Torque queue management system*

Gianluca Moro — gianluca.moro@unipd.it†

January 13, 2012

1 What is Torque

It's a queue management system for clusters: for more informations see on the web: http://www.adaptivecomputing.com/products/torque.php. In this guide you can find a quick reference for the user (who just need to use the working system) in section **User guide**, and a reference for the installation in **Administration guide**, for the person who needs to install the software on a cluster.

This is a guide derived from a particular installation, and **not a complete reference**: if you need a more complete manual, there are a lot of resources in internet.

In this guide the main server node (the one running `pbs_server`, and the cluster main access point to submit the works) is called `cluster`, ip 192.168.0.254, the other nodes (only computation nodes) are called `blade1` 192.168.0.1, `blade2` 192.168.0.2, ... an so on for all the nodes you have.

2 User guide

At http://www.clusterresources.com/torquedocs/. you can find relevant informations.

†Department of Statistical Sciences, University of Padova

3 Administration guide

3.1 Install the program

Various guides are available, i.e. at http://www.clusterresources.com/torquedocs/. Here follows a quick reference of the steps involved in an installation.

- download `wget http://www.adaptivecomputing.com/resources/downloads/torque/torque-3.0.3.tar.gz`
- extract `tar zxf torque-3.0.3.tar.gz`
- `cd torque-3.0.3`
- `./configure`
- `make`
- `make install`
- `make packages`

Now the client packages must be installed in all the computing blades:

```
for i in blade1 blade2 ; do
  scp torque-package-mom-linux-x86_64.sh ${i}:/tmp/. ;
done
for i in blade1 blade2 ; do
  scp torque-package-server-linux-x86_64 ${i}:/tmp/. ;
done
for i in blade1 blade2 ;
  do ssh ${i} /tmp/torque-package-mom-linux-x86_64.sh --install ;
done
for i in blade1 blade2 ; do
  ssh ${i} /tmp/torque-package-server-linux-x86_64 --install ;
done
```

On the server node we need to start `pbs_server`:

```
cp contrib/init.d/debian.pbs_server /etc/init.d/pbs_server
update-rc.d pbs_server defaults
/etc/init.d/pbs_server start
```

on the other nodes `pbs_mom` must be installed in the same way.

3.2 Configuring torque nodes

The server node `cluster` must be initialized:

```
pbs_server -t create
```

and the command `pbsnodes` shows the active nodes:

```
pbsnodes -a
```

Now nothing should be active! To insert nodes in the configuration, the file `/var/spool/torque/server_priv/nodes` must be something like:

```
cluster.loc np=8
blade1.loc np=8
blade2.loc np=8
```

where there `np=8` indicates that the node has 8 cores available.
Each computing node must have the following configuration:

```
root@blade1:/var/spool/torque/mom_priv# cat config
$pbsserver      cluster.loc  # note: hostname running pbs_server
$logevent       255          # bitmap of which events to log
$
root@blade1:/etc# cat /var/spool/torque/server_name
cluster.loc
$
root@blade1:/etc# cat /etc/resolv.conf
search yourdomain.com
nameserver 192.168.0.254
$
```

for each computing node, the local service must be started:

```
root@blade1:~# /etc/init.d/pbs_mom start
```

3.3 The DNS issue

There must be a working **direct and reverse DNS**: failing to do so, the nodes will result as `down` (even if `ping blade1` works correctly). Having the cluster in a local network, an option is to setup a local DNS on the master node (the one named `cluster` in our howto).

To set up a DNS look for the relevant `bind` documentation: here we show just the modified files and we suppose you know how to use them. In the local DNS (`cluster`, with ip 192.168.0.254 in our example) you need `bind` and the following configurations:

File `/etc/bind/named.conf.options`:

```
options {
        directory ‘‘/var/cache/bind’’;

        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };

        version none;
        allow-query { 147.162.35.243; 192.168.0.0/24; };
        allow-transfer { none; };
};
```

File `named.conf.local`:

```
zone ‘‘loc’’ IN {
        type master;
        file ‘‘/etc/bind/db.loc’’;
};

zone ‘‘0.168.192.in-addr.arpa’’ {
        type master;
        file ‘‘/etc/bind/db.0.168.192’’;
};
```

File `db.loc`:

```
$TTL    24h

loc.   IN  SOA   cluster.loc.   root.cluster.loc (
        2012010903 ; serial number
        3h         ; refresh time
        30m        ; retry time
        7d         ; expire time
        3h         ; negative caching ttl
)

; Nameservers
```

```
loc.    IN    NS  192.168.0.254.

; Hosts
cluster.loc.       IN  A   192.168.0.254
blade1.loc.       IN  A   192.168.0.1
blade2.loc.       IN  A   192.168.0.2
```

File `db.0.168.192`:

```
$TTL    24h

0.168.192.in-addr.arpa.   IN  SOA   cluster.loc.   root.cluster.loc (
        2012010902 ; serial number
        3h         ; refresh time
        30m        ; retry time
        7d         ; expire time
        3h         ; negative caching ttl
)

; Nameservers
0.168.192.in-addr.arpa.  IN  NS  192.168.0.254.

; Hosts

254.0.168.192.in-addr.arpa.     IN PTR  cluster.loc.
1.0.168.192.in-addr.arpa.       IN PTR  blade1.loc.
2.0.168.192.in-addr.arpa.       IN PTR  blade2.loc.
```

and all the blades' `resolv.conf` must have the line

```
nameserver 192.168.0.254
```

Restart the `pbs_server` and all `pbs_mom` and the command `pbsnodes -a` should give a list of `free` nodes.

3.4 Management notes

- `qterm` stop the server
- `./torque.setup username` create the admin user
- `kill -SIGUSR1 ‘pgrep pbs_server‘` raise the log level up by one. The same for pbs_mom.
- `momctl -d3` shows mom's status