# Using OpenMPI on the Gollum Midearth Cluster

This file explains how to use OpenMPI the gollum.midearth.cs.kent.edu. You will setup SSH keys which will allow password-less logins to all of the nodes, create a clusterhosts file, a sample MPI program as well as a brief example of running the program on nodes listed in the clusterhosts file.

- To setup your SSH keys, type the following at the command line:

```
[USERNAME@gollum ~]$ /usr/local/bin/ssh-keys-create
```

- Note: You will need to connect to every single node in the cluster once to accept it's keys.
  You would do this via ssh from the headnode (gollum.midearth.cs.kent.edu)

- To setup your environment, we will be using the modules package.
- To see which modules are available, type the following command:

```
[USERNAME@gollum ~]$ module avail
```

- You should get something similar to the following for output:

```
----------------------------------- /usr/share/Modules/modulefiles -------------------------------------
dot          module-git    module-info   modules      mpich2-x86_64 null          use.own

------------------------------------------ /etc/modulefiles --------------------------------------------
compat-openmpi-psm-x86_64 mvapich2-psm-x86_64     mvapich-psm-x86_64       openmpi-x86_64
compat-openmpi-x86_64     mvapich2-x86_64         mvapich-x86_64
```

- You will need to load the “openmpi-x86_64” module. To do so, you simply type the following command:

```
module load openmpi-x86_64
```

- This will set a number of important SHELL Environment variables. For example:

```
[[USERNAME@gollum ~]$ printenv |grep openmpi
MPI_INCLUDE=/usr/include/openmpi-x86_64
MPI_PYTHON_SITEARCH=/usr/lib64/python2.6/site-packages/openmpi
LD_LIBRARY_PATH=/usr/lib64/openmpi/lib
MPI_LIB=/usr/lib64/openmpi/lib
PATH=/usr/lib64/openmpi/bin:/usr/lib64/qt-
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/USERNAME/bin
MPI_BIN=/usr/lib64/openmpi/bin
MPI_COMPILER=openmpi-x86_64
_LMFILES_=/etc/modulefiles/openmpi-x86_64
LOADEDMODULES=openmpi-x86_64
MPI_SYSCONFIG=/etc/openmpi-x86_64
MPI_SUFFIX=_openmpi
MPI_MAN=/usr/share/man/openmpi-x86_64
PYTHONPATH=/usr/lib64/python2.6/site-packages/openmpi
MPI_HOME=/usr/lib64/openmpi
MPI_FORTRAN_MOD_DIR=/usr/lib64/gfortran/modules/openmpi-x86_64
```

- To force the “module” command to always load these environment variables when you log in, you need to tell it to update your login scripts. To do so, type the following:

```
[USERNAME@gollum ~]$ echo "module add " >> ~/.bashrc
[USERNAME@gollum ~]$ module initadd openmpi-x86_64
```

- When all of the nodes are working, there should be 12 total. They have both a normal ethernet address and an infiniband address. You should only use the head node for editing and compiling your programs.

- After setting up the Environment Modules, we need to create a host file. For example, you could define a “clusterhosts” and an “infinihosts” file that contains the lines:

```
clusterhosts:     node1
                  ...
                  node12

infinihosts:      inode1
                  ...
                  inode12
```

- To do so, type the following at the command line:

```
[USERNAME@gollum ~]$ for i in {1..12}; do echo "node$i" >>clusterhosts; done
[USERNAME@gollum ~]$ cat clusterhosts
node1
node2
node3
node4
node5
node6
node7
node8
node9
node10
node11
node12
```

- To create a “infinihosts” file with the infiniband addresses, you would type the following:

```
[USERNAME@gollum ~]$ for i in {1..12}; do echo "inode$i" >>infinihosts; done
[USERNAME@gollum ~]$ cat infinihosts
inode1
inode2
inode3
inode4
inode5
inode6
inode7
inode8
inode9
inode10
inode11
inode12
```

- These host files will list the cluster nodes that we are going to run our program(s) on.

- Create a file containing the following simple “Hello World” program, written in C. Here this file is named demo.c

```
#include <stdio.h>
#include "mpi.h"

int main(int argc, char **argv)
{
    int myid;           /* my rank  */
    int numprocs;       /* the cluster size */

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);

    printf("Hello World from %d of %d\n", myid, numprocs);

    MPI_Finalize();
    return 0;
}
```

- You would then compile the program “demo.c” by typing the following:

```
[USERNAME@gollum ~]$ mpicc -o demo demo.c
```

- This should create an executable file in your directory named “demo”.

```
[USERNAME@gollum ~]$ ls -al demo
-rwxrwxr-x. 1 USERNAME USERNAME 7472 Oct 21 15:21 demo
```

- You can run the executable file on the current node with either of the following commands:

```
[USERNAME@gollum ~]$ mpirun demo
[USERNAME@gollum ~]$ mpirun -n 1 demo
```

- This should produce a response similar to the following:

```
Hello World from 0 of 1
```

- The option “-n” is the number of process to run

- The program could be run on the three nodes using the command:

```
[USERNAME@gollum ~]$ mpirun -n 3 --hostfile clusterhosts demo
```

- This should produce a response similar to the following:

```
Hello World from 0 of 3
Hello World from 1 of 3
Hello World from 2 of 3
```

- The option “--hostfile” is the name of the file with the list of nodes you want to run the program on

- Finally, you can run this executable file on all of the nodes in your “clusterhosts” file as follows:

```
[USERNAME@gollum ~]$ mpirun --hostfile clusterhosts demo
```

- You should receive a response similar to the following:

```
Hello World from 1 of 12
Hello World from 2 of 12
Hello World from 3 of 12
Hello World from 4 of 12
Hello World from 5 of 12
Hello World from 6 of 12
Hello World from 7 of 12
Hello World from 8 of 12
Hello World from 9 of 12
Hello World from 10 of 12
Hello World from 11 of 12
Hello World from 11 of 12
```

- The machines may in fact answer out of order which is normal.

- By not specify the number of processes to run, it will start 1 process per host listed in your file “clusterhosts”. Note: You can do the same thing via the Infiniband interfaces by specifying “infinihosts” as the hostsfile instead.

- If you want to run a large number of process you would just specify a larger number on the command line. For example, running 24 copies of the program would be done by typing the following:

```
[USERNAME@gollum ~]$ mpirun -n 24 --hostfile clusterhosts  demo
Hello World from 0 of 24
Hello World from 1 of 24
Hello World from 2 of 24
Hello World from 3 of 24
Hello World from 4 of 24
Hello World from 5 of 24
Hello World from 6 of 24
Hello World from 7 of 24
Hello World from 8 of 24
Hello World from 9 of 24
Hello World from 10 of 24
Hello World from 11 of 24
Hello World from 12 of 24
Hello World from 13 of 24
Hello World from 14 of 24
Hello World from 15 of 24
Hello World from 16 of 24
Hello World from 17 of 24
Hello World from 18 of 24
Hello World from 19 of 24
Hello World from 20 of 24
Hello World from 21 of 24
Hello World from 22 of 24
Hello World from 23 of 24
```