Debugging code on HPC systems

Historical context – original “bug” were bugs-

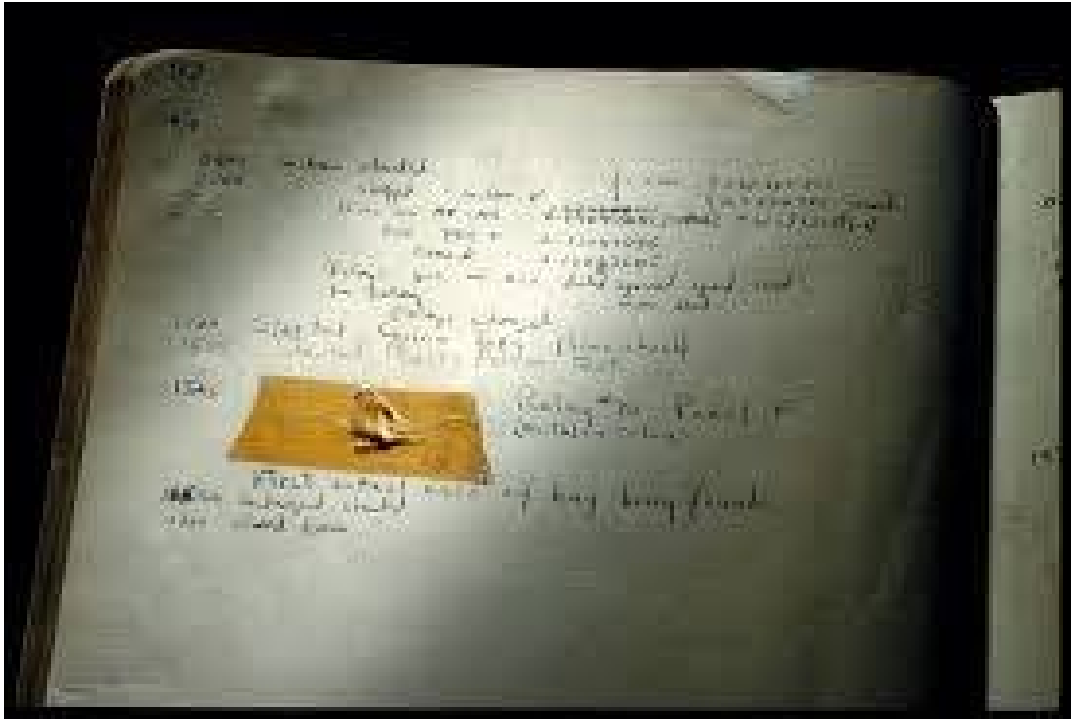


Image Credit: Log Book With Computer Bug, The National Museum of American History.

Eric J. Walter – April 1st, 2019

Brute-force vs. debugger

- “dirty secret” - much of debugging is done via print statements
  Adding print statements changes optimization
  Good idea to turnoff optimization (-O0) to determine if compiler is (maybe partially) responsible

- However, using a debugger can make finding a bug faster and reveal the bug location

- The usual debugging program is gdb
  serial examples (C++, C, Fortran)
  https://www.youtube.com/watch?v=sCtY--xRUyI
  http://www.unknownroad.com/rtfm/gdbtut/
  http://beej.us/guide/bggdb/

- Also, we maintain a parallel debugger (C++,C,Fortran, Python): TotalView
  https://docs.roguewave.com/totalview/8.14.1/pdf/TotalView_User_Guide.pdf
  Temporary suspension of license

- Brute force can be done for parallel codes as well:
  MPI id can be tied to file descriptor / unit number

Core file ; debugging symbols

```
#include <iostream>
using namespace std;
int divint(int, int);
int main()
{
   int x = 5, y = 2;
   cout << divint(x, y);

   x =3; y = 0;
   cout << divint(x, y);

   return 0;
}
int divint(int a, int b)
{
   return a / b;
}
```

This program core dumps with a floating point exception (use -g on compile line to generate debugging symbols)

```
g++ test1.cc -g -O0 -o prog
./prog
Floating point exception (core dumped)
```

