A UT/ORNL PARTNERSHIP
NATIONAL INSTITUTE FOR COMPUTATIONAL SCIENCES
NICS

Introduction to Kraken

Daniel Lucio
User Support

May 10th 2010, ORNL, TN

HowTo

1. Log into Kraken
2. Compile/Link Applications
3. Run jobs
4. Debug Applications
5. Transfer Files
6. Archive Files
7. Obtain Accounts
8. Find other Resources for users

HowTo login

Via SSH, BBCP using OTP

```
% ssh userid@kraken.nics.tennessee.edu
Enter PASSCODE:
```

PASSCODE = PIN + TokenCode

4 digit 6 digit

Changes every 30s

Via GSISSH (*Teragrid portal*)

```
login3$ myproxy-logon
Enter MyProxy pass phrase:
A credential has been received for user userid in /tmp/x509up_u974.
login3$ gsissh kraken-gsi.nics.tennessee.edu
userid@kraken-pwd4(XT5):~>
```

Via GridFTP, UBERFTP

```
gsiftp://gridftp.nics.utk.edu:2811
```

HowTo login

Via the TeraGrid Portal
(Java SSH client)

HowTo use modules

- All software/packages are managed via modules
- This allows environment variables, libraries, include paths to be cleanly entered and/or removed from your software environment.
- Conflicts are detected and loads that would cause conflicts are not allowed
- There are a number of basic modules loaded by default

```
  1) modules/3.1.6.5
  2) torque/2.4.1b1
  3) moab/5.2.5.s12399
  4) /opt/cray/xt-asyncpe/default/modulefiles/xtpe-
istanbul
  5) tgusage/5.0-r2
  6) altd/1.0
  7) DefApps
  8) cray/MySQL/5.0.64-1.0000.2342.16.1
  9) xtpe-target-cnl
 10) xt-service/2.2.41A
 11) xt-os/2.2.41A
 12) xt-boot/2.2.41A
 13) xt-lustre-ss/2.2.41A_1.6.5
 14) cray/job/1.5.5-0.1_2.0202.18632.46.1
 15) cray/csa/3.0.0-1_2.0202.18623.63.1
 16) cray/account/1.0.0-2.0202.18612.42.3
 17) cray/projdb/1.0.0-1.0202.18638.45.1
 18) Base-opts/2.2.41A
 19) pgi/9.0.4
 20) xt-libsci/10.4.1
 21) pmi/1.0-1.0000.7628.10.2.ss
 22) xt-mpt/4.0.1
 23) xt-pe/2.2.41A
 24) xt-asyncpe/3.6
 25) PrgEnv-pgi/2.2.41A
```

HowTo use modules

The complete list of all available modules can be viewed with the command `module avail`. The 3rd party list of software can also be viewed from our website at:

http://www.nics.tennessee.edu/user-support/software/Kraken

Loading commands

`module [load|unload] <my_module>`
Loads/unloads module

`module swap <module1> <module2>`
Replaces <module1> with <module2>

```
> module swap PrgEnv-pgi PrgEnv-gnu
```

Informational commands

`module help [my_module]`
Lists available commands and usage

`module show <my_module>`
Displays the actions upon loading the module <my_module>

`module list`
Displays all currently loaded modules

`module avail <name>`
Lists all modules (beginning with name)

HowTo compile

- Available C, C++ and Fortran compilers: PGI, GNU, Pathscale
- Use the Cray compiler wrappers `cc`, `CC` and `ftn`, to compile programs for the compute nodes.
- The compiler wrappers know where most of the correct Cray provided libraries and include files are, if the corresponding module is loaded.
- You do not need to know where the MPI libraries are.
- The wrappers automatically add the correct tuning parameters for the Istanbul Processor.
- Use module help <name> to learn what you need to manually add for 3rd party modules

HowTo compile

This example shows that a user needs to add ${SUPER_LU} to the compile line

```
lucio@krakenpf2(XT5):~> module help superlu
----------- Module Specific Help for 'superlu/4.0' ----------------
Sets up environment to use parallel SUPERLU 4.0.
Usage:   ftn test.f90 ${SUPERLU_LIB}
    or   cc test.c ${SUPERLU_LIB}
```

