SGE Roll: Users Guide

Version 3.0.0 Edition

SGE Roll: Users Guide :

Version 3.0.0 Edition

Published September 2003

Table of Contents

Preface

The SGE Roll installs and configures the SUN Grid Engine scheduler.

Please visit the SGE site[1] to learn more about their release and the individual software components.

Notes

1. http://gridengine.sunsource.net/

Chapter 1. Requirements

1.1. Rocks Version

The SGE Roll is for use with Rocks version 3.1.0 (not yet released).

1.2. Other Rolls

The SGE Roll is does not require any other Rolls (other than the HPC Roll) to be installed on the Frontend. Compatiblity has been verified with the following Rolls.

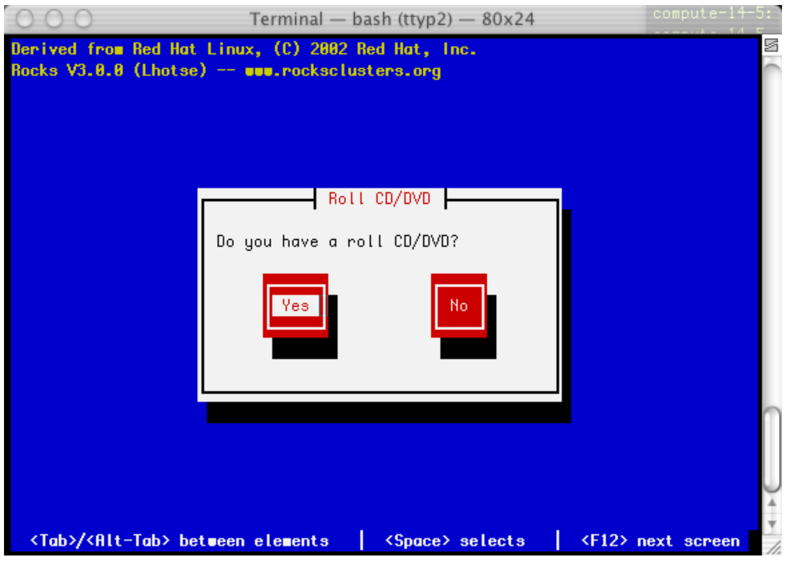
- HPC
- Grid

Chapter 2. Installing the SGE Roll

2.1. Adding the Roll

The SGE Roll must be installed during the Frontend installation step of your cluster (refer to section 1.2 of the Rocks usersguide). Future releases will allow the installation of the SGE Roll onto a running system.

The SGE Roll is added to a Frontend installation in exactly the same manner as the required HPC Roll. Specifically, after the HPC Roll is added the installer will once again ask if you have a Roll (see below). Select 'Yes' and insert the SGE Roll.



```
Derived from Red Hat Linux, (C) 2002 Red Hat, Inc.
Rocks V3.0.0 (Lhotse) -- www.rocksclusters.org

Do you have a roll CD/DVD?

<Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

Chapter 3. Using the SGE Roll

3.1. How to use SGE

This section tells you how to get started using Sun Grid Engine (SGE). SGE is a distributed resource management software and it allows the resources within the cluster (cpu time,software, licenses etc) to be utilized effectively. Also, the SGE Roll sets up Sun Grid Engine such that NFS is not needed for it's operation. This provides a more scalable setup but it does mean that we will lose the high availability benefits that a SGE with NFS setup offers. Another thing that the Roll does is that that generic queues are setup automatically the moment new nodes are being integrated within the Rocks cluster and booted up.

3.2. Setting the SGE environment

When you log into the cluster, the SGE environment would have already been set up for you. The SGE commands should have been automatically added into your $PATH.

```
[sysadm1@frontend-0 sysadm1]$ echo $SGE_ROOT
/opt/gridengine
[sysadm1@frontend-0 sysadm1]$ which qsub
/opt/gridengine/bin/glinux/qsub
```

3.3. Submitting Batch Jobs to SGE

Batch jobs are submitted to SGE via scripts. Here is an example of a serial job script, sleep.sh[1]. It basically executes the sleep command.

```
[sysadm1@frontend-0 sysadm1]$ cat sleep.sh
#!/bin/bash
#
#$ -cwd
#$ -j y
#$ -S /bin/bash
#
date
sleep 10
date
```

Note: Entries which start with `#$` will be treated as SGE options.

- `-cwd` means to execute the job for the current working directory.
- `-j y` means to merge the standard error stream into the standard output stream instead of having two separate error and output streams.
- `-S /bin/bash` specifies the interpreting shell for this job to be the Bash shell.

To submit this serial job script, you should use the **qsub** command.

