Setting up Queue Systems
with TORQUE & Maui

Piero Calucci
Scuola Internazionale Superiore di Studi Avanzati
Trieste

November 2008
Advanced School in High Performance and Grid Computing
Outline

1. Obtaining and compiling TORQUE and Maui
2. Configuration
3. Diagnostics & Troubleshooting
TORQUE Source Code

TORQUE is available from www.clusterresources.com
Building TORQUE

- `configure --prefix=/whatever/you/like`
  `make`
  `make install`

- not very clean, actually:
  quite a lot of important files go into `/var/spool`
  — including configuration files!

You can build only the server or MOM components, just tell
`--disable-mom` or `--disable-server`

My favorite install uses a directory that is shared among the
masternode and the computing nodes, so that I need to
build only once.
Maui Source Code

Maui too is available from www.clusterresources.com

You need to register to their site to download the code, and they may contact you later and ask what are you going to do with their software (and offer commercial support for it)
Building Maui

- **same** «*configure; make; make install*»
- `maui` build system need to know where `TORQUE` has been installed
- again, important files go into `/var/spool`
TORQUE Common Configuration Files

- **pbs_environment** contains the environment variables for TORQUE; any minimal set will do e.g.
  
  ```
  PATH=/bin:/usr/bin
  LANG=en_US
  ```

- **server_name** contains the «official» name of the machine where `pbs_server` runs (this is usually your master node)
  The server name **must** be identical to the FQDN e.g.
  ```
  cerbero.hpc.sissa.it
  ```

Both these files reside in the spool directory (`/var/spool/torque`)
server_priv/nodes contains the list of available computing nodes and a list of attributes for each node.

<table>
<thead>
<tr>
<th>node name</th>
<th># of CPUs</th>
<th>«features»</th>
</tr>
</thead>
<tbody>
<tr>
<td>node01</td>
<td>np=2</td>
<td>opteron myri</td>
</tr>
<tr>
<td>node02</td>
<td>np=2</td>
<td>opteron myri</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>node51</td>
<td>np=4</td>
<td>opteron IB</td>
</tr>
<tr>
<td>node52</td>
<td>np=4</td>
<td>opteron IB</td>
</tr>
</tbody>
</table>
The bulk of \texttt{pbs\_server} configuration is written in a (binary) database. You first need to create the empty database with

\texttt{pbs\_server -t create}

This will \texttt{destroy any existing configuration}, create the empty database and start a \texttt{pbs\_server}.

Configuration can then be edited using the \texttt{qmgr} tool. Configuration data are written to \texttt{server\_priv/server\_serverdb} as well as in various other files.
qbmgr doesn’t actually edit the configuration database. It only sends configuration commands to pbs_server which in turn writes the configuration.

This means that:

- you need a running pbs_server to use qmgr (no big issue)
- the pbs_server process needs write access to its own configuration files this is usually considered very bad in any security-conscious environment – unfortunately no easy workarounds are available
One of the most common configuration issues, that prevents the batch system from running any job, involves missing or incorrect `set server managers` and/or `set server operators` lines.
**pbs_mom configuration**

*pbs_mom configuration* can be fairly minimal, the only thing the Mom needs to know is the hostname where *pbs_server* is running on. Useful additions include log configuration, how to handle user file copy and which filesystem to monitor for available space.

*mom_priv/config:*

```
$clienthost    master.hpc
$logevent     0x7f
$usecp        */home  /home
$size[fs=/local_scratch]
```
Maui Configuration
How to Connect to Resource Manager

- simpler approach: a single configuration file (maui.cfg)
- Maui needs to know what RM to connect to and how

  SERVERHOST       borg.cluster
  RMCFG[BORG.CLUSTER]   TYPE=PBS
  RMPOLLINTERVAL 00:00:30
  SERVERPORT 42559
  SERVERMODE NORMAL
  ADMIN1 root
Job priority is recomputed at each scheduler iteration, according to site-defined parameters. If no parameters are set only queue time is taken into account, i.e. the scheduling is strictly FIFO.

