Slurm for users

genotoul

06-11-2017

Bull
atos technologies

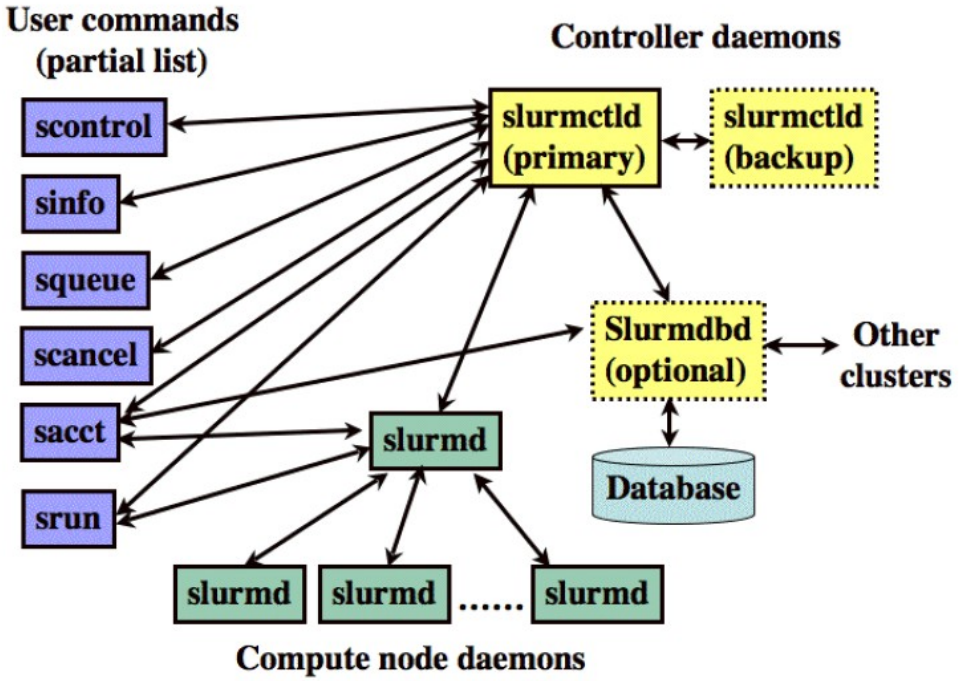
Summary

1. Introduction
   - Slurm architecture
   - Slurm terms
2. User and admin commands
   - Basics
   - User commands (srun, salloc, sbatch)
   - User and admin commands (sinfo, squeue, scancel, scontrol, sstat, sprio, sacctmgr)
3. Additional informations
   - Commands format
   - Slurm environment
   - Job arrays
   - Job dependencies
4. Slurm and MPI
   - Slurm MPI integration
   - Running MPI jobs
   - Running hybrid MPI OpenMP jobs

Introduction

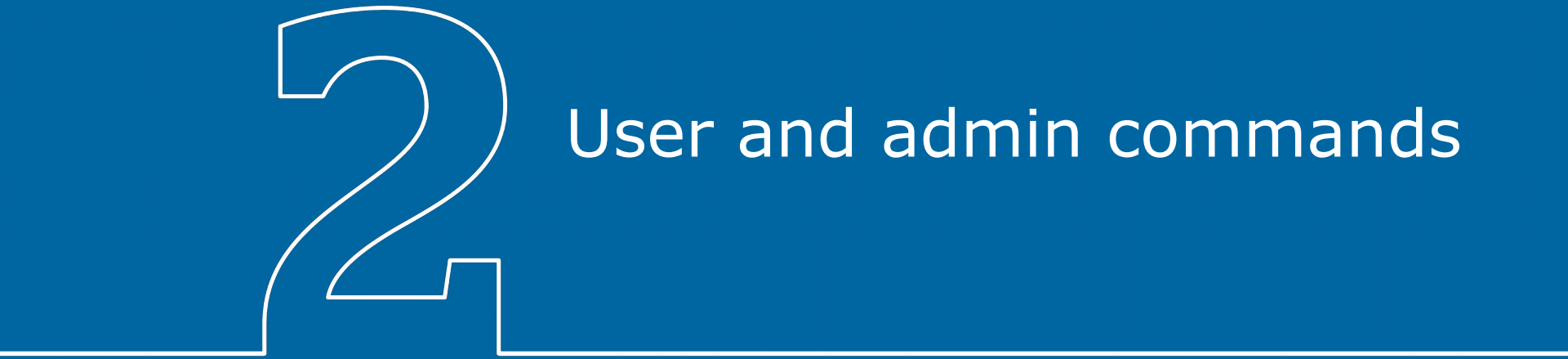
Introduction
SLURM architecture

- One central controller daemon slurmctld
- A daemon upon each computing node slurmd
- One central daemon for the database controls slurmdbd



Introduction
SLURM terms

- Computing node : computer used for the execution of programs
- Partition : group of nodes with specific characteristics (job limit, access controls, etc)
- Job : allocation of ressources assignet to a user for some time
- Step : sets of (possible parallel) tasks within a job



User and admin commands

Basics

sinfo, sbatch, scancel

- Where can I launch ? sinfo command displays resource usage and availability information for parallel jobs

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
workq*       up 4-00:00:00     46   idle node[101-110,112-147]
workq*       up 4-00:00:00      2   down node[111,148]
interq       up   infinite      1    mix genoview
smpq         up   infinite      1   idle genosmp02
```

- How can I launch a job script ? sbatch command

```
> sbatch HELLO_WORLD.sub
Submitted batch job 6595
```

- How to cancel a job ? scancel command

```
> scancel 6595
```

User commands

srun,salloc,sbatch

Command	Description
srun	used to submit a job for execution or initiate job steps in real time (option -n number of core, -N number of node, --time for time limit, etc). If necessary, srun will first create resource allocation in wich to run the parallel job
salloc	allocate resources (nodes, tasks, partition, etc), either run a command or start a shell. Request launch srun from shell (interactive commands within one allocation)
sbatch	allocate resources (nodes, tasks, partition, etc.) Launch a script containing sruns for series of steps

User commands

srun,salloc,sbatch

- All request an allocation of resources
- Similar set of command line options
- Request number of nodes, tasks, cpus, constraints, user info, dependencies, and lots more
- srun launches tasks (command) in parallel on the requested nodes
- salloc obtain a slurm job allocation. It can launch a task such as mpirun on the client, or open a shell on the client. srun is then used to launch tasks within the allocation
- sbatch is a shell script that contains multiple sruns within the allocation (multistep job)
- Command line options are preponderant
- The default is one task per node (unless -n or --cpus-per-task is used)

User commands

Sample srun

- srun launches a job that allocates resources (number of nodes, tasks, etc.) and is executed on each allocated cpu. Some basic parameters for srun command

```
> srun -l -J hostname --time=00:10:00 -N 2 --mem=40G -p workq --exclusive hostname
1: node102
0: node101
```

- -l : prepand task number to output
- -J : job name
- --time= : allocation time limit (format is days-hours:minutes:seconds)
- -p workq : specify the partition to use
- -N 2 : nomber of nodes required
- --mem : memory required per node
- --exclusive : exclusive acces to nodes (default is shared ressources)
- hostname : command to run

User commands
sbatch

- submit a batch script to slurm
- the batch script may contain options preceed with « #SBATCH » before any executable call
- assigned a jobId when script is successfully transferred to the slurm controler
- when job alloction is finally granted, Slurm runs a single copy of the batch script on the first node in the set of allocated nodes
- default stdout and stderr are directed to a file *slurm-%j.out* (can be modified with options -e and -o). %j is replaced by the jobID
- the script can be written in multiples languages like bash, ksh, python, etc

User commands
Sample sbatch

- Same as previous srun in sbatch script HOSTNAME.sub

```
#!/bin/bash
#SBATCH –time=00:10:00   # job time limit
#SBATCH -J hostname # job name
#SBATCH -N 2        # number of nodes
#SBATCH -p workq         # partition to use
#SBATCH --exclusive # exclusive acces to nodes
srun -l hostname         # submit parallel command
```

- Options submitted on command line are preponderant

```
> sbatch -J toto HOSTNAME.sub
> squeue -a
Submitted batch job 6604
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              6604     workq     toto     root  R       0:00      2 node[101-102]
```

User commands
salloc

- salloc is used to allocate resources for a job in real time.
- Typically this is used to allocate resources and spawn a shell
- The shell be used to execute srun commands
- Basic parameters similar with srun and sbatch

```
> salloc -N 2
salloc: Granted job allocation 6612
salloc: Waiting for resource configuration
salloc: Nodes node[101-102] are ready for job

