[Quick Install of Open MPI with Grid Engine](http://idolinux.blogspot.com/2010/04/quick-install-of-open-mpi-with-grid.html)

A prerequisite to this install log is a configured and validated Sun Grid Engine (SGE) installation. Read more about that at [Deploying Sun Grid Engine on a Cluster](http://idolinux.blogspot.com/2008/09/deploying-sun-grid-engine-on-cluster.html).

The Open MPI Project is an open source MPI-2 implementation that is developed and maintained by a consortium of academic, research, and industry partners. Open MPI is therefore able to combine the expertise, technologies, and resources from all across the High Performance Computing community in order to build the best MPI library available. -[www.open-mpi.org](http://www.open-mpi.org/)

Download and build the source rpm:

# cd /usr/global/src/  
# wget http://www.open-mpi.org/software/ompi/v1.4/downloads/openmpi-1.4.1-1.src.rpm  
# rpm -ihv openmpi-1.4.1-1.src.rpm  
# rpmbuild -bb --define 'configure\_options --with-sge --with-contrib-vt-flags=--disable-iotrace' /usr/src/redhat/SPECS/openmpi-1.4.1.spec

Note that building with the "--with-sge" option enables openmpi to be gridengine-aware. Also note that the option "--with-contrib-vt-flags=--disable-iotrace" fixes the following compiler error:

vt\_iowrap.c:1242: error: expected declaration specifiers or '...' before numeric constant  
 vt\_iowrap.c:1243: error: conflicting types for '\_\_fprintf\_chk'

Install the binary rpm on the head node and create a parallel execution environment template file.

# rpm -ihv /usr/src/redhat/RPMS/x86\_64/openmpi-1.4.1-1.x86\_64.rpm  
# vim ompi.template

Paste the following into the ompi.template file:

pe\_name ompi  
slots 64  
user\_lists NONE  
xuser\_lists NONE  
start\_proc\_args /bin/true  
stop\_proc\_args /bin/true  
allocation\_rule $fill\_up  
control\_slaves TRUE  
job\_is\_first\_task FALSE  
urgency\_slots min  
accounting\_summary FALSE

Add the parallel execution environment to grid engine:

# qconf -Ap ompi.template

Add the "ompi" parallel environment to the allowed pe\_list after "make", space separated:

# qconf -mq all.q  
# qconf -sq all.q | grep pe\_list  
 pe\_list make ompi

Copy the binary rpm to a shared NFS directory and install on all execution nodes. Here I use [dsh](http://www.netfort.gr.jp/~dancer/software/dsh.html.en) for concurrent ssh, but you could use a for loop instead.

# cp /usr/src/redhat/RPMS/x86\_64/openmpi-1.4.1-1.x86\_64.rpm /usr/global/src/  
# dsh -acM "rpm -i /usr/global/src/openmpi-1.4.1-1.x86\_64.rpm

And validate your setup by submitting a job with one command:

$ qrsh -V -q all.q -pe ompi 20 mpirun -np 20 hostname

Or submit a job interactively:

$ qlogin -q all.q -pe ompi 20  
nodeXX$ mpirun -np 20 ~/mpi\_hello

Or submit a job script:

$ qsub ~/mpi\_hello.sh  
$ \qstat -f

mpi\_hello.c example:

// a simple mpi test  
// compile with:  
// $ mpicc -o ~/mpi\_hello mpi\_hello.c  
#include <stdio.h>  
#include <mpi.h>  
int main (argc, argv)  
 int argc;  
 char \*argv[];  
{  
 int rank,size;  
 MPI\_Init(&argc,&argv); /\* starts MPI \*/  
 MPI\_Comm\_rank(MPI\_COMM\_WORLD,&rank); /\* get current process id \*/  
 MPI\_Comm\_size(MPI\_COMM\_WORLD,&size); /\* get number of processes \*/  
 printf("Hello world from process %d of %d\n",rank,size);  
 MPI\_Finalize();  
 return 0;  
}

mpi\_hello.sh example:

#!/bin/sh  
## a simple openmpi example  
## submit with:   
## $ qsub ~/mpi\_hello.sh  
# Export all environment variables  
#$ -V  
# Your job name  
#$ -N mpi\_hello  
# Use current working directory  
#$ -cwd  
# Join stdout and stderr  
#$ -j y  
# PARALLEL ENVIRONMENT:  
#$ -pe ompi 20  
# Enable resource reservation  
#$ -R y  
# The max hard walltime for this job is 16 minutes (after this it will be killed)  
#$ -l h\_rt=00:16:00  
# The max soft walltime for this job is 15 minute (after this SIGUSR2 will be sent)  
#$ -l s\_rt=00:15:00  
# The following is for reporting only. It is not really needed  
# to run the job. It will show up in your output file.  
echo "Got $NSLOTS processors."  
# The mpirun command.  
mpirun -np $NSLOTS hostname  
mpirun -np $NSLOTS ~/mpi\_hello

Run a job outside of gridengine.

$ ssh node01  
$ mpirun --machinefile /etc/machines.list -np 4 hostname  
node01  
node03  
node02  
node04  
$ mpirun --machinefile /etc/machines.list -np 4 ./test  
Hello World from Node 0  
Hello World from Node 1  
Hello World from Node 2  
Hello World from Node 3

More information: