PBS安装使用

来自作物遗传改良国家重点实验室集群系统

软件介绍

PBS(Portable Batch System)最初由NASA的Ames研究中心开发，主要为了提供一个能满足异构计算网络需要的软件包，用于灵活的批处理，特别是满足高性能计算的需要，如集群系统、超级计算机和大规模并行系统。PBS的主要特点有：代码开放，免费获取；支持批处理、交互式作业和串行、多种并行作业，如MPI、PVM、HPF、MPL；PBS是功能最为齐全, 历史最悠久, 支持最广泛的本地集群调度器之一。 PBS的目前包括openPBS, PBS Pro和Torque三个主要分支. 其中OpenPBS是最早的PBS系统, 目前已经没有太多后续开发,PBS pro是PBS的商业版本, 功能最为丰富. Torque是Adaptive Computing Enterprises Inc.（2009年前公司名字是Cluster Resources Inc. ）接过了OpenPBS, 并给与后续支持的一个开源版本.

Torque PBS主要有三个服务：

pbs_server这个这个系统的核心服务，它保证server节点和compute节点的正常通信，运行在server节点上

pbs_sched 这个是作业调度服务，pbs_sched进程与pbs_server进程交互，以判定compute 节点资源使用情况以及生成节点作业分配策略。当用户提交作业之后，这个服务决定作业分配在哪些节点上，这个软件也运行在server节点上。这个软件是Torque内置默认的调度软件，MAUI是Adaptive Computing公司维护的另外一个很强大的开源作业调度软件。

pbs_mom 这个是作业执行服务，作业分配到具体的compute节点之后，由这个软件负责开始、杀掉、管理作业，这个软件运行在compute节点上。

trqauthd是从4.0.0版开始引入的服务，以替代pbs_iff，用于授权pbs_mom进程与pbs_server进程之间建立互信连接。该进程已经启动就会驻留在系统内存，客户端程序利用该进程调用本机loopback接口的15005端口通信。该进程支持多线程执行机制，可以处理大量并发请求。对于trqauthd 这个服务，官方文档时这么说的，trqauthd must be run as root. It must also be running on any host where TORQUE client commands will execute.也就是说这个服务是安装在提交作业的节点上。

用户可以使用qsub指令将作业提交到pbs_server，当pbs_server收到一个新的作业时，该进程会通知pbs_sched进程，pbs_sched从集群中寻找可以分配用于计算该作业的节点资源，并将找到的节点清单发送给pbs_server进程。psb_server进程根据pbs_sched提供的信息将待计算的作业提交到节点清单中的第一个节点并让该节点加载该作业，该节点被指定为该作业的主要执行者，由该节点负责调用清单中其他节点协同并行处理该作业，该节点称为Mother Superior ，协同执行该作业的其他节点则称为sister moms。

安装前的准备

配置好各个节点间的网络互通

实验中一共使用了三个节点

manage 192.168.241.100

slavery01 192.168.241.101

slavery02 192.168.241.102

关闭所有节点的selinux

cd /etc/selinux

vi config

改为：SELINUX=disabled

关闭所有节点的防火墙

Service iptables stop

chkconfig iptables off

配置好测试用户在各节点间的ssh无密码访问

安装torque PBS所需的软件

torque安装前要先安装libxml2包、openssl包、C编译器(gcc)和POSIX编译器(gmake)。

openssl是一款开源的SSL(安全套接字协议)软件包，采用SSL公开密钥技术，实现传输层的通信加密功能。包含很多的证书认证方式、密码算法库、SSL协议库以及应用程序。用openssl version –a可以查看是否安装了openssl。如果是类Debian系统，还要sudo apt-get install libssl-dev。

SSL 认证流程

SSL 客户端在TCP连接建立之后，发出一个消息（包含了客户端所支持的算法列表）给服务器端，然后服务器端返回一个数据包（确定了这次通信所需算法）和 SSL 服务器端的证书（包含了公钥）给客户端。客户端随后会用收到的公钥将消息加密再传送，该加密消息只能用 SSL 服务器端私钥解密，即便中途被截取也无法获知内容。

libxml2是个C语言的XML程式库，能简单方便的提供对XML文件的各种操作，并且支持XPATH查询，及部分的支持XSLT转换等功能。

libxml2提供了解决方法，它很体贴地在/usr/local/bin目录下为您提供了xml2-config、xmlcatalog、xmllint三个便利的工具。可以用这个来看是否安装过。编译程序时如果用到xml2库，使用g++ a.cpp -lxml2。

在HP上安装libxml2时，从ftp://xmlsoft.org/libxml2/这里选择下载libxml2-2.8.0.tar.gz，使用./configure（需要加上选项--disable-shared –without-pic，否则会出现错误，错误为LibXML Compile Error: relocation R_X86_64_32 against `a local symbol' can not be used when making a shared object; recompile with -fPIC），make，make install。

本实验系统为redhat，因此可以用yum安装所需的软件

yum install libtool-devel libxml2-devel openssl-devel gcc gcc-c++ boost-devel

在已有的系统中，上面有些软件或许已经存在，可以用rpm命令检查软件是否安装

rpm –qa |grep libxml

软件包下载

下载地址：http://www.adaptivecomputing.com/support/download-center/torque-download/

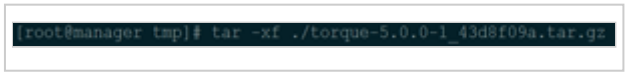
此次实验中使用torque-5.0.0-1_43d8f09a.tar.g这个版本。

Server端软件安装

软件安装

将下载的软件包解压

进入到软件的解压路径中进行安装，此时路径为/tmp/torque-5.0.0-1_43d8f09a ./configure
Make make install 注：--with-server-home=DIR
使用这个参数可以设置$TORQUE_HOME的值，其默认值为/var/spool/torque



初始化serverdb

serverdb文件描述pbs_server配置属性以及队列信息，要运行pbs_server必须先对serverdb做初始化。TORQUE官方推荐的初始化方法是运行./torque.setup脚本，该脚本文件在TORQUE安装包内提供。torque.setup脚本是通过调用pbs_server –t指令来初始化serverdb，附加的帐号将被指定为TORQUE的进程管理员和执行者。设置管理员用户为root

./torque.setup root

Compute节点软件安装

制作安装包

在刚刚的安装目录即/tmp/torque-5.0.0-1_43d8f09a中制作compute节点所需的包

make packages

这一步会在当前目录中生成如下的脚本

./torque-package-clients-linux-i686.sh

./torque-package-mom-linux-i686.sh

./torque-package-server-linux-i686.sh

./torque-package-gui-linux-i686.sh

./torque-package-devel-linux-i686.sh

在各个compute节点安装所需软件

将上述生成的./torque-package-clients-linux-i686.sh ./torque-package-mom-linux-i686.sh 两个脚本cp到各个compute节点，然后进行安装

./torque-package-clients-linux-i686.sh --isntall

./torque-package-mom-linux-i686.sh --install

如果系统有用NFS共享的目录，可以将这两个脚本cp到该目录中，然后用psh、nprsh之类的命令在各个节点中运行上述两个安装命令。

将TORQUE设置为服务

软件安装完成之后，会有$TORQUE_HOME变量，默认采用默认选项安装时，其值为/var/spool/torque。

设置server节点的服务

在server节点，将pbs_sched 、pbs_server、trqauthd设置为系统服务。

进入步骤4中的软件安装目录，即/tmp/torque-5.0.0-1_43d8f09a

> cp contrib/init.d/pbs_server /etc/init.d/pbs_server

> chkconfig --add pbs_server

> cp contrib/init.d/pbs_server /etc/init.d/pbs_sched

> chkconfig --add pbs_sched

> cp contrib/init.d/pbs_server /etc/init.d/trqauthd

> chkconfig --add trqauthd

对于trqauthd 这个服务，官方文档时这么说的，trqauthd must be run as root. It must also be running on any host where TORQUE client commands will execute.也就是说这个服务是安装在提交作业的节点上。