Example of what the wrappers do for you

```
/sw/altd/bin/ld /usr/lib64/crt1.o /usr/lib64/crti.o /opt/pgi/9.0.3/linux86-64/9.0-3/lib/trace_init.o /
usr/lib64/gcc/x86_64-suse-linux/4.1.2/crtbeginT.o -m elf_x86_64 -dynamic-linker /lib64/ld-linux-
x86-64.so.2 /opt/pgi/9.0.3/linux86-64/9.0-3/lib/pgi.ld -L/opt/mpt/3.5.0/xt/mpich2-pgi/lib -L/opt/xt-
libsci/10.3.9/pgi/lib -L/opt/mpt/3.5.0/xt/sma/lib -L/opt/mpt/3.5.0/xt/util/lib -L/opt/mpt/3.5.0/xt/pmi/
lib -L/opt/xt-pe/2.2.41A/lib -L/opt/xt-pe/2.2.41A/lib/snos64 -L/usr/lib/alps -L/opt/pgi/9.0.3/
linux86-64/9.0-3/lib -L/usr/lib64 -L/usr/lib64/gcc/x86_64-suse-linux/4.1.2 -Bstatic -rpath=/opt/xt-pe/
2.2.41A/lib -rpath=/opt/mpt/3.5.0/xt/mpich2-pgi/lib -rpath=/opt/xt-libsci/10.3.9/pgi/lib -rpath=/opt/
mpt/3.5.0/xt/sma/lib -rpath=/opt/mpt/3.5.0/xt/util/lib -rpath=/opt/mpt/3.5.0/xt/pmi/lib -lsci_istanbul
-lsma -lmpich -lrt --start-group -lpmi -lalpslli -lalpsutil -lportals -lpthread -lm --end-group -
rpath /opt/pgi/9.0.3/linux86-64/9.0-3/lib -lpgf90 -lpgf90_rpm1 -lpgf902 -lpgf90rtl -lpgftnrtl -lnspgc -
lpgc -lrt -lpthread -lm -lgcc -lgcc_eh -lc -lgcc -lgcc_eh -lc /usr/lib64/gcc/x86_64-suse-linux/4.1.2/
crtend.o /usr/lib64/crtn.o
```

How to compile

MPI Hello World example

```
/* C Example */
#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( "Hello2 world2 from process %d of %d\n", rank, size );

  MPI_Finalize();

  return 0;
}
```

```
> cc -o hello hello.c
```

How to compile

MPI Hello World example with another compiler

```
> module swap PrgEnv-pgi PrgEnv-gnu
> cc –o hello hello.c
```

MPI Hello World example with an older compiler version

```
> module swap pgi/9.0.3 pgi/7.2.5
> module swap xtpe-istanbul xtpe-barcelona
> cc –o hello hello.c
```

What compiler flags to use?

`-fast –Mipa=fast` `-Minfo=all –Mneginfo=all`

How to compile

Using 3rd party hdf5/1.6.7 library example

```
> module load hdf5/1.6.7
> module help hdf5/1.6.7

----------- Module Specific Help for 'hdf5/1.6.7' -----------------

Sets up environment to use serial HDF5 1.6.7 with any compiler.
Usage:   ftn test.f90 ${HDF5_FLIB}  OR  h5fc test.f90
    or   cc test.c ${HDF5_CLIB}     OR  h5cc test.c
The hdf5 module must be reloaded if you change the PrgEnv
   or you must issue a 'module update hdf5' command.
This version is deprecated and will soon be no longer available.

> cc -o myhdf5test  h5_copy18.c ${HDF5_CLIB}
```

HowTo run a job

Remember that the compute nodes can only access the Lustre scratch file system. Therefore all input/output files for your program must be within Lustre.
Non interactive jobs are launched with a batch job script with the help of the 'qsub' command.

Job script example

```
#PBS -A UT-NTNLEDU
#PBS -l size=12
#PBS -l walltime=00:05:00

cd $PBS_O_WORKDIR
aprun -n 4 ./hello
```

1. *Specify project account*
2. *Specify number of cores to allocate for this job. It must always be a multiple of 12*
3. *$PBS_O_WORKDIR is set to the directory from where you issued the qsub command*

HowTo run a job

Before submitting your job, make sure your current directory is somewhere in Lustre. Here is an example when it is not:

Error from submitting a job from home directory

```
lucio@krakenpf11(XT5):~> qsub /lustre/scratch/lucio/helloMPI/ hello_mpi.pbs
384515.nid00016
lucio@krakenpf11(XT5):~> ls hello_mpi.pbs*
-rw------- 1 lucio nicsstaff 104 2009-12-07 09:14 hello_mpi.pbs.e384515
-rw------- 1 lucio nicsstaff   0 2009-12-07 09:14 hello_mpi.pbs.o384515
lucio@krakenpf11(XT5):~> cat hello_mpi.pbs.e384515
[NID 16327] 2009-12-07 09:14:41 Exec ./hello failed: chdir /nics/a/home/
lucio No such file or directory
```