Must set core dump size to “unlimited”

tcsh: limit ; limit coredumpsize unlimited
bash: ulimit -a ; ulimit -c unlimited

Then you will see **core.xxxxx** generated

C++ example

With the core file, we try to see what went wrong:

```
#include <iostream>
using namespace std;
int divint(int, int);
int main()
{
   int x = 5, y = 2;
   cout << divint(x, y);

   x =3; y = 0;
   cout << divint(x, y);

   return 0;
}
int divint(int a, int b)
{
   return a / b;
}
```

Internal C++ lib

```
21 [vortex] gdb prog core.58495
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-92.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /sciclone/home10/ewalter/gdb/prog...done.
[New Thread 58495]
Reading symbols from /usr/lib64/libstdc++.so.6...(no debugging symbols found)...done.
Loaded symbols for /usr/lib64/libstdc++.so.6
.
.
Core was generated by `./a.out'.
Program terminated with signal 8, Arithmetic exception.
#0  0x0000000000400777 in divint (a=3, b=0) at test1.cc:18
18        return a / b;
Missing separate debuginfos, use: debuginfo-install glibc-2.12-1.209.el6_9.2.x86_64
(gdb) quit
```

Breakpoints, printing and stepping – C example

```
# include <stdio.h>
int main()
{
 int i, num, j;
 printf ("Enter the number: ");
 scanf ("%d", &num );
 for (i=1; i<num+1; i++) {
   j=j*i;
 }
 printf("The factorial of %d is \
%d \n",num,j);
}
```

```
45 [vortex] gcc -g -O0 test2.c -o prog

46 [vortex] ./prog

Enter the number: 3

The factorial of 3 is 0
```

```
47 [vortex] gdb prog
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-92.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
.
.
.
Reading symbols from /sciclone/home10/ewalter/gdb/prog...done.
(gdb) list (list omitted for brevity)
(gdb) break 1
Breakpoint 1 at 0x40053c: file f1.c, line 1.
(gdb) run
Starting program: /sciclone/home10/ewalter/gdb/prog

Breakpoint 1, main () at f1.c:6
6         printf ("Enter the number: ");
Missing separate debuginfos, use: debuginfo-install glibc-2.12-1.209.el6_9.2.x86_64
(gdb) step
7         scanf ("%d", &num );
(gdb) step
Enter the number: 3
9         for (i=1; i<num+1; i++) {
(gdb) step
10          j=j*i;
(gdb) print j
$1 = 0
(gdb) print i
$2 = 1
(gdb) cont
Continuing.
The factorial of 3 is 0

Program exited with code 031.
(gdb) quit
```

Watchpoints – C example

```
# include <stdio.h>
int main()
{
 int i, num, j;
 printf ("Enter the number: ");
 scanf ("%d", &num );
 for (i=1; i<num+1; i++) {
   j=j*i;
 }
 printf("The factorial of %d is \
%d \n",num,j);
}
```

```
45 [vortex] gcc -g -O0 test2.c -o prog

46 [vortex] ./prog

Enter the number: 3

The factorial of 3 is 0
```

```
47 [vortex] gdb prog
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-92.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
.
.
.
Reading symbols from /sciclone/home10/ewalter/gdb/prog...done.
(gdb) list 0
1       # include <stdio.h>
2
3       int main()
4       {
5         int i, num, j;
6         printf ("Enter the number: ");
7         scanf ("%d", &num );
8
9         for (i=1; i<num+1; i++) {
10          j=j*i;
(gdb) b 9
Breakpoint 1, main () at f1.c:9
(gdb) run
.
.
.
Breakpoint 1, main () at test2.c:9
9         for (i=1; i<num+1; i++) {
Missing separate debuginfos, use: debuginfo-install glibc-2.12-1.209.el6_9.2.x86_64
(gdb) awatch j
Hardware access (read/write) watchpoint 2: j
(gdb) c
Continuing.
Hardware access (read/write) watchpoint 2: j

Value = 0
```