> echo $SLURM_JOB_NODELIST
node[101-102]
```

alloc 1 task no GPU node

```
> salloc -n 1 --constraint=K40 -p interq
salloc: Granted job allocation 6999
salloc: Waiting for resource configuration
salloc: Nodes genoview are ready for job
```

User and admin commands

Command	Description
sinfo	display characteristics of nodes, partitions, reservations…
squeue	display jobs and their state
scancel	cancel a job or set of jobs
scontrol	administrative tool used to view and/or modify SLURM state. But can also be used to get information on jobs, partitions, reservations …
sstat	show status of running jobs
sprio	view the factors that comprise a job’s scheduling priority
sacctmgr	setup accounts, specify limitations on users and groups

User and admin commands

sinfo

- sinfo display information about Slurm nodes and partitions

```
> sinfo -n node[108-110]
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
workq*       up 4-00:00:00      3   idle node[108-110]
unlimitq     up   infinite      3   idle node[108-110]
wflowq       up   infinite      3   idle node[108-110]
interq       up 1-00:00:00      0    n/a
smpq         up   infinite      0    n/a
```

- list reasons nodes are in the down, drained or fail state

```
> sinfo  -Rl
Mon Nov  6 14:42:53 2017
REASON               USER         TIMESTAMP           STATE  NODELIST
Node unexpectedly re root(0)      2017-10-16T16:30:58 down   node111
```

- list by state :

```
> sinfo -t ALLOC
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
workq*       up 4-00:00:00      3  alloc node[114,126,135]
unlimitq     up   infinite      3  alloc node[114,126,135]
wflowq       up   infinite      3  alloc node[114,126,135]
```

User and admin commands

squeue

- squeue display jobs and their state. Basic parameters for squeue command :
  - -a : display info about all jobs and partitions
  - -j <job_list> : report more info about a particular job or jobs
  - -u <user> : report job information for a specific user
  - -i <seconds> : repeatedly gather and report thre requested information
  - --start : expected start time of pending job (if backfill scheduling plugin is used)

```
> squeue -a
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
              6612     workq     bash     root  R       7:56      2 node[101-
102]
              6542    interq TurboVNC dgorecki  R 1-06:19:31      1 genoview
```

User and admin commands

scancel

- scancel is used to signal jobs or job steps. Usage :

```
> scancel <jobID> <stepID>
```

- Usefull options
  - -n : restrict cancel to jobs with this job name
  - -p : restrict cancel to jobs in this partition
  - -t : restrict cancel to jobs in this state
  - -s : send signal to job or step
  - -u : restrict cancel to jobs owned by this user
  - -w : cancel any job using any of the given hosts

```
> scancel -p workq -u myuser -t PD
```

User and admin commands

scontrol

- scontrol is a tool used to view and modify SLURM configuration state
- can be used to get information on configuration, jobs, nodes, partitions, reservations …
  - scontrol show config
  - scontrol show job <jobId>
  - scontrol show node <node>
  - scontrol show partition <partition>
  - scontrol show reservation <res>

```
> scontrol show job 6924
JobId=6924 JobName=xhpl
   UserId=dgorecki(13549) GroupId=BULL(3000) MCS_label=N/A
   Priority=1 Nice=0 Account=bull QOS=others
   JobState=RUNNING Reason=None Dependency=(null)


   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   RunTime=00:24:54 TimeLimit=03:00:00 TimeMin=N/A
   SubmitTime=2017-11-02T17:30:23 EligibleTime=2017-11-02T17:30:23
   StartTime=2017-11-06T14:35:23 EndTime=2017-11-06T17:35:23 Deadline=N/A
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
```

User and admin commands

sprio

- use to view the components of a job's scheduling priority when multi-factor priority plugin is installed
- return
- the PRIORITY column report the global priority of job (highest is prior)
- by default, returns information for all pending jobs

```
sprio -u dgorecki
          JOBID     USER   PRIORITY
           6930 dgorecki          1
           6931 dgorecki          1
           6933 dgorecki          1
```

```
scontrol show config | grep Priority
PriorityParameters      = (null)
PriorityDecayHalfLife   = 7-00:00:00
```

User and admin commands

QOS, sacctmgr

- QOS (Quality of Services) are used in SLURM for grouping limitations and priorities
- Show QOS list

  > sacctmgr show qos

- Show usable QOS by user