HowTo run a job

Hello world example

```
lucio@krakenpf3(XT5):~> cd /lustre/scratch/lucio/helloMPI
lucio@krakenpf3(XT5):/lustre/scratch/lucio/helloMPI> qsub hello_mpi.pbs
384361.nid00016
lucio@krakenpf3(XT5):/lustre/scratch/lucio/helloMPI> qstat 384361
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
384361.nid00016           hello_mpi.pbs    lucio           00:00:00 C small
lucio@krakenpf3(XT5):/lustre/scratch/lucio/helloMPI> ls
hello  hello_mpi.c  hello_mpi.pbs  hello_mpi.pbs.e384361  hello_mpi.pbs.o384361
lucio@krakenpf3(XT5):/lustre/scratch/lucio/helloMPI> cat hello_mpi.pbs.o384361
Hello world from process 1 of 4
Hello world from process 2 of 4
Hello world from process 0 of 4
Hello world from process 3 of 4
Application 1529031 resources utime 0, stime
```

HowTo run a job

Using a 3rd party application like NAMD

```
#!/bin/bash
#PBS -A TG-DMR090083
#PBS -j oe
#PBS -m abe
#PBS -N
#PBS -l walltime=3:00:00,size=144

module load namd/2.71-09Jul21

cd $PBS_O_WORKDIR

export MPICH_PTL_SEND_CREDITS=-1
export MPICH_MAX_SHORT_MSG_SIZE=8000
export MPICH_PTL_UNEX_EVENTS=80000
export MPICH_UNEX_BUFFER_SIZE=100M

aprun -n 144 namd2 start.namd 2>&1 > start .log
```

HowTo run an Interactive job

Sprng example

```
$ cd /lustre/scratch/lucio/sprngExample
$ showusage
Usage on kraken

   Project     StartDate  EndDate     Allocation     Remaining      Usage
-------------+----------+----------+------------+------------+------------
TG-STA080002N 2009-06-01 2010-06-01    200000.00     22827.92    177172.08
    UT-ACCEPT 2008-09-01 2010-09-30   7000000.00   1259246.04   5740753.96
   UT-SUPPORT 2009-03-01 2010-09-30   3000000.00    924565.75   2075434.25

NOTE - For more detailed job charges, use 'glsjob -u {username}
$ qsub -I -A UT-SUPPORT -lwalltime=00:05:00,size=12
qsub: waiting for job 504380.nid00016 to start
qsub: job 504380.nid00016 ready
$
```

HowTo run an Interactive job

Sprng example (cont)

```
$ pwd
/nics/a/home/lucio
$ cd /lustre/scratch/lucio/sprngExample/
$ aprun -n 8 ./sprng_mpi
Evailable generators; use corresponding numeral:
   lfg     --- 0
   lcg     --- 1
   lcg64     --- 2
   cmrg     --- 3
   mlfg     --- 4
   pmlcg     --- 5
3
Type in a generator type (integers: 0,1,2,3,4,5):
Process 7, random number 3: 0.01232133043670
Process 0, print information about stream:
Application 2041147 resources: utime 0, stime 0
```

HowTo debug and profile

The following tools are available on Kraken for debugging, profiling and analysis parallel programs

Debugging	Profiling and Analysis
Totalview, lgdb, atp, gdb, pgdbg	Cray PAT, pgprof, TAU, FPMPI, PAPI, Scalasca

Always check the compatibility of the compiler options you want to use. For example, the following PGI compiler options are not supported:

`-Mprof=mpi, -Mmpi, and -Mscalapack`

HowTo monitor memory

The following tools are available on Kraken for monitoring memory usage

•Reading the 'Vm' entries from '/proc/<pid>/status/' for your Unix process. NICS has developed an API to do this, source code and documentation is located at:

/nics/b/proj/nicsstaff/memory

•FPMPI, light-weight profiling libraries

http://www.nics.tennessee.edu/user-support/software/Kraken?&software=fpmpi

•`/usr/bin/free`, will only return the amount of memory that it has not beentouched yet.

HowTo transfer & archive

There are five ways to transfer files to/from Kraken:

globus-url-copy (GridFTP)	The fastest way to transfer files to/from other (TG) systems. Can only be used with Kraken's GridFTP nodes. Requires GSI authentication. Use `gsiftp://gridftp.nics.utk.edu:2811`
Uberftp	Convenient FTP/SFTP like client, that uses the GridFTP protocol. Same requirements as 'globus-url-copy'.
BBCP	Yields much better performance than standard scp. Recommended if GridFTP is not available. *The right transfer parameters are critical for getting the best transfer rates!*
SCP (HPN)	A modified version of `scp` that uses dynamic flow control buffers which yields better transfer rates than the vanilla version that comes with OpenSSH. Available at all OTP/GSI nodes.
HSI/HTAR	Used to archive files and extract to/from the mass storage HPSS system. Only available at the OTP login nodes. *Highly desirable to bundle files together when archiving files!*

HowTo do offline archiving

The scheduler will assign your job to the right queue based upon the number of cores and walltime allocated. Do not specify a queue (except for jobs that archive files).