Backtrace, operations – Fortran example

```
program test3
  implicit none
  real last
  real c(10)
  integer p

  ! Initialise c with integer values
  do p=1,10
      c(p)=p
   enddo

   ! Calculate and print ratios
   last = 0.0
   do p=1,10
       call divide(last, c(p))
       last = c(p)
   enddo
end program test3

subroutine divide(d,e)
   implicit none
   real d,e
   print *,e/d
end subroutine divide
```

```
94 [vortex] gfortran -g test3.f90 -o prog -ffpe-trap=zero
95 [vortex] ./prog
Floating point exception (core dumped)
96 [vortex] gfortran -g test2.f90 -o prog
97 [vortex] ./prog
      +Infinity
  2.0000000
  1.5000000
.
.
```

```
47 [vortex] gdb prog
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-92.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
.
Reading symbols from
/sciclone/home10/ewalter/gdb/prog...done.
(gdb) b 1
Breakpoint 1 at 0x40076c: file test2.f90, line 1.
(gdb) run
Starting program: /sciclone/home10/ewalter/gdb/prog

Breakpoint 1, test3 () at test3.f90:1
1       program test3
Missing separate debuginfos, use: debuginfo-install glibc-
2.12-1.209.el6_9.2.x86_64 libgcc-4.4.7-18.el6_9.2.x86_64
libgfortran-4.4.7-18.el6_9.2.x86_64

(gdb) cont
Continuing.
Program received signal SIGFPE, Arithmetic exception.
0x0000000000400887 in divide (d=0, e=1) at test3.f90:23
23          print *,e/d
(gdb) bt
#0  0x0000000000400887 in divide (d=0, e=1) at test3.f90:23
#1  0x00000000004007fc in test3 () at test3.f90:15
#2  0x00000000004008ea in main ()
#3  0x0000003cec61ed1d in __libc_start_main () from
/lib64/libc.so.6
#4  0x00000000004006a9 in _start ()
(gdb) p d
$1 = 0
(gdb) p e
$2 = 1
(gdb) p e/d
$3 = inf
```

GDB summary

- On HPC systems, coredumpsize is set to 0. Remember to **change to unlimited** with **tcsh:** *limit coredumpsize unlimited*, bash: *ulimit -c unlimited*
- gdb **-d <dir>** : will allow you to point to other source directories
- Use **run** to start program; **kill** to stop program
- Set **breakpoints** with break or b to pause execution
- Set **watchpoints** with awatch/rwatch to monitor values of a variable
- Use **print** to report variable value
- gdb has built in "**help**" (type **help** at gdb prompt for more info)

Python Debugger

python has a built-in debugger “pdb” which has similar features to gdb (not as many)

pdb can be launched on the command line or within the python code

```
48 [vortex] more test4.py
import os
def my_exploding_func():
   my_local_var = 'hi'
   os.abort()

my_exploding_func()
```

```
49 [vortex] more test5.py
import os
import pdb
pdb.set_trace()
def my_exploding_func():
   my_local_var = 'hi'
   os.abort()

my_exploding_func()
```