设置compute节点的服务

进入步骤4中的软件安装目录，即/tmp/torque-5.0.0-1_43d8f09a，将contrib/init.d/pbs_mom拷贝到各个计算节点的/etc/init.d/目录中，并设为系统服务。（有的）

>scp ./contrib/init.d/pbs_mom root@slavery01:/etc/init.d/

>chkconfig --add pbs_mom

>service pbs_mom start

>scp ./contrib/init.d/pbs_mom root@slavery02:/etc/init.d/

>chkconfig --add pbs_mom

初始化server节点上的TORQUE

初始化serverdb

serverdb文件描述pbs_server配置属性以及队列信息，要运行pbs_server必须先对serverdb做初始化。TORQUE官方推荐的初始化方法是运行./torque.setup脚本，该脚本文件在TORQUE安装包内提供。torque.setup脚本是通过调用pbs_server –t指令来初始化serverdb，附加的帐号将被指定为TORQUE的进程管理员和执行者。设置管理员用户为root

./torque.setup root

重启pbs_server服务

Serverdb初始化完成之后，需要重启pbs_server服务。在server节点上运行下面的命令

> qterm –t quick

> pbs_server

运行qmgr命令

>qmgr –c “p s”

可以看到如下的Torque默认配置

#

Create queues and set their attributes.

#

#

Create and define queue batch

#

create queue batch

set queue batch queue_type = Execution

set queue batch resources_default.nodes = 1

set queue batch resources_default.walltime = 01:00:00

set queue batch enabled = True

set queue batch started = True

#

1. Set server attributes.

#

set server scheduling = True

set server acl_hosts = manager

set server managers = root@manager

set server operators = root@manager

set server default_queue = batch

set server log_events = 511

set server mail_from = adm

set server pbs_sched_iteration = 600

set server node_check_rate = 150

set server tcp_timeout = 6

set server job_stat_rate = 300

set server poll_jobs = True

set server mom_job_sync = True

set server keep_completed = 30

set server submit_hosts = manager

set server allow_node_submit = True

set server next_job_number = 12

set server moab_array_compatible = True

set server nppcu = 1

指定compute节点

在TORQUE_HOME/server_priv/nodes文件中制定用于计算的compute节点，其语法为

node-name[:ts] [np=] [gpus=] [properties]

[:ts]参数标记节点为timeshared，该类节点会被列入pbs_server的管控清单并汇报节点状态，但pbs_server不会给该类节点分发作业。

[np=]参数定义节点的处理核心数量，该参数是个虚拟数值，可以不严格按照节点的实际内核数量来设定。如果pbs_server设定了auto_node_np 属性，则np参数可以自动被server检测到。设置命令为：

qmgr -c set server auto_node_np = True

[gpus=]参数用于标记节点附加的GPU数量，数值也可以不严格按照节点GPU实际数量来填写。

[properties]参数允许用设定特殊的字符串对节点进行标记，字串必须以字母开头。

此实验中的文件内容为

slavery01 np=1

slavery02 np=1

重启服务

上述配置完成之后，重启server节点pbs_server、pbs_sched。

> qterm -t quick

> pbs_server

> pbs_sched

启动compute节点上的pbs_mom

> pbs_mom

PBS测试

上述工作完成之后可以测试刚刚安装配置的pbs。

检查所有的队列是否正确配置

> qstat –q

server: manager

Queue Memory CPU Time Walltime Node Run Que Lm State

---------- ------ -------- -------- ---- --- --- -- -----

batch -- -- -- -- 0 0 -- E R

----- -----

1. 0

这里只有一个默认的batch队列

检查默认配置

>qmgr -c 'p s'

