Moab

How can I send OS signals to jobs

Issue: In a HPC environment it is sometimes nessassary to send a signal to a job before it reaches its walltime. The mechnism in Moab and TORQUE to make use of this is "-l signal=15@00:30 ", where 15 is the signal and 00:30 is the time before the job reaches its walltime. In this case send signal 15 thirty seconds before the jobs ends. The job script should have a catch or trap for the signal and code to handle any checkpointing or cleanup in the remaining time.

Solution:

Moab TORQUE

When Moab manages TORQUE; signals are much more seemless. By adding a -l signal=nn@tt:tt will allow the use of signals and be interpreted correctly and seemlessly between Moab and TORQUE. Note that the signaling happens via Moab.

Note in the following checkjob the trigger associated with a job was submitted with "-l signal=15@00:30".

[jbooth@support-mpi ~]$ checkjob 83513

job 83513

AName: STDIN

State: Running

Creds: user:jbooth group:jbooth class:batch

WallTime: 00:00:00 of 00:00:30

SubmitTime: Thu Jul 23 09:54:31

(Time Queued Total: 00:00:07 Eligible: 00:00:00)

StartTime: Thu Jul 23 09:54:38

TemplateSets: DEFAULT

NodeMatchPolicy: EXACTNODE

Triggers: termsignal.10$end+-20@0.000000:internal@job:-:signal:15:FALSE

Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: pbs

NodeCount: 1

Allocated Nodes:

[support-mpi2:1]

SystemID: Moab

SystemJID: 503

Notification Events: JobFail

IWD: $HOME

SubmitDir: $HOME

Executable: /opt/moab/spool/moab.job.7g1VeQ

StartCount: 1

Flags: GLOBALQUEUE

Attr: checkpoint

StartPriority: 1

Reservation '83513' (-00:00:10 -> 00:00:20 Duration: 00:00:30)

[jbooth@support-mpi ~]$ qstat -f

Job Id: 83513.support-mpi

Job_Name = STDIN

Job_Owner = jbooth@support-mpi

resources_used.cput = 00:00:00

resources_used.mem = 0kb

resources_used.vmem = 0kb

resources_used.walltime = 00:00:12

job_state = E

queue = batch

server = support-mpi

Checkpoint = u

ctime = Thu Jul 23 09:54:31 2015

Error_Path = support-mpi:/home/jbooth/STDIN.e83513

exec_host = support-mpi2/0

exec_port = 15003

Hold_Types = n

Join_Path = n

Keep_Files = n

Mail_Points = a

mtime = Thu Jul 23 09:54:50 2015

Output_Path = support-mpi:/home/jbooth/STDIN.o83513

Priority = 0

qtime = Thu Jul 23 09:54:31 2015

Rerunable = False

Resource_List.nodect = 1

Resource_List.nodes = 1

Resource_List.signal = 15@00:20

Resource_List.walltime = 00:00:30

session_id = 5579

Variable_List = PBS_O_QUEUE=batch,PBS_O_HOME=/home/jbooth,

PBS_O_LOGNAME=jbooth,

PBS_O_PATH=/opt/moab-7.2-1518952b-b-Unknown12/bin:/opt/8.0.1/bin:/opt

/8.1.0/bin:/usr/lib64/qt-3.3/bin:/opt/mam/bin:/usr/local/sbin:/usr/loc

al/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/opt/moab/bin:/opt/moab

/sbin,PBS_O_SHELL=/bin/bash,PBS_O_WORKDIR=/home/jbooth,

PBS_O_HOST=support-mpi,PBS_O_SERVER=support-mpi

etime = Thu Jul 23 09:54:31 2015

exit_status = 271

start_time = Thu Jul 23 09:54:38 2015

start_count = 1

checkpoint_dir = /tmp

fault_tolerant = False

job_radix = 0

submit_host = support-mpi

nppcu = 1

x = signal:15@00:20;SID:Moab;SJID:503;SRMJID:503

Also note the qstat output as it contains x args with the signal (x = signal:15@00:20;SID:Moab;SJID:503;SRMJID:503). In the event Moab looses track of the job it will re-learn it from TORQURE and use the "x" arguments to re-create the trigger.

```
[jbooth@support-mpi ~]$ qstat -f
Job Id: 83513.support-mpi
  Job_Name = STDIN
  Job_Owner = jbooth@support-mpi
  resources_used.cput = 00:00:00
  resources_used.mem = 0kb
  resources_used.vmem = 0kb
  resources_used.walltime = 00:00:12
  job_state = E
  queue = batch
  server = support-mpi
  Checkpoint = u
  ctime = Thu Jul 23 09:54:31 2015
  Error_Path = support-mpi:/home/jbooth/STDIN.e83513
  exec_host = support-mpi2/0
  exec_port = 15003
  Hold_Types = n
  Join_Path = n
  Keep_Files = n
  Mail_Points = a
  mtime = Thu Jul 23 09:54:50 2015
  Output_Path = support-mpi:/home/jbooth/STDIN.o83513
  Priority = 0
  qtime = Thu Jul 23 09:54:31 2015
```