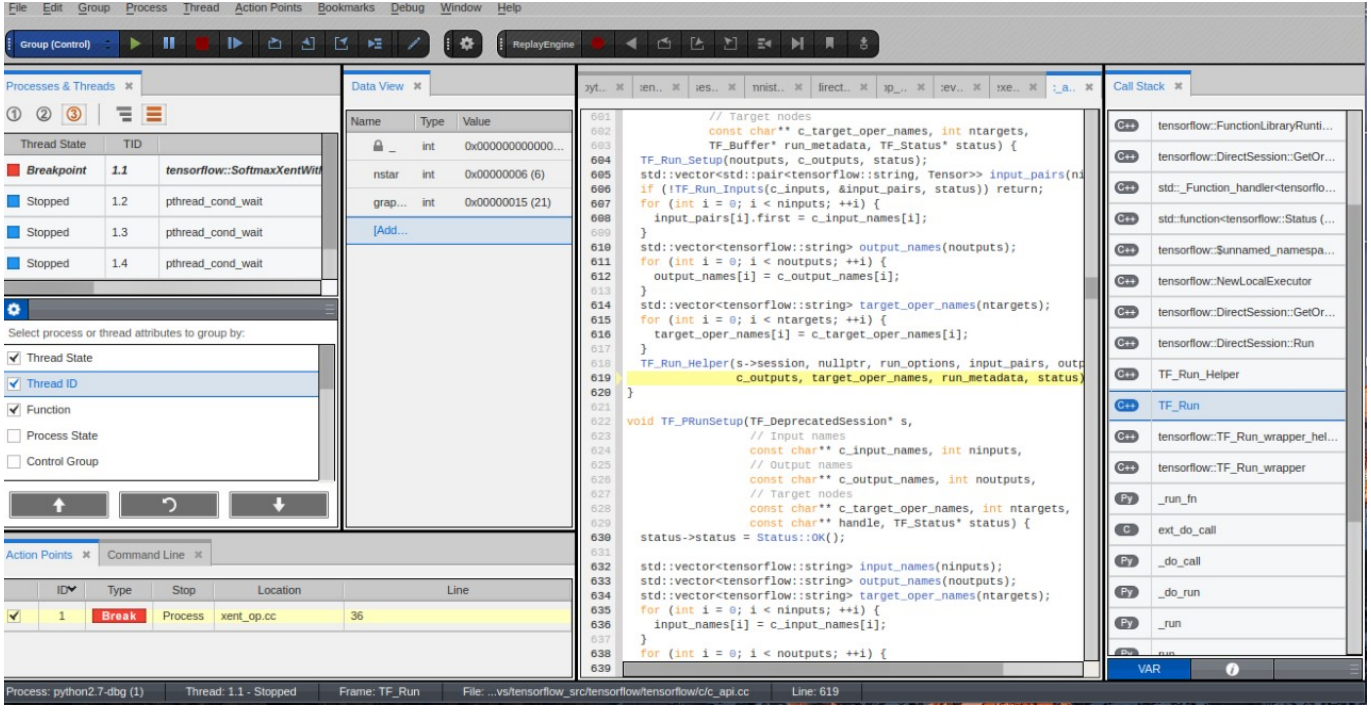
```
50 [vortex] python -m pdb test4.py
> /sciclone/home10/ewalter/gdb/test4.py(1)<module>()
-> import os
(Pdb)
```

```
51 [vortex] python test5.py
> /sciclone/home10/ewalter/gdb/test5.py(5)<module>()
-> def my_exploding_func():
(Pdb)
```

https://codeburst.io/how-i-use-python-debugger-to-fix-code-279f11f75866
https://docs.python.org/2/library/pdb.html

TotalView Parallel Debugger

HPC also maintains TotalView debugger by RogueWave
https://www.roguewave.com/products-services/totalview



- Can used on Vortex and hurricane/whirlwind clusters
- Each cluster can use up 16 cores total

Advanced batch commands/topics

- **freenodes** – shows which nodes are available for running
- **showbf** – shows which nodes are available for running given job duration
- **pbstop** – graphical display of node usage
- **showstart** – predict when my job will start

- **dirty shell** – Will prevent stderr, stdout from being delivered

- **Torque env vars** – Can be used batch script

- **array jobs** – useful for set of similar jobs / parameter study

- **qsubdep** – submit with dependency on other jobs

freenodes

```
1 [vortex] freenodes
Checking nodes for reservations/jobs, please wait...
Rain       (c9)        57
Hurricane  (c10/a)     0
Whirlwind  (c11/c11a)  0
Hail       (c14/a/b)   26
Ice        (c15a/b)    2
Wind       (c16)       4
Vortex     (c18a/b)    0
Vortex-a   (c18c)      0
Bora       (c21)       0
Hima       (c22)       0
```

freenodes – how many nodes are idle?

