

Introduction to Torque and Maui Batch and Queue Systems

Piero Calucci
SISSA



**Scuola Internazionale Superiore
di Studi Avanzati**

Outline

- shopping list
 - how to install
 - minimal config
 - security
 - real-world config
 - monitoring
 - more bells&whistles:
 - reserved queues, acl, reservations
 - prologue&epilogue
 - account manager
- 

Shopping List

<http://www.adaptivecomputing.com/products/open-source/>

- resource manager: **TORQUE**
- scheduler: Maui
- (optional) account manager: Gold

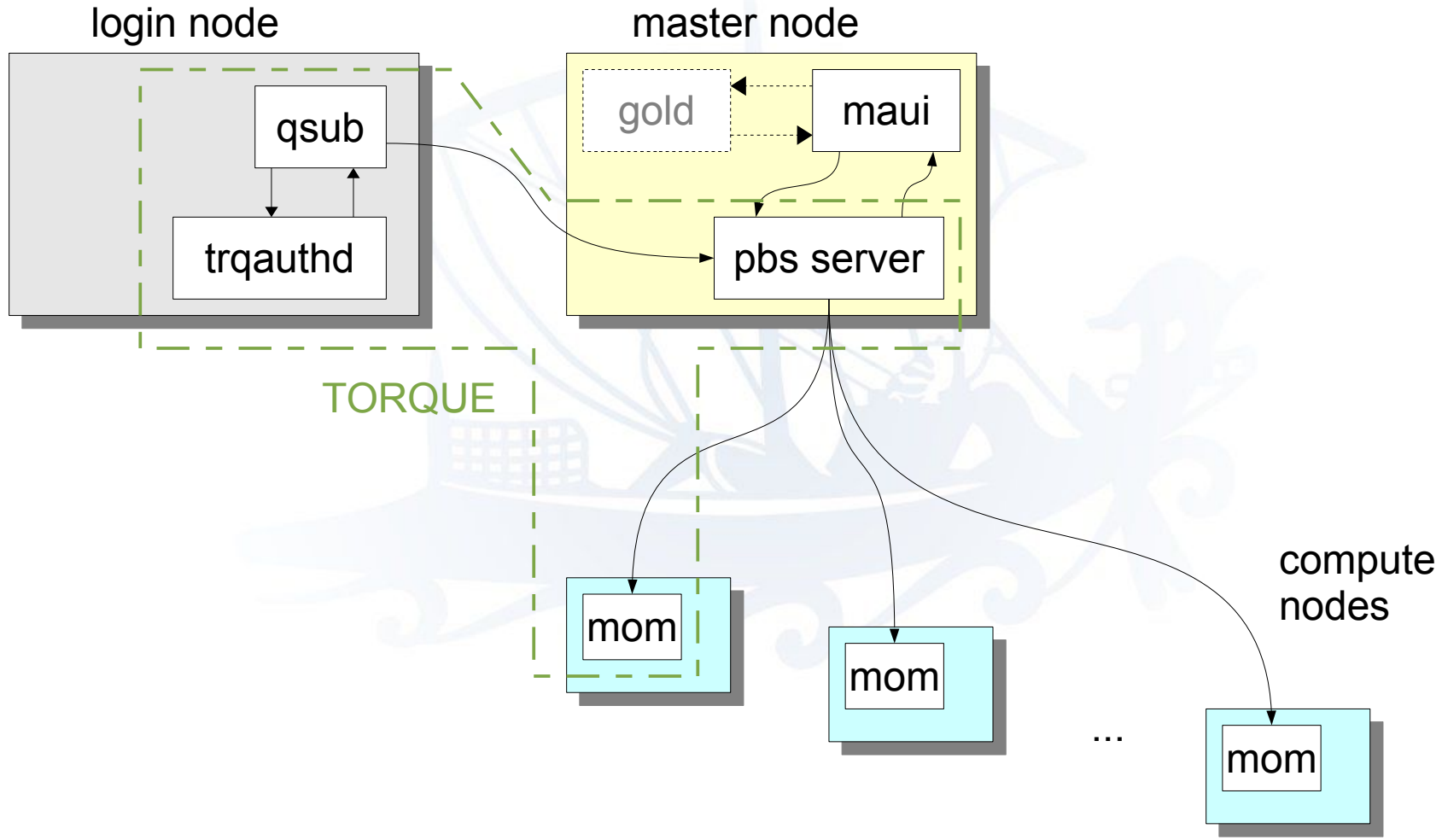
Installation

- good old “configure ; make ; make install”
- but keep an eye on paths
 - TORQUE “wants” to keep config files alongside runtime data
- RPM (or other package format of your choice) strongly recommended for TORQUE
 - you have to install (part of) it to computing nodes!