Priority components include:

- **Queue Time**: how long the job has been idle in the queue
- **Credentials**: a static priority can be assigned on a user, group, queue basis
- **Fair Share**: historical usage data
- **Resources** requested for the job
Maui Configuration

Job Prioritization: Queue Time and Credentials

QUEUETIMWEIGHT 1
XFACTORWEIGHT 10
CLASSCFG[batch] PRIORITY=1
CLASSCFG[fast] PRIORITY=1000
GROUPCFG[guests] PRIORITY=1
GROUPCFG[users] PRIORITY=1000
GROUPCFG[devel] PRIORITY=10000
USERCFG[DEFAULT] PRIORITY=2000
USERCFG[luser1] PRIORITY=0
The FS priority component must be explicitly enabled by setting its weight to a non-zero value.

- **FSINTERVAL**: 86400 (duration of each FS window)
- **FSDEPTH**: 30 (number of FS windows)
- **FSDECAY**: 0.90 (decay factor applied to older FS windows)
- **FSWEIGHT**: 1
- **FSGROUPWEIGHT**: 240
- **FSUSERWEIGHT**: 10
Usage targets can be set on a per-user, per-group and per-queue basis.

```plaintext
USERCFG[DEFAULT] FSTARGET=1
GROUPCFG[users] FSTARGET=30
GROUPCFG[devel] FSTARGET=40
```

You can set also FS floors or caps so that priority is affected only when usage drops below the floor or goes above the cap:

```plaintext
GROUPCFG[guests] FSTARGET=5-  
  give a negative priority component if usage is above 5%
USERCFG[master] FSTARGET=20+  
  give a priority boost if usage is below 20%
```
Prologue & Epilogue scripts

pbs_mom looks for scripts in its configuration directory mom_priv. If found, the prologue script is executed just before job start and the epilogue script at job termination. The prologue script performs any initialization that is required on the node for the job to run, while the epilogue undoes the modifications.

/etc/security/access.conf

before prologue

-:ALL EXCEPT root:ALL
disallows login to everybody except root, from anywhere

after prologue

-:ALL EXCEPT root
someuser:ALL
now allows someuser to login
Query and control remote `pbs_mom`:

```
# momctl -d3 -h i602
```

Host: i602/i602.hpc  Server: master.hpc  Version: 1.2.0p6

HomeDirectory: /var/spool/PBS/mom_priv
MOM active: 6907718 seconds
Last Msg From Server: 213582 seconds (DeleteJob)
Last Msg To Server: 1 seconds
Server Update Interval: 45 seconds
Init Msgs Received: 10 hellos/2 cluster-addrs
Init Msgs Sent: 190 hellos
LOGLEVEL: 0 (use SIGUSR1/SIGUSR2 to adjust)
Communication Model: RPP
TCP Timeout: 20 seconds
Prolog Alarm Time: 300 seconds
Alarm Time: 0 of 10 seconds
Trusted Client List: ...
JobList: NONE
diagnostics complete
Check who is doing what on a node and show node capabilities

# checknode a034

checking node a034
State: Busy (in current state for 1:13:38:12)
Configured Resources: PROCS: 2 MEM: 3949M SWAP: 7242M DISK: 59G
Utilized Resources: PROCS: 2 DISK: 10G
Dedicated Resources: PROCS: 2
Opsys: DEFAULT Arch: [NONE]
Speed: 1.00 Load: 2.000 (ProcSpeed: 2600)
Network: [DEFAULT]
Features: [myri][opteron][opteron-sc]...
Attributes: [Batch]
Classes: [smp2 2:2][smp4 2:2][mpi4 0:2][mpi8 2:2]...
Total Time: 25:14:33:36 Active: 25:04:53:26 (98.43%)
Reservations:
Job ‘30069’(x2) -1:13:38:44 -> 2:10:20:16 (3:23:59:00)
JobList: 30069
"That’s all Folks!"