Rerunable = False

Resource_List.nodect = 1

Resource_List.nodes = 1

Resource_List.signal = 15@00:20

Resource_List.walltime = 00:00:30

session_id = 5579

Variable_List = PBS_O_QUEUE=batch,PBS_O_HOME=/home/jbooth,
PBS_O_LOGNAME=jbooth,
PBS_O_PATH=/opt/moab-7.2-1518952b-b-Unknown12/bin:/opt/8.0.1/bin:/opt
/8.1.0/bin:/usr/lib64/qt-3.3/bin:/opt/mam/bin:/usr/local/sbin:/usr/loc
al/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/opt/moab/bin:/opt/moab
/sbin,PBS_O_SHELL=/bin/bash,PBS_O_WORKDIR=/home/jbooth,
PBS_O_HOST=support-mpi,PBS_O_SERVER=support-mpi

etime = Thu Jul 23 09:54:31 2015

exit_status = 271

start_time = Thu Jul 23 09:54:38 2015

start_count = 1

checkpoint_dir = /tmp

fault_tolerant = False

job_radix = 0

submit_host = support-mpi

nppcu = 1

x = signal:15@00:20;SID:Moab;SJID:503;SRMJID:503

Moab SLURM

When Moab manages SLURM: the HPC admin will need to make a decision if the resource manager will mange signals or if the scheudler will be sending signal instruction. Although SLURM supports sending signals outside of Moabs knowledge; it is suggested that Moab control this mechnism. Currently when you submit with "sbatch --signal=<sig_num>[@<sig_time>]" the signal is not passed up to Moab. Likewise the signal with Moab is sent down to SLURM as a comment and not with --signal.

Comment='SIGNAL=sigurg@30??SJID:2123?SID:Moab'

To submit via sbatch and have Moab manage the signal: you submit as following:

```
sbatch -t 30 --comment='SIGNAL=sigurg@30' cmdl
```

Moab will create a trigger for the signal when Moab learns about the job from SLURM. Moab consumes the comment and sees the note to create a trigger for this job and send a signal down.

checkjob output

```
Triggers: termsignal.6$end+-30@0.000000:internal@job:-:signal:sigurg:FALSE
```

---

mdiag -T output

Compared with a msub trigger

Triggers: termsignal.7$end+-30@0.000000:internal@job:-:signal:23:FALSE

Logging: In the following logging note that Moab is the one sending instructions to SLURM to singal the job.

2015-07-23T09:24:14.166-0600 32392 TRACE1 MRMJob.c:MRMJobSignal:1867 0 MRMJobSignal(2125,-1,15,EMsg,SC)

2015-07-23T09:24:14.166-0600 32392 TRACE1 MWikiJob.c:MWikiJobSignal:110 0 MWikiJobSignal(2125,slurm,-1,15,EMsg,SC)

2015-07-23T09:24:14.166-0600 32392 TRACE1 MWikiI.c:MWikiDoCommand:887 0 MWikiDoCommand(slurm,jobsignal,P,FALSE,CHECKSUM,FALSE,DataP,DataSizeP,CMD=SIGNALJOB ARG=2125 ACTION=SIGNAL VALUE=15,EMsg,SC)

...

2015-07-23T09:24:14.167-0600 32392 TRACE1 MWikiI.c:MWikiDoCommand:1122 0 message sent: 'CMD=SIGNALJOB ARG=2125 ACTION=SIGNAL VALUE=15'

...

2015-07-23T09:24:14.167-0600 32392 INFO MWikiJob.c:MWikiJobSignal:216 0x1002973 job:2125 Job '2125' successfully signaled.

Moab Cray

Moab

Integration with Moab and Cray is done via TORQUE. See the section on TORQUE for singal behavior. The only difference here is that Moab will send the signal via a trigger to the mom where aprun was run. Aprun will then need to translate the signal provided and pass it down to the Cray compute nodes.

Note: Some Cray sites have reported that only signals 23 and 28 work on the Cray with Moab/Torque.

Additional information:

Note that in this checkjob output you see the trigger associated with this job. You can use checkjob to validate that a trigger has been created and associated with the job.

[root@support-slurm moab]# checkjob ALL
job 2125

AName: trap.sh
State: Running
Creds: user:jbooth group:jbooth class:extra
WallTime: 00:00:00 of 00:01:00
SubmitTime: Thu Jul 23 09:23:24
(Time Queued Total: 00:00:20 Eligible: 00:00:00)

StartTime: Thu Jul 23 09:23:44
TemplateSets: DEFAULT
Triggers: termsignal.8$end+-30@0.000000:internal@job:-:signal:15:TRUE
Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: slurm
NodeCount: 1

Allocated Nodes:
[support-sn1:1]

SystemID: Moab
SystemJID: 2125

IWD: /home/jbooth
SubmitDir: $HOME
Executable: /opt/moab/spool/moab.job.EKFula

StartCount: 1
Partition List: slurm
Flags: GLOBALQUEUE
StartPriority: 1
Reservation '2125' (-00:01:00 -> 00:00:00 Duration: 00:01:00)

Likewise you can use mdiag -T to check the triggers on a system to see if the signal was sent, recieved and created:

```
[root@support-slurm moab]# mdiag -T
TrigID Object ID Event AType ActionDate State
---------- ---------- -------- ------ ---------- -----------
termsignal.8* job:2125 end intern -00:01:03 Successful
* indicates trigger has completed
```

Unique solution ID: #1043
Author: Jason Booth
Last update: 2015-07-23 18:57