- Only lists nodes that are idle and not reserved
- Available on all front-ends at W&M and VIMS

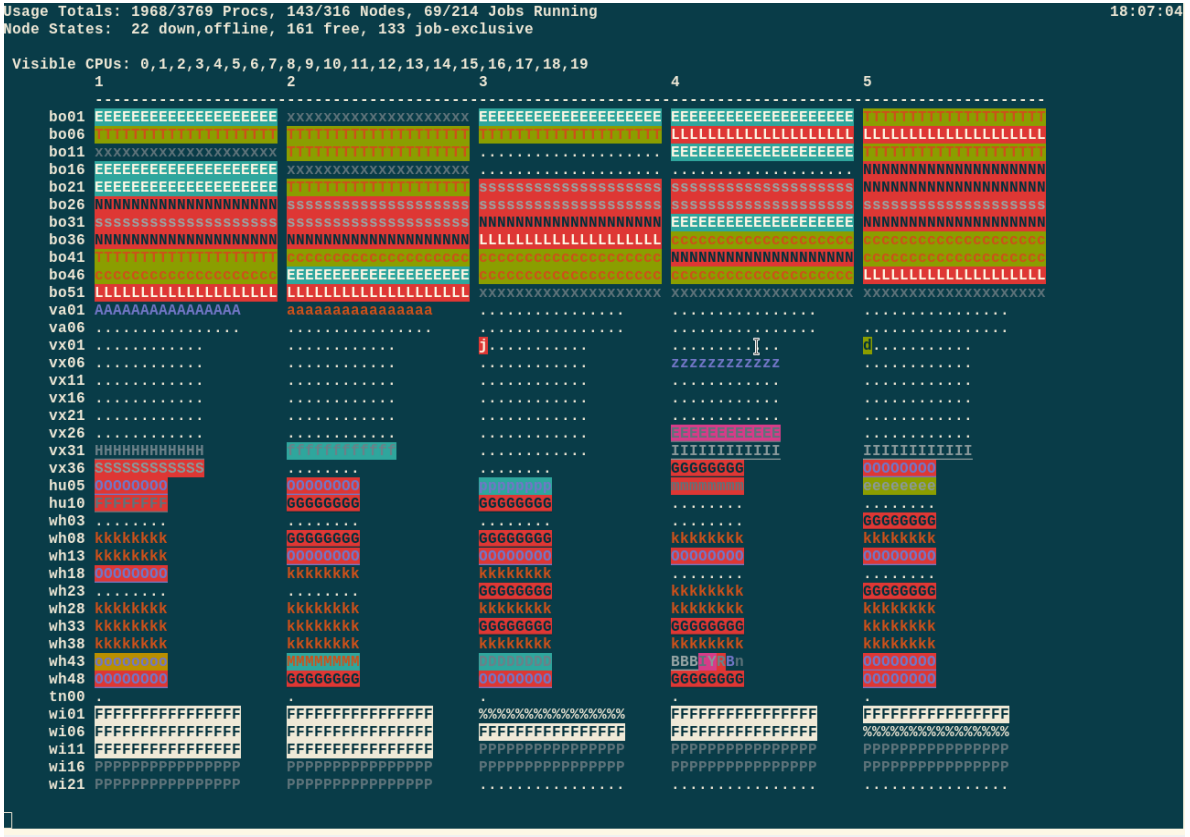
showbf

```
1 [vortex] showbf -f hima
backfill window (user: 'ewalter' group: 'hpcf' partition: ALL) Tue Mar 26 18:01:26

154 procs available for      11:17:52
96 procs available for    1:22:30:33
41 procs available for    2:23:57:24



2 [vortex] showbf -f hima:gpu
backfill window (user: 'ewalter' group: 'hpcf' partition: ALL) Tue Mar 26 18:01:26

96 procs available for    1:22:30:33
41 procs available for    2:23:57:24
```

showbf – shows the backfill window for jobs with a particular feature for a user

- shows how many cores with a given feature are available now and for how long

pbstop



- Shows current population of nodes with glpyhs for each different job
- By default it shows only your jobs.
- Can add `show_user=all` to `~/.pbstoprc`
- Also available at VIMS

showstart

```
132 [vortex] showstart 5402665
job 5402665 requires 40 procs for 1:23:00:00
Earliest start in         14:35:33 on Mon Apr  1 07:54:20
Earliest completion in  2:13:35:33 on Wed Apr  3 06:54:20
Best Partition: DEFAULT
```