其输出见上面的步骤6。

检查所有的节点是否正常

>pbsnodes –a

slavery01

```
state = free
```

```
power_state = Running
```

```
np = 1
```

```
ntype = cluster
```

```
status = rectime=1414240729,macaddr=00:0c:29:08:60:19,cpuclock=Fixed,varattr=,jobs=,state=free,netload=206960824,gr
```

```
mom_service_port = 15002
```

```
mom_manager_port = 15003
```

slavery02

```
state = free
```

```
power_state = Running
```

```
np = 1
```

```
ntype = cluster
```

```
status = rectime=1414240732,macaddr=00:0c:29:22:8a:a7,cpuclock=Fixed,varattr=,jobs=,state=free,netload=50334819,gre
```

```
mom_service_port = 15002
```

```
mom_manager_port = 15003
```

这里有两个节点slavery01、slavery02，正常情况下state = free，如果state = down，可能是需要过一会儿再运行这个命令。

检查作业提交

使用普通用乎进行测试

> su –nisuser1

> echo "sleep 30" | qsub

verify jobs display

> qstat

JJob ID Name User Time Use S Queue

------------------------ ---------------- --------------- -------- - -----

12.manager STDIN nisuser1 0 R batch

可以看到测试正常。

如果测试中遇到任何问题，可以查看$TORQUE_HOME/server_logs/中的日志，找出问题所在。

可能会出现的问题

可能出现的问题：

1、在service trqauthd start或是启动pbs_mom等进程时报错：/usr/local/sbin/trqauthd: symbol lookup error: /usr/local/sbin/trqauthd: undefined symbol，一般出现这样的错误都是因为trqauthd依赖的库文件路径不正确或是依赖的库文件为旧版的库，可用ldd /usr/local/sbin/trqauthd查看它的依赖库。

我们安装的是torque，因此首先查看是否是libtorque.so.2库出现问题，查看它是否为新安装的库，用ls –l查看它的日期，我遇到这个问题是因为之前有人安装过torque，因此libtorque.so.2这个库版本为旧版本，替换为新的版本后就解决了该问题。

2、pbsnodes时，显示节点状态为down，停机。解决方案是qmgr –a –c ‘set node XXX state=free’ 就解决了。

3、qterm –t quick是立即停止pbs相关服务。

4、在提交作业时错误，错误提示为qsub: Bad UID for job execution MSG=ruserok failed validating …，解决方法是在运行pbs_server的服务端执行命令：qmgr -c 'set server allow_node_submit = True'，允许节点提交作业到服务端。

5、提交作业后作业一直处于排队状态，解决方案是qmgr -c "set server scheduling=true"，设置服务端可以调度执行程序。

6、软件安装完成之后，我们可能会忘记启动trqauthd服务，这时运行pbs的命令时出现的报错信息中一般都会包含这句话could not connect to trqauthd，这时只要启动trqauthd这个服务即可。

7、安装过程中运行pbs_mom等命令时，可能出现/usr/local/sbin/pbs_mom: error while loading shared libraries: libtorque.so.2: cannot open shared object file: No such file or directory 这种错误，其原因是没有将/usr/local/lib加入动态连接库，缺少的这个文件/usr/local/lib下。

解决方法：

>echo '/usr/local/lib' > /etc/ld.so.conf.d/torque.conf

>ldconfig

环境变量模块化管理工具modules

在linux服务器上，有时需要安装同一个软件的不同版本，按照需要调用。比如，g03和g09。不幸的是，它们的环境变量会互相冲突，不能共存，鱼和熊掌只能选择选择其一。这个问题在大型超算服务器上更为严重。很多软件依赖的环境变量、动态库各不相同。这是需要一个环境变量模块化工具modules。它的功能很简单，将每个软件的环境变量写一个模块文件modulefile，需调用哪个软件就加载其环境变量模块，执行完成后卸载模块，与之相关的环境变量全部消除。这个神奇的工具是个开源软件。目前，在大型超算服务器上，PBS+modules已经成为标配。

这个软件在我们实际使用的集群中已经配置好了，但在此次实验中我没有配置。

本文主要参考文档：torqueAdminGuide-5.0.1 (http://docs.adaptivecomputing.com/torque/5-0-0/help.htm)

