# Intel® Manycore Testing Lab (MTL) - Linux Getting Started Guide

## *Introduction*

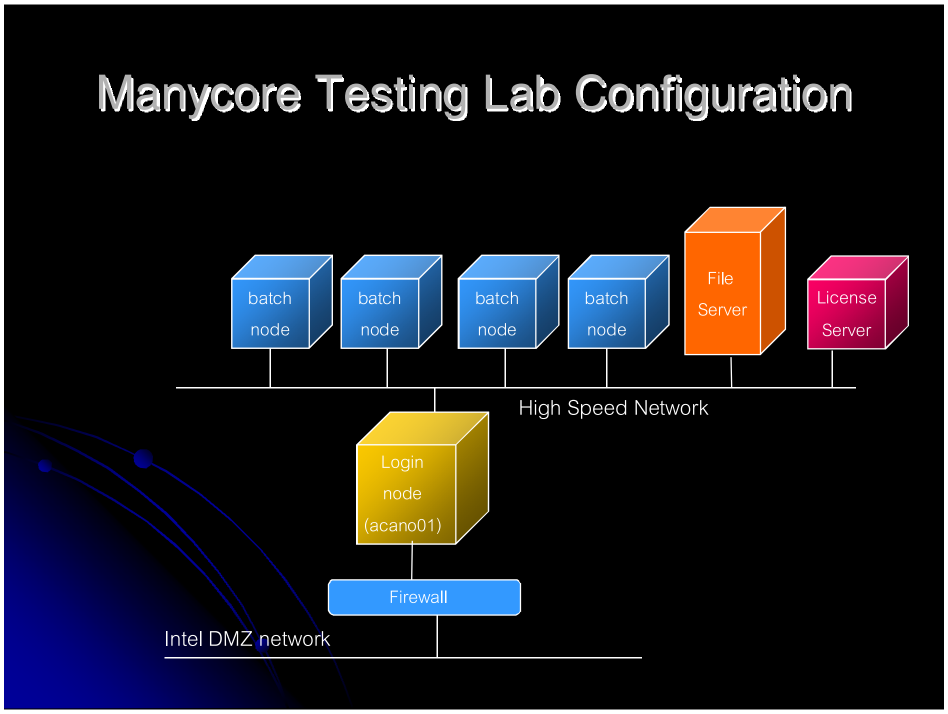
### What are the intended uses of the MTL?

The MTL is prioritized for supporting the Intel Academic Community for the testing, validation and scaling of parallel algorithms and workloads, primarily for courseware delivery, and secondly for research - based on availability.

The MTL supports a default shared login node (many users), for workload development and a limited number of batch nodes for benchmarking. The nodes are located in DuPont, Washington, USA, and are directly connected to the Internet via dedicated firewall devices.

The MTL is not configured or setup as a cluster, so MPI is not a supported option for our community. The current system resources therefore do not support a distributed memory programming model.

### What is the configuration of the MTL?



**Note:** The batch nodes, don't necessary have the same memory and CPU configurations as the login node. Look for system configuration update announcements on the MTL forum: http://software.intel.com/en-us/forums/intel-manycore-testing-lab/

## How do I get an account?

Use of the MTL is a benefit for members of the Intel Academic Community, available free of charge, for the asking. However, users must request an account from the Intel Academic Community; http://software.intel.com/en-us/academic/

## How do I connect?

Once an account is available, use **SSH** to connect to one of the following IP addresses using your favorite terminal connection software, e.g. ssh, Putty, F-Secure, etc:

| **IP-address** | **System name** | **Connection Model** |
|---|---|---|
| 36.81. 203.1 | acano01 | VPN |
| 192.55.51.81 | acano01 | Direct IP |

**Note1:** If you're using a VPN client to connect to the MTL, you must first invoke the VPN client to establish the connection to the Host (named above), before using the appropriate terminal connection model.

**Note2:** Ping is not configured to pass through the MTL firewall without a VPN tunnel, so all ping requests to acano01, will be rejected.

