Dealing with zombies on the Discovery Cluster:

Often programs like gromacs ([http://redmine.gromacs.org/projects/gromacs/issues](http://redmine.gromacs.org/projects/gromacs/issues)) due to various reasons terminate unexpectedly and create zombie processes ([http://en.wikipedia.org/wiki/Zombie_process](http://en.wikipedia.org/wiki/Zombie_process)) that use resources and prevent other similar jobs from the same software stack from running. It also may cause a user's shell to be corrupt.

If you have such jobs its best to use the submit script shown below that generates a persistent set of hosts the job ran on. Then run a "kill_zombies" script as explained below.

This will work only if you use all cores on a given node by using the option “#BSUB –R “span[pitle=<max_number_of_cores_in_compute_node>]” and have no other jobs running on the given compute nodes.

The bash submit script example template is below:

```
#!/bin/sh
#BSUB -L /bin/bash
# the name of your job on the queue system
#BSUB -J test1

# the queue that you will use, the example here use the queue called "normal"
# please use bqueus command to check the available queues
#BSUB -q ser-par-10g-3

# the system output and error message output, %J will show as your jobID
#BSUB -o %J.out
#BSUB -e %J.err

#the number of processors that you will use – span all cores on every compute
#BSUB -n 120
#BSUB -R "span[pitle=40]"

#BSUB -cwd /home/nilay.roy/kill_zombies
#BSUB -W 23:40

### enter your working directory, change to your own dir ###
work="/home/nilay.roy/kill_zombies"
mkdir -p $work
cd $work

```
tempfile1=hostlistrun
tempfile2=hostlist
```
tempfile3=hostenumerate
count=0
echo $LSB_MCPU_HOSTS > $tempfile1
declare -a hosts
read -a hosts < ${tempfile1}
for ((i=0; i<${#hosts[@]}; i += 2)) ; do
    HOST=${hosts[$i]}
    CORE=${hosts[($i+1)]}
echo $HOST:$CORE >> $tempfile2
    #for ((j=0; j<$CORE; j++)) ;
    #do
    #echo $HOST >> $tempfile3
    #done
    #count=$(($count+$j))
```
done
```
```
Note the following in the script above:
A file called “host-enumerate” is created. The user then uses this file to kill all the zombies - if present - on the compute nodes the job ran on.
From the same directory after the job completes the user now calls the script “kill_zombies” that is shown below:

```
#!/bin/bash
IFS=$'
for (( i=0; i<5; i++ ));
do
for line in `cat ./host-enumerate`
done
temp_host="$line"
ssh -C $temp_host "$PWD/kill_user_jobs" >> kill_details.txt
done
#Remember to run "rm host-enumerate" from this folder before running another job from the same folder
```

From the above script we see that each compute node has the script “kill_user_jobs” run five times. This is because many “zombie” processes respawn themselves are difficult to kill. In general one needs to adjust the upper limit of “i” in the “for” loop in the script above. I found that 5 work well for me for 99.99% of the time. In your case the number may be higher. I just double this to 10 when killing zombies on the “par-gpu” queue due to processes running on the GPU – do this if you use this queue with GPU enabled code that crashes on the compute nodes with GPU’s.

The script above calls the script “kill_user_jobs” and this too must be in the same directory as the “kill_zombies” script. This script is shown below:

```
#!/bin/bash
killall -u nilay.roy
```

In the script above use your login user name that will be your <myneu_id>.
Remember to run “rm host-enumerate” from this folder before running another job from the same folder.

Output from the run is shown below as an example:

```
Job 758023 is submitted to queue ser-par-10g-3.
JOBID USER STAT QUEUE FROM_HOST EXEC_HOST JOB_NAME SUBMIT_TIME
758023 nilay.roy PEND ser-par-10g-3 discovery4 - test1 May 11 09:26
```

Contact researchcomputing@neu.edu if you have questions or problems or stop by our office at 2 Ell Hall (behind NEU bookstore).

Nilay K. Roy, PhD