Maui安装配置

Maui是一个强大的作业调度系统，torqure自身带有调度系统pbs_sched，但如果需要做比较复杂的作业和资源控制，则需要使用maui。

Maui安装比较简单，只需要有一个注意的地方，下面安装过程中会说到。

软件下载

下载地址：maui (http://www.adaptivecomputing.com/products/open-source/maui/)

下载maui前需要注册这个官方，我使用的是最新版的Maui 3.3.1 。

Maui安装

Maui因为是作业调度软件，因此只需要在server端安装，compute节点不需要。将软件包解压，进入安装目录maui-3.3.1，

./configure --with-pbs=/usr/local/bin/

--with-pbs这个参数指定pbs-config的位置，如果没有指定，则会报configure: error: can't find pbs-config or libpbs.a这种错误。此外，还可以用—prefix这个参数来指定maui安装的位置，这里使用默认安装，位置为/usr/local/maui。

Maui配置

Maui的配置文件为/usr/local/maui/maui.cfg，如果没有特殊要求，可以不用修改。

将软件包目录下的./maui-3.3.1/contrib/service-scripts/redhat.maui.d拷到/etc/init.d/目录下并更名为maui.d

cp ./maui-3.3.1/contrib/service-scripts/redhat.maui.d /etc/init.d/maui.d

将maui.d文件的18行处改为daemon $MAUI_PREFIX/sbin/maui，将maui.d设为开机启动。

Chkconfig maui.d on

Maui使用

我们可以在maui配置文件中根据我们的需求更改作业调度和资源分配策略，如

USERCFG[test] PRIORITY=100 MAXPROC=100 MAXJOB=50

表示只允许test用户的所有作业最多申请使用100个CPU核心，只需要同时最多执行50个作业。

其它更详细的配置策略可以参考官方文档。

PBS使用

提交作业

命令：qsub

参数：

-l 资源列表

-I（大写的i）交互式提交作业

-m 发送邮件，a 作业放弃时发送 b 作业开始时发送 e 作业退出时发送

-M user 邮件发送给哪个用户

-N 作业名称

-o 作业输出目录，若不指定则输出到工作目录

-e 作业错误输出目录，若不指定则输出到工作目录

-q 指定队列，三种形式queue、@server、queue@server

-d 作业工作目录，默认为用户家目录，可以用环境变量PBS_O_INITDIR来设置

查询作业状态

命令：qstat，用于查询作业状态信息

命令格式：

qstat [-f][-a][-i] [-n][-s] [-R] [-Q][-q][-B][-u]