```
[sysadm1@frontend-0 sysadm1]$ qsub sleep.sh
your job 16 ("sleep.sh") has been submitted
```

For a parallel MPI job script, take a look at this script, linpack.sh[2]. Note that you need to put in two SGE variables, `$NSLOTS` and `$TMPDIR/machines` within the job script.

```
[sysadm1@frontend-0 sysadm1]$ cat linpack.sh
#!/bin/bash
#
#$ -cwd
#$ -j y
#$ -S /bin/bash
#
MPI_DIR=/opt/mpich/gnu

$MPI_DIR/bin/mpirun -np $NSLOTS -machinefile $TMPDIR/machines \
        /opt/hpl/gnu/bin/xhpl
```

The command to submit a MPI parallel job script is similar to submitting a serial job script but you will need to use the `-pe mpich N`. `N` refers to the number of processes that you want to allocate to the MPI program. Here's an example of submitting a 2 processes linpack program using this HPL.dat[3] file:

```
[sysadm1@frontend-0 sysadm1]$ qsub -pe mpich 2 linpack.sh
your job 17 ("sleep.sh") has been submitted
```

If you need to delete an already submitted job, you can use **qdel** given it's job id. Here's an example of deleting a fluent job under SGE:

```
[sysadm1@frontend-0 sysadm1]$ qsub fluent.sh
your job 31 ("fluent.sh") has been submitted
[sysadm1@frontend-0 sysadm1]$ qstat
job-ID  prior name       user         state submit/start at     queue      master  ja-task-
ID
```

```
--------------------------------------------------------------------------------
------
     31     0 fluent.sh  sysadm1      t     12/24/2003 01:10:28 comp-pvfs- MASTER
[sysadm1@frontend-0 sysadm1]$ qdel 31
sysadm1 has registered the job 31 for deletion
[sysadm1@frontend-0 sysadm1]$ qstat
[sysadm1@frontend-0 sysadm1]$
```

Although the example job scripts are bash scripts, SGE can also accept other types of shell scripts. It is trivial to wrap serial programs into a SGE job script. Similarly, for MPI parallel jobs, you just need to use the correct **mpirun** launcher and to also add in the two SGE variables, `$NSLOTS` and `$TMPDIR/machines` within the job script. For other parallel jobs other than MPI, a Parallel Environment or PE needs to be defined. This is covered withn the SGE documentation.

3.4. Monitoring SGE Jobs

To monitor jobs under SGE, use the **qstat** command. When executed with no arguments, it will display a summarized list of jobs

```
[sysadm1@frontend-0 sysadm1]$ qstat
job-ID  prior name       user         state submit/start at     queue      master  ja-task-
ID
--------------------------------------------------------------------------------
------
     20     0 sleep.sh   sysadm1      t     12/23/2003 23:22:09 frontend-0 MASTER
     21     0 sleep.sh   sysadm1      t     12/23/2003 23:22:09 frontend-0 MASTER
     22     0 sleep.sh   sysadm1      qw    12/23/2003 23:22:06
```

Use **qstat -f** to display a more detailed list of jobs within SGE.

```
[sysadm1@frontend-0 sysadm1]$ qstat -f
queuename            qtype used/tot. load_avg arch      states
----------------------------------------------------------------------------
comp-pvfs-0-0.q      BIP   0/2       0.18     glinux
----------------------------------------------------------------------------
comp-pvfs-0-1.q      BIP   0/2       0.00     glinux
----------------------------------------------------------------------------
comp-pvfs-0-2.q      BIP   0/2       0.05     glinux
----------------------------------------------------------------------------
frontend-0.q         BIP   2/2       0.00     glinux
     23     0 sleep.sh   sysadm1      t     12/23/2003 23:23:40 MASTER
     24     0 sleep.sh   sysadm1      t     12/23/2003 23:23:40 MASTER
```

```
############################################################################
 - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS
############################################################################
     25     0 linpack.sh sysadm1      qw    12/23/2003 23:23:32
```

You can also use **qstat** to query the status of a job, given it's job id. For this, you would use the `-j N` option where `N` would be the job id.