## Where can I get a copy of a VPN client to access MTL?

Cisco® owns the rights for the VPN client – it can be obtained from this site:
http://www.supreme.state.az.us/downloads/VPN/
Download and install the Cisco VPN client either the 32-bit or 64-bit client as appropriate.
**Note:** These only work with Windows.

Within the VPN client please specify the following (under Group Authentication):

Host: **192.55.51.80**

Name: <VPN_GROUP_NAME>

Password: <VPN_PASSWORD>

Connection Entry & Description (not significant)

**Note:** when the VPN tunnel is connected you will not be able to use your machine to connect to anything else, i.e. the web or other systems.

## *Intermediate*

### What tools are available on the MTL?

The login node as well as the rest of the MTL is built upon Linux. The current distribution in use is Redhat Enterprise Linux (RHEL 5.4, kernel 2.6.18-194.11.4.el5) but that particular distribution is subject to change.

A reasonable set of login shells are available including the default, bash, as well as tcsh and zsh. In addition most of the typical Linux command line tools such as vim, emacs, gcc, make, python, perl, mc, etc. are available.

VNC(GUI) and screen(tty) interfaces are available for multiplexing your ssh connection as well as preserving the state of user logins. VNC will need to be routed through the primary ssh connection (e.g. Putty) using port forwarding. Please see a relevant MTL forum posting for more details how set up an X connection: http://software.intel.com/en-us/forums/showthread.php?t=76570&o=a&s=lr

or search the MTL forum: http://software.intel.com/en-us/forums/intel-manycore-testing-lab/

Man pages are available for most commands and are a good starting point when you have questions.

### What resources are available on the MTL?

A PBSPro 10.2 batch system, is available and required for batch job submission. For the latest PBSPro v10.2 commands, please. read:

/opt/docs/PBSProUserGuide10.2.pdf

The /home directories are NFS-mounted from a standard storage server with a total capacity of over 1TB.

Intel performance tools: Intel® Parallel Studio XE which includes:, Intel® Composer XE, Intel® VTune Amplifier XE and Intel® Inspector XE are available.

**Note:** If you wish to use the Intel® VTune Amplifier XE; you must have permission to write to the driver in order to proceed. Also, you'll need to run an X-terminal to use the VTune GUI.

To use the Intel® VTune Amplifier XE:
First set your environment variables using:

```
$ source /opt/intel/vtune_amplifier_xe_2011/amplxe-vars.sh
```

To start the graphical user interface:

```
$ amplxe-gui
```

To use the command-line interface:

```
$  amplxe-cl
```

To view a table of getting started documents:

```
/opt/intel/vtune_amplifier_xe_2011/documentation/en/documentation_amplifier.htm
```

To use the Intel® Inspector XE:
First set your environment variables using:

```
$ source  /opt/intel/inspector_xe_2011/inspxe-vars.sh
```

To start the graphical user interface:

```
$ inspxe-gui
```

To use the command-line interface:

```
$  inspxe-cl
```

To view a table of getting started documents:

```
/opt/intel/inspector_xe_2011/documentation/en/documentation_inspector_xe.htm
```

Additional system wide tools will be made available, so look for announcements on the MTL forum: http://software.intel.com/en-us/forums/intel-manycore-testing-lab/

## How do I compile my code using a specific compiler or library?

gcc/g++ (v4.1.2) is the default compiler

To compile with the updated version (4.5.1) of the gcc compiler suite (that supports OpenMP v3.0), use the following commands in your makefile:

```
GCC_VERSION = 4.5.1
PREFIX = /opt/gcc/${GCC_VERSION}/bin
CC = ${PREFIX}/gcc
CPP = $(PREFIC}/g++
LD_LIBRARY_PATH = /opt/mpfr/lib:/opt/gmp/lib:/opt/mpc/lib
```

It may be necessary to export the following, to get an application to compile.

```
$ export LD_LIBRARY_PATH =/opt/mpfr/lib:/opt/gmp/lib:/opt/mpc/lib
```

**Note:** To use the following Intel tools it is necessary to replace "intel64" with "ia32" when executing these source commands if you are using a 32-bit platform.