TORQUE

- `pbs_server` (aka `qserverd`) on master node
- `pbs_mom` (aka `qnode`) on compute nodes
 - also `pbsdsh`
- admin commands on master node
 - `qmgr`, `tracejob`, `pbsnodes`, ...
- user commands on login node
 - `qsub`, `qdel`, `qstat`, ...
- authorization component everywhere

TORQUE/Maui big picture



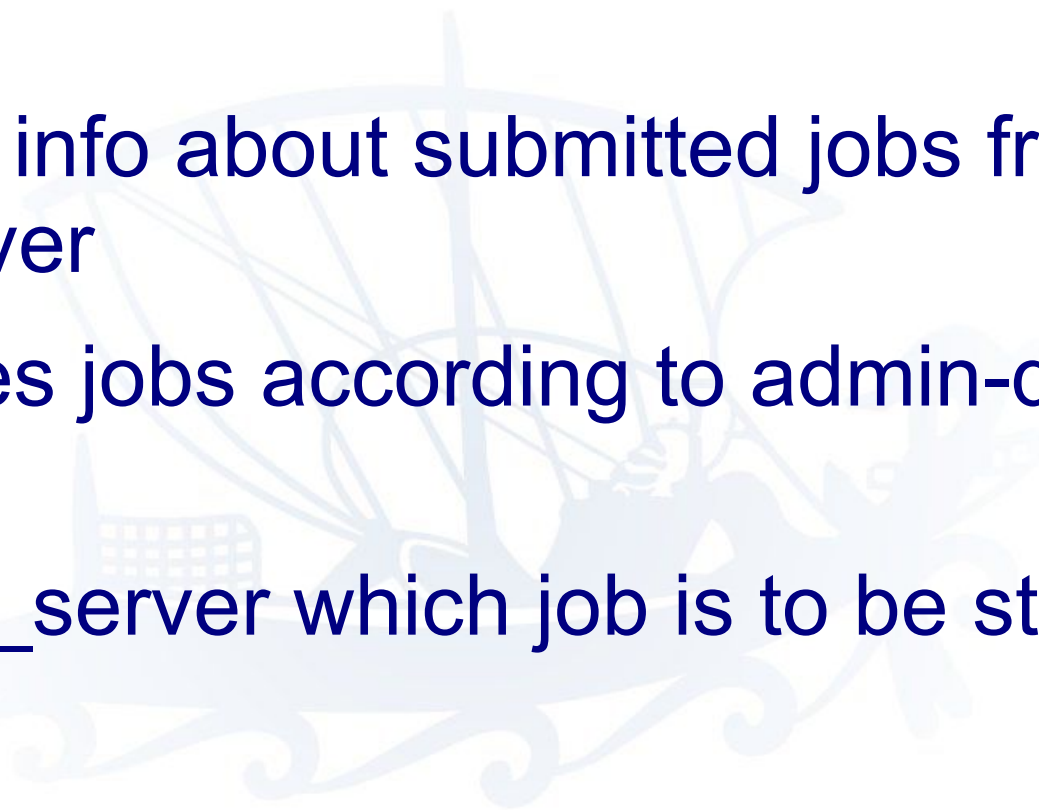
pbs_server

- receives job submissions from qsub (user)
- services info requests (user/admin)
- services configuration commands from qmgr (admin)
- talks with maui to schedule jobs
- delivers jobs to nodes
- notifies users of job status

pbs_mom

- waits for orders from pbs_server
- starts user processes on compute node
- talks to other pbs_moms (“sisters”) to setup multi-nodes jobs
- reports user processes status to pbs_server
- collects stdout/stderr from user processes
- kills user processes if resource limits are exceeded

maui

- receives info about submitted jobs from pbs_server
 - schedules jobs according to admin-defined policies
 - tells pbs_server which job is to be started when
- 
- A faint, light blue watermark is visible in the background of the slide. It depicts a person sitting in a boat, possibly a traditional Hawaiian outrigger canoe, with a large, curved sail or canopy structure behind them. The watermark is centered horizontally and vertically behind the text.