```
[sysadm1@frontend-0 sysadm1]$ qsub -pe mpich 1 single-xhpl.sh
your job 28 ("single-xhpl.sh") has been submitted
[sysadm1@frontend-0 sysadm1]$ qstat -j 28
job_number:                 28
exec_file:                  job_scripts/28
submission_time:            Wed Dec 24 01:00:59 2003
owner:                      sysadm1
uid:                        502
group:                      sysadm1
gid:                        502
sge_o_home:                 /home/sysadm1
sge_o_log_name:             sysadm1
sge_o_path:                 /opt/sge/bin/glinux:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/usr/
sge_o_mail:                 /var/spool/mail/sysadm1
sge_o_shell:                /bin/bash
sge_o_workdir:              /home/sysadm1
sge_o_host:                 frontend-0
account:                    sge
cwd:                        /home/sysadm1
path_aliases:               /tmp_mnt/ * * /
merge:                      y
mail_list:                  sysadm1@frontend-0.public
notify:                     FALSE
job_name:                   single-xhpl.sh
shell_list:                 /bin/bash
script_file:                single-xhpl.sh
parallel environment:  mpich range: 1
scheduling info:            queue "comp-pvfs-0-1.q" dropped because it is temporarily not available
                            queue "comp-pvfs-0-2.q" dropped because it is temporarily not available
                            queue "comp-pvfs-0-0.q" dropped because it is temporarily not available
```

3.5. Managing SGE queues

To display a list of queues within the Rocks cluster, use **qconf -sql**.

```
[sysadm1@frontend-0 sysadm1]$ qconf -sql
comp-pvfs-0-0.q
comp-pvfs-0-1.q
comp-pvfs-0-2.q
frontend-0.q
```

If there is a need to disable a particular queue for some reason, e.g scheduling that node for maintenance, use **qmod -d Q** where `Q` is the queue name. You will need to be a SGE manager in order to disable a queue like the `root` account. You can also use wildcards to select a particular range of queues.

```
[sysadm1@frontend-0 sysadm1]$ qstat -f
queuename            qtype used/tot. load_avg arch      states
----------------------------------------------------------------------------
comp-pvfs-0-0.q      BIP   0/2       0.10     glinux
----------------------------------------------------------------------------
comp-pvfs-0-1.q      BIP   0/2       0.58     glinux
----------------------------------------------------------------------------
comp-pvfs-0-2.q      BIP   0/2       0.02     glinux
----------------------------------------------------------------------------
frontend-0.q         BIP   0/2       0.01     glinux
[sysadm1@frontend-0 sysadm1]$ su -
Password:
[root@frontend-0 root]# qmod -d comp-pvfs-0-0.q
Queue "comp-pvfs-0-0.q" has been disabled by root@frontend-0.local
[root@frontend-0 root]# qstat -f
queuename            qtype used/tot. load_avg arch      states
----------------------------------------------------------------------------
comp-pvfs-0-0.q      BIP   0/2       0.10     glinux    d
----------------------------------------------------------------------------
comp-pvfs-0-1.q      BIP   0/2       0.58     glinux
----------------------------------------------------------------------------
comp-pvfs-0-2.q      BIP   0/2       0.02     glinux
----------------------------------------------------------------------------
frontend-0.q         BIP   0/2       0.01     glinux
```

To enable back the queue, you can use **qmod -e Q**. Here is an example of `Q` being specified as range of queues via wildcards.

```
[root@frontend-0 root]# qmod -e comp-pvfs-*
Queue "comp-pvfs-0-0.q" has been enabled by root@frontend-0.local
root - queue "comp-pvfs-0-1.q" is already enabled
root - queue "comp-pvfs-0-2.q" is already enabled
[root@frontend-0 root]# qstat -f
queuename            qtype used/tot. load_avg arch      states
----------------------------------------------------------------------------
comp-pvfs-0-0.q      BIP   0/2       0.10     glinux
----------------------------------------------------------------------------
comp-pvfs-0-1.q      BIP   0/2       0.58     glinux
----------------------------------------------------------------------------
comp-pvfs-0-2.q      BIP   0/2       0.02     glinux
----------------------------------------------------------------------------
frontend-0.q         BIP   0/2       0.01     glinux
```

For more information in using SGE, please refer to the SGE documentation and the man pages.

Notes

1. examples/sleep.sh
2. examples/linpack.sh
3. examples/HPL.dat

Chapter 4. Copyrights

4.1. Sun Grid Engine

The project uses the Sun Industry Standards Source License, or SISSL. This license is recognized as a free and open license by the Free Software Foundation and the Open Source Initiative respectively. This license is a good license where interoperability and commercial considerations are important. Under the SISSL license a user may do what they like with the source base, such as modifing it and extending it, but the user must maintain compatibility if they want to provide their modified version to others. It is not mandatory for the user to contribute back code changes to the project, though of course this is encouraged. If a user wishes to make available a modified version which is not compatible with the project, the license requires that the licensee must provide a reference implementation of sources which constitute the modification, thereby opening the details of any incompatibility/modification which has been introduced. Additionally the non-compliant modified version cannot use the Grid Engine name.