To use the Intel® Composer XE package:

Set the environment variables for a terminal window using the following:

```
$ source /opt/intel/bin/compilervars.sh  intel64
```

To invoke the installed compiler:

For C: `icc`
For C++: `icpc`
For Fortran: `ifort`

To use the TBB library, execute the following source command:

```
$ source /opt/intel/composerxe/tbb/bin/tbbvars.sh  intel64
```

To compile with the stand-alone Intel compiler, execute the following source command:

```
$ source /opt/intel/Compiler/latest/bin/iccvars.sh   intel64
```

To compile with the stand-alone Intel Fortran compiler, execute the following source command:

```
$ source /opt/intel/Compiler/latest/bin/ifortvars.sh   intel64
```

**Note:** older possibly supported versions of these tools are located in:

```
/opt/intel/Compiler/<version>/<release>
```

## How do I run a program in an exclusive mode (say for benchmarking)?

Qsub is the command used to submit jobs to the batch system. Using a script to "wrap" the actual program or programs is the most effective way to submit jobs. Here is a very simple script that merely prints a series of "hello world" when submitted to PBS using the following qsub command:

```
$ qsub –l select=1:ncpus=<xx> $HOME/myjob
```

**Note:** the value of ncpus should be aligned typically to the number of threads that your application plans to use, if unknown or greater then **40**, then use the value of **40**. The reason for specifying the ncpus argument to qsub, is to better utilize the batch nodes, in that if less than **40** cores are in use on a specific batch node, the remaining free cores could be available for other jobs/users.

If the argument above is not used, qsub will return with the following error:

```
$ qsub: Rejecting job, Pls. use syntax -l select=1:ncpus=xx ..
```

The contents of the 5 line myjob (for OpenMP) could be:

```
#!/bin/sh
#PBS -N myjob
#PBS -j oe
export OMP_NUM_THREADS=16
./hello_world
```

The output of the batch job will be left in a file in the working directory where the qsub command was entered. The file will be named using the PBS job name (e.g. myjob) followed by a suffix built using .o + the job number, e.g. myjob.o28802.

For latest PBSPro v10.2 qsub commands, please. read:
/opt/docs/PBSProUserGuide10.2.pdf - example subsection

**Note:** Because of the shared environment on the MTL login node (acano01), it's not recommended that repeatable testing (and results) be performed on this node, the batch system it set up for this purpose.

## How do I monitor the programs launched?

The qstat utility will output the status of the jobs submitted to PBS. See the man page for the various formatting options. Here is an example output:

```
$ qstat -a

acano01:
                                                            Req'd    Req'd       Elap
Job ID          Username  Queue    Jobname    SessID  NDS  TSK  Memory   Time   S  Time
--------------- --------  -------- ---------- ------  ---  ---  ------   -----  -  ---------
323.acano01      user01   workq    myjob      19290    1    32    --     01:00  R  00:03
324.acano01      user01   workq    myjob      19659    1    32    --     01:00  R  00:02
325.acano01      user01   workq    myjob         --    1    32    --     01:00  Q    --
326.acano01      user01   workq    myjob         --    1    32    --     01:00  Q    --
327.acano01      user01   workq    myjob         --    1    32    --     01:00  Q    --
```

## How do I abort a program after launch?

The qdel command can be used to abort a job submitted with qsub. The job ID is the only parameter required:

```
$ qdel <job_id>
```

**Note:** If you kill a job (using qdel), the PBS exit status will show 271.

## *Advanced*

### How can a program be run interactively?

Use the –I option to qsub and you will be dropped into a shell on the first compute node:

```
$ qsub –I
```

**Notes:**

Using interactive mode is not necessarily a good citizen on the MTL, given the limited number of batch nodes available and the option of specifying long batch run duration.

So as not to block other users from running (scripted) batch jobs, a hard walltime limit (see below), has been imposed for **all** interactive jobs. This hard limit is currently set to two hours, i.e. 2:00:00. Any interactive job submitted with a walltime over this hard-limit, will automatically fail to execute and generate the message:

```
qsub: Job exceeds queue and/or server resource limits
```

It is strongly suggested that interactive batch runs only be submitted when PBS scripting **does not work** as it is very easy to tie up an idle batch node, while waiting for interactive shell input.