HPSS batch script example

```
#!/bin/bash
#PBS -A TG-EXAMPLE
#PBS -l size=0
#PBS -l walltime=10:00:00
#PBS -q hpss
#PBS -W depend=afterok:123456.nid00016

cd $PBS_O_WORKDIR
hsi put file
htar cvf this_run.tar dir/
```

Batch jobs that use the `hpss` queue must request zero cores

HowTo to get an account

You can apply for an account on Kraken in two ways:

-Via a Teragrid allocation

(Four times a year proposals are reviewed)

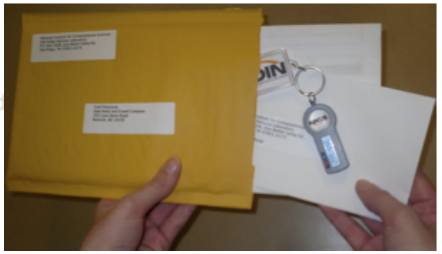
http://www.teragrid.org/userinfo/access/dac.php

- Via a Director's Discretionary allocation

(Only for courses and workshops)

http://www.nics.tennessee.edu/user-support/request-an-account

You will receive from us a welcome package, that includes both your token and login instructions.



What do you get with your account

- A Unix account (userid and Project account)
- A One Time Password generator (token) to login via ssh
- Access through Globus Grid tools (gssissh, GridFTP)
- A NFS home area (default 2GB quota)
- Lustre scratch space (<2.4PT)
- HPSS mass storage archival via `hsi/htar` (OTP only)
- >100 applications ready to run on Kraken
- Up to 99,072 cores
- User Assistance
- Bash as default Unix shell

More information

Cray offers most of their documentation online at

http://docs.cray.com/

Two excellent documents for new users are:

- Cray XT System Overview (S-2423-22)
- **Cray XT Programming Environment User's Guide** (S-2396-22)
- Using Cray Performance Analysis Tools (S-2376-50)

Kraken's User Guide developed by:

http://www.nics.tennessee.edu/computing-resources/kraken/user-guide

Cray XT5 workshop slides

http://www.nersc.gov/projects/workshops/CrayXT/agenda.php

http://www.nccs.gov/user-support/training-education/workshop-archives/fall-2009-cray-xt5-hex-core-workshop/

How to get help

Send your questions via email to

help@teragrid.org

Or contact us by phone

1.865.241.1504

or
to the TG helpdesk
1.866.907.2383 (off hours)

NICS survival kit

This is a list of Unix commands available on Kraken that all users should be aware of:

`module <command>`	All software packages like compilers, libraries and applications, are handled via modules
`qsub <jobscript>`	All jobs are submitted with the `qsub` command
`aprun -{n|N|S|d|cc}`	Programs get executed on the compute nodes with this command
`showq [-r]`	Shows the state of the queue

NICS survival kit

`qstat <jid>`	Shows the status of a job with job id `<jid>`
`glsjob <jid>`	It can be used to query information about a previous job or all previous jobs with `-u <uid>`
`showstart <jid>`	Shows approximate start time for job with job id `<jid>`
`showusage`	Displays the current balance of all the project accounts on Kraken a user has access to
`showbf`	Shows what resources are available for "immediate" use.

Other commands include: `checkjob, apstat, glsuser, glsproject`

HowTo use Lustre

Interesting facts:

- Kraken has 1.73PB of data with ~300M files
- Jaguar has 1.43PB of data with ~201M files

Configuration:

- 2.4PB of total space
- 48 OSS servers
- 7 OST per OSS (336 OSTs total)
- Peak sustained bandwidth: ~30GB/s
- Defaults: Stripe count 4, Stripe size 1MB
- Location: `/lustre/scratch/userid`

HowTo use Lustre

Best practices:

- Change your default stripe count to one! Specially if doing one file per process.
- Use stripe count of more than one only when needed.
- Use single/multiple shared files, and stripe counts multiple of 48 to get the best bandwidth
- Avoid using in Lustre: `ls -lt`
- If you need to monitor the progress, you want to use instead something like: `ls -t1 $destination | head`
- Learn to use the `lfs` command
- Visit our I/O page for more information

http://www.nics.tennessee.edu/io-tips

NICS survival kit

A better/faster way to work with files in Lustre, can be done with the help of the `lfs` instead of the standard Unix commands: `ls`, `find`, `df`.

```
lfs <command> [options]
```

`setstripe/getstripe`	Used to manipulate the striping of files and directories in Lustre
`find`	A much faster way to find files in Lustre. Example: `lfs find /lustre/scratch/lucio --name *.c`
`df`	Shows how much space is left in Lustre
`quota`	Shows how much space I am using in Lustre. Example: `lfs quota -u lucio/lustre/scratch | sed -n 3p`