Queues

- a *routing queue* receives jobs from users and sends them to one or more *execution queues*
 - routing queues are optional
- an *execution queue* sends jobs to compute nodes
 - (when the scheduler says so)

Queue Config

- queue name
- queue type (routing or execution)
- queue state (enabled, started)
- resource limits and resource defaults
- ACL

Queue Config Example

```
create queue q1
set queue q1 queue_type = Execution
set queue q1 max_user_queuable = 400
set queue q1 resources_max.nodect = 16
set queue q1 resources_max.walltime = 12:00:00
set queue q1 resources_default.nodes = 1:ppn=20
set queue q1 resources_default.walltime = 00:01:00
set queue q1 enabled = True
set queue q1 started = True
```

- most parameters are optional, but **resource_max.*** are highly recommended
- **max_user_queuable** or **max_queuable** also recommended (you don't want to see how your scheduler performs when a rogue script submits a dozen millions jobs all at once)

Maui Config

- how to connect to resource manager
- **job priority config**
 - this is probably the most «interesting» part of it
- node priority config
- limits for users, groups, queues, ...
- standing reservations

Maui Config Example

```
SERVERHOST      master1
ADMIN1          root
RMCFG [MASTER1] TYPE=PBS
RMPOLLINTERVAL 00:00:30
SERVERPORT      42559
SERVERMODE      NORMAL

FSPOLICY        DEDICATEDPES
FSDEPTH         30
FSINTERVAL      86400
FSDECAY         0.90
FSWEIGHT        1
FSUSERWEIGHT    360
FSGROUPWEIGHT   1440
USERCFG [DEFAULT] FSTARGET=1
GROUPCFG [DEFAULT] FSTARGET=10
USERWEIGHT      1
GROUPWEIGHT     1
QUEUETIMEWEIGHT 1
XFACTORWEIGHT  10
```

Security

- not much...
 - trqauthd running as root validates the user making the request
 - accept requests from a given list of hosts
- **never** allow requests from untrusted hosts!
- **never** allow untrusted hosts on the same network!

Real-World Config: Queues

```
# qstat -q
```

Queue	Walltime	Node	Run	Que	Lm	State
wide	08:00:00	32	2	2	3	E R
gpu	12:00:00	2	8	86	--	E R
regular	12:00:00	16	37	77	--	E R
reserved2	12:00:00	2	0	0	--	D R
devel	01:00:00	2	0	0	2	E R
long	48:00:00	8	2	0	8	E R
test	96:00:00	--	0	0	--	D R
reserved3	12:00:00	2	0	0	--	E R
reserved1	96:00:00	8	2	1	--	E R
extra	12:00:00	1	0	0	7	E R
			51	166		

queue name

resource limits

currently running
& queued jobs

queue state:
Enabled/Disabled
Running/Stopped

Real-World Config: Scheduler

```
USERCFG[DEFAULT]      FSTARGET=1      PRIORITY=1000  MAXPROC=1280
GROUPCFG[DEFAULT]    FSTARGET=0
GROUPCFG[no-organization] FSTARGET=0.1    PRIORITY=-1000000
GROUPCFG[ap]          FSTARGET=8
GROUPCFG[cm]          FSTARGET=27
GROUPCFG[he]          FSTARGET=2
GROUPCFG[ma]          FSTARGET=11
```