### How is batch run duration specified?

```
$ qsub -l  "walltime=0:30:00" ./my_script
```

Amount of Wall-clock time during which is the job can run, it establishes a job resource limit.

**Notes:**

The current default is set to ten (10) minutes, if walltime is not defined.
A PBS exit status of **271** from: `tracejob job_id`, may indicate that the job exceeded the wall-clock time.

### How can a run-away or unresponsive program be stopped?

Try using qsig command.

```
$ qsig –s 9 job_id
```

## How can I determine if my batch job will possibly run immediately?

There are a number of batch nodes on the MTL, but they may be busy running other jobs. You can run the `qfree` script to determine if there are any batch nodes free at this time. If not then you may have to wait for your batch job to be scheduled once you submit it with `qsub`. This `qfree` script will not reserve a batch node for your use, it just reports the current status of all the available MTL batch nodes. Here is an example output:

```
$ qfree

          Number of MTL Batch nodes free: 2

          Number of MTL Batch nodes busy: 1
```

## How can I specify which CPUs my application should use?

Use the taskset command, e.g.

```
$ taskset –c 0-15 <application>
```

This will specify that the application should run only on the first 16 CPUs, i.e. CPU affinity.

```
$ taskset –c 0,32 <application>
```

This will specify (on a 64 thread SMT system), that the application should run on both the 1st physical and the 1st logical core.

See: man taskset for more details

## How can I request a number of CPUs my job should use?

To run batch jobs with a requested number of CPUs or cores you should include the following arguments to your qsub command:

```
qsub -l select=1:ncpus=<xx> <job_name>
```

Replace **xx** with the number of CPUs that you want to request. Currently the maximum value of **xx** can be set to **40**. If you specify a number greater than **40**, your job will be queued indefinitely as this number of CPUs is not available.

**Note: ncpus** is a number of processors requested, it does not limit your job to an equal number of threads. Likewise the OpenMP directive OMP_NUM_THREADS operates independently to the **ncpus=xx qsub** argument as in:

```
export OMP_NUM_THREADS=64
```

## How can I specify which MTL node my application should use?

You may have a software license that is HOSTID locked and you want to only run on a specific batch node, understanding that if that node is already busy with another job, that your job will be queued until that host resource becomes available.

To run jobs on a specific batch node you should include the following arguments to your qsub command:

```
qsub -l select=1:host=<acano0x> <job_name>
```

Replace **x** with the batch node number that you want to test with, currently 2, 3 or 4.

### How can I transfer files to the MTL and from the Internet?

You can always initiate a scp (winscp) session to MTL from your own system and use this to upload and download files to and from your local login system. However, due to firewall restrictions, this will only work from the login node (acano01). This login node (acano01) is also restricted in its ability to directly access the internet for security reasons.

It's not recommended to transfer large files or data sets (GBs), using scp – they should be split into smaller chunks and submitted in parallel.

### How do I backup my code and/or data sets?

The MTL does not support any kind of backup of user data. It is suggested that users keep a local copy of their data/code, to mitigate any potential loss of data from the MTL.

### Why when I run my workload, do I get resources temporarily unavailable or unable to create thread?

This is possibly due to exceeding the number of processes/threads that are allocated on a per user basis.

### Why are my processes getting niced on the login node?

The login node (acano01) is provided as a compilation platform and test environment, so that users have a chance to debug their programs in an environment as close as possible to what's on the batch nodes. However, some users have been running jobs on there that are using up a lot of CPU. Because of this, an automatic re-nicer has been set up. For every 60 seconds of 100% CPU time a process uses, it will be niced one level – ideling for 60 seconds gives up one niced level. Thus, after about 20 CPU minutes, that process will have the lowest priority. **This means that any code executed on the login node (acano01), may results in the duration timings becoming indeterminate, based on it being executed at a lower priority.** This should minimize the impact of these processes on other users.