- **`showstart`** only shows the result based on current priorities.
- The results of showstart may change
  - as priorities shift
  - as nodes are reserved / offlined for repairs/updates

dirty shell

```
[ewalter@particle ~]$ ssh vortex
Last login: Wed Mar 27 14:37:31 2019 from particle.hpc.wm.edu
------------------------------------------------------------------------------
--
                College of William & Mary / SciClone Cluster
.
.

------------------------------------------------------------------------------
--
isa(3):ERROR:105: Unable to locate a modulefile for 'isa/seouly'
1 [vortex]
```

```
PBS Job Id: 5381634
Job Name:   test
Exec host:  vx01/0-11
Execution terminated
Exit_status=0
resources_used.cput=00:00:00
resources_used.vmem=0kb
resources_used.walltime=00:00:03
resources_used.mem=5588kb
resources_used.energy_used=0
sched_hint=Unable to copy files back - please see the mother superior's log for exact details.
req_information.task_count.0=1
req_information.lprocs.0=12
req_information.thread_usage_policy.0=allowthreads
req_information.hostlist.0=vx01:ppn=12
req_information.task_usage.0.task.0={"task":{"cpu_list":"0-11","mem_list":"0-1","cores":0,"threads":12,"host":"vx01"}}
Error_Path: vortex.sciclone.wm.edu:/sciclone/home10/ewalter/test.o5381634
Output_Path: vortex.sciclone.wm.edu:/sciclone/home10/ewalter/test.o5381634
```

- Torque will not return batch output files (*.o*, *.e*) if your shell produces any output
- You will only see the error below if you turn on
- Torque email in your batch script (-m)

Torque env vars

PBS_VERSION=TORQUE-6.1.1.1
PBS_JOBNAME=test
PBS_ENVIRONMENT=PBS_BATCH
PBS_O_WORKDIR=/sciclone/home10/ewalter
PBS_TASKNUM=1
PBS_O_HOME=/sciclone/home10/ewalter
PBS_WALLTIME=600
PBS_MOMPORT=15003
PBS_GPUFILE=/var/local/torque-6.1.1.1/aux//5381653gpu
PBS_O_QUEUE=submit
PBS_O_LOGNAME=ewalter
PBS_O_LANG=en_US.UTF-8
PBS_JOBCOOKIE=FF66786E4B7AF37266A6E7C14E784B70
PBS_NODENUM=0
PBS_NUM_NODES=1
PBS_O_SHELL=/bin/bash
PBS_JOBID=5381653
PBS_DEFAULT=vx00.sciclone.wm.edu
PBS_O_HOST=vortex.sciclone.wm.edu
PBS_VNODENUM=0
PBS_QUEUE=qq
PBS_O_MAIL=/var/spool/mail/ewalter
PBS_MICFILE=/var/local/torque-6.1.1.1/aux//5381653mic
PBS_NP=12
PBS_O_SERVER=vx00.sciclone.wm.edu
PBS_NUM_PPN=12
PBS_NODEFILE=/var/local/torque-6.1.1.1/aux//5381653
PBS_O_PATH=/sciclone/home10/ewalter/bin:/usr/local/torque-6.1.1.1/bin:/usr/local/torque-6.1.1.1/sbin:/usr/local/maui-r156-GRes/bi
n:/usr/local/Modules/3.2.10/bin:/usr/local/torque-6.1.1.1/bin:/usr/local/torque-6.1.1.1/sbin:/usr/lib64/qt-3.3/bin:/usr/lib64/cca
che:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/ibutils/bin:/opt/puppetlabs/bin

Torque sets a number of environmental variables that can be used in your batch script.

