardware to what your production job will run on. Some examples of this would be

1. interactive testing on a large memory node
2. testing on a node using more cores than what would be appropriate on a login node
3. benchmarking the performance on a node that is identical to what will be running production jobs to get a more accurate idea of the performance

Those are just some examples of when using qlogin would be a good idea. There are also some guidelines that we ask you to follow when using qlogin sessions.

Requesting a qlogin session will try to allocate as many job slots as possible on a node, with a minimum of 1 job slot. It is possible that you will receive an entire node but that is not guaranteed. By attempting to allocate as many slots on a node as possible we try to increase the probability that you will have sufficient resources to experiment on the node. If you need a specific number of slots and/or require specific resources then you will need to make your qlogin resource requests explicit.

1. Qlogin sessions should **not** be used to "reserve" nodes or slots.
2. Please only launch the number of processes that you requested. For instance, if you only request 1 slot on one machine, then please only use 1 processor core on that machine. It is quite possible that your qlogin session is shared with another job on the node and oversubscribing the node would have an adverse impact on someone else. If you need more slots, request them.
3. Please logout of your interactive session when you are done to free up the resources.
4. There is a default wall clock time limit of 24 hours. This can be overridden by specifying a value via `-l h_rt` but please make it reasonable.

It is possible to run X11 programs via qlogin. First, log in to the cluster with X11 forwarding enabled

```
ssh -Y argon.hpc.uiowa.edu
```

Then, launch your qlogin session. Once it is established then you can run programs that need X11. This is likely only to be usable from an on-campus connection for performance reasons.

If you use FastX then X11 forwarding is already in place.

It is important to remember that a qlogin job is just like any other job that is submitted via SGE. You can request resources in exactly the same way, including parallel environments with the appropriate number of slots. However, in the end, the qlogin session is an ssh session connection to one of the compute nodes. This means that the environment that the user is in at the shell prompt is a "fresh" environment. None of the special SGE variables that are used during a normal batch job are present in the qlogin environment. Probably the most important of these is the `$PE_HOSTFILE` environment variable that contains the list of hosts selected by SGE to be used for a parallel job. To be clear, the hostfile created by SGE is still created, but the environment variable that points to it is not present in the environment of the interactive session.

As such, the environment variables that are needed for a job need to be handled explicitly in an interactive session, just as one would do on the login host. For single slot interactive jobs, or even multi-slot, single node jobs, it is simply a matter of setting up the environment that you need for your session. However, qlogin requests that claim several hosts in a session and need to manage those hosts require a bit more work. As an example, say that an openmpi job is being debugged in a qlogin session. The qlogin session could be started as:

```
qlogin -pe orte 16
```

The ort PE is generally not recommended. It is used in this example to demonstrate what the host file can look like in its most complex form.

Something like the following will be echoed:

```
local configuration argon-login-1.local not defined - using global configuration
Your job 665450 (**QLOGIN**) has been submitted
waiting for interactive job to be scheduled ...
Your interactive job 665450 has been successfully scheduled.
Establishing builtin session to host compute-3-41.local ...
```

The two bits of information that are important in the output are the job number on line 2 and the masterq host on the last line. To find out the names of the hosts allocated by SGE in the above example, one could do:

```
cat /opt/gridengine/default/spool/compute-3-41/active_jobs/665450.1/pe_hostfile
```
These are the hosts, and the number of slots on each host, that have been allocated by SGE. It is up to you to make sure that any mpi job started in the `qlogin` session only uses these hosts, with the appropriate slot counts. In a batch job, `openmpi` detects the SGE environment and gets the total slots, hosts, and slots per host from the SGE environment. Since a `qlogin` session is not an SGE environment, this task must be done by the user. The above SGE hostfile must be converted to a format that `openmpi` can use. The following command will do this for `openmpi`:

```
cat /opt/gridengine/default/spool/compute-3-41/active_jobs/665450.1/pe_hostfile | awk '{print $1," slots=",$2}' > hostfile
```

This will produce the following hostfile:

```
compute-3-41.local slots=1
compute-2-29.local slots=1
compute-5-129.local slots=6
compute-3-56.local slots=6
compute-2-35.local slots=2
```

Also, while the environment can be passed to hosts via the `-V` flag to `qsub` in a batch job, that does not apply to `qlogin` sessions. Thus, you must make sure to get the necessary environment variables passed to every node that will be used in the computation. OpenMPI provides a mechanism to do this with the `-x <env>` flag.

To exit an interactive session, simply type `exit` at the shell prompt, just as you would with a normal ssh session.

**Note**

Note that if you happen to request resources that are unavailable, or to which you do not have access (such as an Investor queue to which you are not a member), you may see a message such as the following:

```
Your job 3747412 ("QLOGIN") has been submitted
waiting for interactive job to be scheduled ...timeout (5 s) expired while waiting on socket fd 4
.timeout (10 s) expired while waiting on socket fd 4
```

If you attempt to `qlogin` to a queue to which you do not have access, you may see the above message again, even when submitting your request to the proper queue. This is because the `qlogin` socket is left open for a short period of time after an unsuccessful request. The solution under both circumstances (lack of available nodes or improper queue request) is to simply wait a few minutes and then retry your request to the appropriate queue.

---

This would return something like:

```
compute-3-41.local 1 all.q@compute-3-41.local UNDEFINED
compute-2-29.local 1 all.q@compute-2-29.local UNDEFINED
compute-5-129.local 6 all.q@compute-5-129.local UNDEFINED
compute-3-56.local 6 all.q@compute-3-56.local UNDEFINED
compute-2-35.local 2 all.q@compute-2-35.local UNDEFINED
```