...several more groups...

```
XFACTORWEIGHT  10
XFACTORCAP     10
XFMINWCLIMIT   1:00:00
```

```
NODEALLOCATIONPOLICY  PRIORITY
NODECFG[DEFAULT]     PRIORITY=1  PRIORITYF="PRIORITY-CMEM"
```

```
# max 30% (of 224 nodes) for 'long' jobs
CLASSCFG[long]      MAXPROC=1340  MAXNODE=67
```

```
# max 30% (of 224 nodes) for 'wide' jobs
CLASSCFG[wide]      MAXPROC=1340  MAXNODE=67
```

fair share
config

priority function
for node allocation

job priority
fine tuning

hard
limits

Monitoring

- what is the «normal» behavior of your cluster?
- how do you identify anomalies?
- are those Bad Things happening, or just unexpected events?
- tools:
 - log analysis
 - (near-)real-time view
 - detection of system anomalies

Monitoring: log analysis

- good old logwatch, with a couple custom scripts
 - TORQUE can log to syslog
 - be careful with mom logs: plan for 1k lines/day/node (and more if you run many short jobs)
- maui logs compress each job in a single line
 - great for statistics / accounting
 - but you have no data at all until the job is completed

Monitoring: log analysis

group	jobs	cores*hours	avg. queue hours
ap	31	12335	15
cm	136	17868	0
ictp	2	1612	11
ma	56	8097	0
sbp	80	22002	54
staff	1	18	0
uniud	10	0	0

class	jobs	cores*hours	avg. queue hours
devel	14	10	0
gpu	22	4583	141
long	5	2713	7
regular	291	55843	3
wide	8	972	108

feature	jobs
gpu	22
mem160	9
nogpu	318
public	340

Monitoring: real-time

- **queue status:**
 - (TORQUE) `qstat [-a|-r|-i|-q|-Q]`
 - (Maui) `showq [-r|-i|-b], diagnose -c`
- **job status:**
 - (TORQUE) `tracejob, qstat -f`
 - (Maui) `checkjob, diagnose -j`
- **node status:**
 - (TORQUE) `momctl, pbsnodes`
 - (Maui) `checknode, diagnose -n`

Monitoring: other anomalies

- node allocated to job, but no user process running
 - someone is wasting resources
- node exclusively allocated to job by user *X*, but processes by user *Y* running on it
 - something went bad at the end of previous job
- world-writable directories (e.g. `/tmp`, `/dev/shm`) filling up

Reservations

- **standing reservations**
 - periodical, statically configured in maui.cfg
 - SRCFG[operator] PERIOD=DAY
STARTTIME=08:30:00 ENDTIME=18:00:00
DAYS=MON,TUE,WED,THU,FRI
TASKCOUNT=1 RESOURCES=PROCS:20
HOSTLIST=cn01-01 USERLIST=calucci
- **administrative reservations**
 - one shot, configured on the fly from command line
 - setres -s 08:00:00_09/20 -d 2:00:00
-n replace_DIMM_10 cn02-15

ACL

- host that can connect to pbs_server
 - only allow trusted hosts!
- users / groups that can submit jobs
 - this can be configured per-queue
- who can send administrative commands to pbs_server
- can jobs be submitted from compute nodes?

Reserved Queues

- with a proper combination of ACLs, reservations, node features you can effectively «lock» certain combinations of users-nodes-job types

```
create queue reserved1
set queue reserved1 acl_user_enable = True
set queue reserved1 acl_users = someuser
set queue reserved1 acl_users += otheruser
set queue reserved1 resources_default.neednodes = reserved1
```

```
create node gn05-10
set node gn05-10 properties += reserved1
```

(all other queues use a `resources_default.neednodes` with some node feature that is **not** defined for `gn05-10`)

Prologue & Epilogue

- **privileged** scripts to be executed on compute node just before a job starts, and after job completion
 - executed only on the first nodes in a multi-node job
 - `{pro, epi}logue` executed unconditionally
 - `{pro, epi}logue.somename` can be requested by the user in the job submission
- can create/delete temporary directories, u/mount filesystems, output debug/log data about the job, ...

Account Manager

- **maui** can optionally interface with **gold**
- **gold** keeps track of resource usage by each job/user/group
 - **gold projects** can be linked to user or groups, but can also be defined independently
 - each user can participate in multiple projects, and draw resource credits from multiple accounts
- **maui** can be configured to **not schedule** a job unless enough credit is available

(as you may have guessed, all this quickly becomes quite complex...)

Questions?



<calucci at sissa dot it>