参数：

-f jobid 列出指定作业的信息

-a 列出系统所有作业

-i 列出不在运行的作业

-n 列出分配给此作业的结点

-s 列出队列管理员与scheduler所提供的建议

-R 列出磁盘预留信息

-Q 操作符是destination id，指明请求的是队列状态

-q 列出队列状态，并以alternative形式显示

-au userid 列出指定用户的所有作业

-B 列出PBS Server信息

-r 列出所有正在运行的作业

-Qf queue 列出指定队列的信息

-u 若操作符为作业号，则列出其状态。

若操作符为destination id，则列出运行在其上的属于user_list中用户的作业状态。

例：# qstat -f 211 查询作业号为211的作业的具体信息。

删除作业

命令：qde，用于删除已提交的作业

命令格式：

qdel [-W 间隔时间] 作业号

例：# qdel -W 15 211 15秒后删除作业号为211的作业

-p 清除作业。当作业被分配的节点不能使用时，作业不会退出，这时可以用这个命令清楚作业。

创建并设置作业队列

-c 表示其后是命令，主要有active、create、delete、set、unset、list、print，均可去其首字母

acl_groups 描述可通过此队列提交作业的用户组，前提是acl_group_enable设置为true。只有在第一用户组在acl_groups中的用户才能使用此队列。

```
> qmgr -c "set queue batch acl_groups=lab"
> qmgr -c "set queue batch acl_groups+=conda@cu01"
> qmgr -c "set queue batch acl_groups+=conda@cu02"
acl_group_enable 输入值为布尔值，设置是否允许通过组来定义使用权限。
> qmgr -c "set queue batch acl_group_enable=true"
acl_group_sloppy 输入值为布尔值，设置是否检索该用户的所有组，而不是只检索其第一用户组
> qmgr -c "set queue batch acl_group_sloppy=true"
acl_hosts 描述一系列主机，可以通过该队列进行提交作业的
> qmgr -c "set queue batch acl_hosts=cu01+cu02"
acl_host_enable描述可提交作业的主机
> qmgr -c "set queue batch acl_host_enable=true"
acl_logic_or描述用户和组限制管理冲突处理，若为TRUE，则二者满足其一即可，若为FALSE，则二者需要都满足
acl_users描述哪些用户可以向此队列提交任务
> qmgr -c "set queue batch acl_users=john"
> qmgr -c "set queue batch acl_users+=steve@h2"
> qmgr -c "set queue batch acl_users+=stevek@h3"
acl_user_enable是否启用acl_users选项
enabled描述该队列是否允许提交作业
> qmgr -c "set queue inter enabled=true"
keep_completed描述在作业退出后应该保持完整状态多少秒
> qmgr -c "set queue inter keep_completed=100"
kill_delay描述一个任务被取消后SIGTERM和SIGKILL信号发送间隔秒
max_queuable描述在给定时间段(running,idle,blocked)，队列中允许的最大作业数
max_running描述在任何给定时间内允许运行的最大作业数
```

max_user_queuable描述每个用户最大作业数，包括runing,idle,blocked段

max_user_run描述每个用户在任何时间段运行的作业数

priority描述该队列的优先级

queue_type

e表示execution，r表示route，描述队列类型，所有的队列均需要描述此选项

resources_available

描述该队列所有可以用累计资源

resources_default描述该队列所有作业的默认资源请求

resources_max描述该队列作业的最大资源限制

resources_min描述该队列作业的最小资源限制

route_destinations描述潜在的与该队列相关的目标队列，旨在queue_type=r中可用

started描述作业是否允许执行

Example

用qmgr创建队列normal

qmgr -c “c q normal”

其中：c 表示穿件，q表示队列

设定队列的类型为可执行队列

qmgr -c “s q normal queue_type=Execution”

其中s表示设置。

设定队列中任务的最大运行时间为24小时(CPU时间)

qmgr -c “s q normal resources_max.cput=24:00:00″

设定该队列中任务最小运行时间为1秒(CPU时间)

qmgr -c “s q normal resources_min.cput=1″

设定该队列中任务默认运行时间为12分钟(CPU时间)

qmgr -c “s q normal resources_default.cput=12:00″

使队列生效

qmgr -c “s q normal enabled=true”

启用队列

qmgr -c “s q normal started=true”

#将normal队列设定为默认队列

qmgr -c “s s default_queue=normal”

#在创建完成normal队列后，还可按照a～h各步骤创建拥有其他属性的队列
#为了方便安装过程，同时也避免重复操作时出错，可使用已有的配置文件完成队列设置工作：

设配置文件名为 pbsconf，使用命令 qmgr < pbsconf 导入队列设置。

其它命令

1. qrun 强制作业运行 管理员用
2. qhold 挂起作业

-h 作业列表，挂起所列出的作业

1. qrls 释放挂起的作业

-h 作业列表

qmove destination jobId 移动作业从当前队列到另一队列，destination有三种形式queue、@server、queue@server

1. qrerun 重新跑作业
2. pbsnodes

-a 显示所有节点的状态信息

-o 让某个节点离线，不参与任务分配

-c 使离线的节点上线

-l 列出某种状态的节点 all 列出所有节点的所有属性 active 列出job-exclusive, job-sharing, or busy的节点 up 列出所有up状态的节点，包括job-exclusive, job-sharing, reserve, free, busy and time-shared的节点

-N 管理员给节点添加任意注释信息

category:PBS