e.g.

```
cd $PBS_O_WORKDIR
./a.out >& output.”$PBS_JOBID”
./a.out >& output.”$PBS_JOBNAME”

cat $PBS_NODEFILE > hostfile
nodes=`cat hostfile|wc -l`
mpirun -np $nodes --hostfile hostfile  ./a.out >& OUT
```

qsubdep

Using job dependencies in Torque can be confusing:

```
94 [vortex] qsub run
5381746
95 [vortex] qme

vortex.sciclone.wm.edu:
                                                                  Req'd       Req'd       Elap
Job ID                  Username    Queue    Jobname          SessID  NDS   TSK   Memory      Time    S   Time
----------------------- ----------- -------- ---------------- ------ ----- ------ --------- --------- - ---------
5381746                 ewalter     qq       test                --      2     24       --   00:10:00 Q       --

96 [vortex] qsub -W depend=afterany:5381746 run

97 [vortex] qme

vortex.sciclone.wm.edu:
                                                                  Req'd       Req'd       Elap
Job ID                  Username    Queue    Jobname          SessID  NDS   TSK   Memory      Time    S   Time
----------------------- ----------- -------- ---------------- ------ ----- ------ --------- --------- - ---------
5381754                 ewalter     qq       test                --      2     24       --   00:10:00 Q       --
5381755                 ewalter     qq       test                --      2     24       --   00:10:00 H       --

98 [vortex]
```

qsubdep simplifies this process:

qsubdep

```
114 [vortex] qsub run
5381765
115 [vortex] qme

vortex.sciclone.wm.edu:
                                                                  Req'd       Req'd       Elap
Job ID                  Username    Queue    Jobname          SessID  NDS   TSK   Memory      Time    S   Time
----------------------- ----------- -------- ---------------- ------ ----- ------ --------- --------- - ---------
5381765                 ewalter     qq       test                --      2     24       --   00:10:00 Q       --
116 [vortex] qsubdep run
5381766
117 [vortex] qme

vortex.sciclone.wm.edu:
                                                                  Req'd       Req'd       Elap
Job ID                  Username    Queue    Jobname          SessID  NDS   TSK   Memory      Time    S   Time
----------------------- ----------- -------- ---------------- ------ ----- ------ --------- --------- - ---------
5381765                 ewalter     qq       test                --      2     24       --   00:10:00 Q       --
5381766                 ewalter     qq       test                --      2     24       --   00:10:00 H       --
118 [vortex]
```

- **qsubdep** will submit new jobs that will be held until all of your other jobs have completed
- it allows for skipping certain jobids and there is a **man** page on all systems (also has **-h**)

array jobs

Torque allows for a single job script to generate multiple jobs, each with a different parameter

```
#!/bin/tcsh
#PBS -N arraytest
#PBS -l nodes=1:vortex:ppn=1
#PBS -l walltime=1:00:00
#PBS -j oe

cd $PBS_O_WORKDIR

echo $PBS_ARRAYID
```

```
153 [vortex] qsub -t 1-10

154 [vortex] qme

vortex.sciclone.wm.edu:

Job ID                    Username    Queue    Jobname          SessID  NDS  ...
------------------------- ----------- -------- ---------------- ------ ----- ...
5381815[]                 ewalter     qq       arraytest           --      1 ...
```

```
153 [vortex] qme -t

vortex.sciclone.wm.edu:
                                                                      Req'd       Req'd       Elap
Job ID                    Username    Queue    Jobname          SessID  NDS   TSK   Memory      Time    S   Time
------------------------- ----------- -------- ---------------- ------ ----- ------ --------- --------- - ---------
5381807[1]                ewalter     qq       arraytest-1         --      1     12       --   00:10:00 Q       --
5381807[2]                ewalter     qq       arraytest-2         --      1     12       --   00:10:00 Q       --
5381807[3]                ewalter     qq       arraytest-3         --      1     12       --   00:10:00 Q       --
.
.
5381807[9]                ewalter     qq       arraytest-9         --      1     12       --   00:10:00 Q       --
5381807[10]               ewalter     qq       arraytest-10        --      1     12       --   00:10:00 Q       --
154 [vortex]
```

```
139 [vortex] cat araytest.o5381807-1
1
140 [vortex] cat arraytest.o5381807-2
2
142 [vortex] cat arraytest.o5381807-8
8
143 [vortex]
```

- Can use `$PBS_ARRAYID` in jobscripts – e.g. use the nth value in a list
- To qdel an array job you must use \ : `qdel 5381807\[1\]`
- To qdel all jobs use: `qdel 5381807\[\]` (or `qdel `qselect -u $USER``)

Questions?