```
> sacctmgr show assoc user=dgorecki
   Cluster    Account       User  Partition     Share GrpJobs       GrpTRES GrpSubmit     GrpWall   GrpTRESMins MaxJobs
MaxTRES MaxTRESPerNode M
axSubmit     MaxWall    MaxTRESMins                  QOS   Def QOS GrpTRESRunMin
---------- ---------- ---------- ---------- --------- ------- ------------- --------- ----------- ------------- -------
-------------- -
-------- ----------- ------------- -------------------- --------- -------------
  genobull       bull    dgorecki    unlimitq         1
    2500              cpu=6000000       others_unlimit others_u+
  genobull       bull    dgorecki                   1
    2500              cpu=6000000   contributors,others     others
```

Additional informations

Commands format

- some commands like sacct and squeue give the possibility to tune output format

```
sacct -D --format=jobid%-13,user%-15,uid,jobname%-15,state
%20,exitcode,Derivedexitcode,nodelist% -X –job 6969
```

```
        JobID            User    UID         JobName                State ExitCode DerivedExitCode        NodeList
------------- --------------- ------ --------------- -------------------- -------- -------------- ---------------
6969          root                 0 toto                         COMPLETED      0:0              0:0   node[101-102]
```

```
squeue --format="%10i %12u %12j %.8M %.8l %.10Q %10P %10q %10r %11v %12T %D %R" -S "T"
```

```
JOBID      USER         NAME             TIME TIME_LIM   PRIORITY PARTITION  QOS        REASON     RESERVATION STATE        NODES
NODELIST(REASON)
6612       root         bash            16:09 4-00:00:00          1 workq      normal     None       (null)      RUNNING      2 node[101-
102]
6542       dgorecki     TurboVNC     1-06:27:44 UNLIMITE          1 interq     normal     None       (null)      RUNNING      1 genoview
```

Slurm job environment

evironment variables

Environment variable	Correspondence
SLURM_JOBID	Job ID
SLURM_NNODES	#SBATCH -N
SLURM_NODELIST	Nodelist which is allocated to the job
SLURM_NTASKS	#SBATCH -n
SLURM_NTASKS_PER_NODE	#SBATCH --tasks-per-node
SLURM_CPUS_PER_TASK	#SBATCH -c
SLURM_SUBMIT_DIR	Job submission directory

https://slurm.schedmd.com/sbatch.html

Slurm job environment

evironment variables

- --export option identify which environment variables are propagated to the batch job
  - --export=ALL : all current shell variables are propagated
  - --export=NONE : no variable propagated
  - --export=VARIABLE=value : propagate the current variable

```
srun --export=LOGTYPE=debug,LOGFILE=log.out ./program
```

Job arrays
sbatch

- sbatch –a | --array=<indexes>
  - submit a job array, multiple jobs to be executed with identical parameters
  - multiple values may be specified using a comma separated list and/or a range of values with a "-" separator
    - --array=1-10
    - --array=0,6,16-32
    - --array=0-15:4 : a step of 4
    - --array=1-10%2 : a maximum of 2 simultaneously running tasks

Variable	Correspondance
SLURM_ARRAY_TASK_ID	Job array ID (index) number
SLURM_ARRAY_JOB_ID	Job array's master job ID number
SLURM_ARRAY_TASK_MAX	Job array's maximum ID (index) number
SLURM_ARRAY_TASK_MIN	Job array's minimum ID (index) number
SLURM_ARRAY_TASK_COUNT	total number of tasks in a job array

Job dependencies
sbatch

- sbatch –d | --dependency=<dependency_list>
  - defer the start of this job until the specified dependencies have been satisfied completed
  - <dependency_list> is of the form <type:jobId[:jobID][,type:jobID[:jobID]]>, example : `sbatch --dependency=afterok:6265 HELLO.job`

Type	Correspondance
after	this job can begin execution after the specified jobs have begun execution
afterany	this job can begin execution after the specified jobs have terminated
afterok	This job can begin execution after the specified jobs have successfully executed (ran to completion with an exit code of zero)
afternotok	This job can begin execution after the specified jobs have terminated in some failed state (non-zero exit code, node failure, timed out, etc)
singleton	This job can begin execution after any previously launched jobs sharing the same job name and user have terminated

SLURM and MPI

SLURM MPI integration

- MPI use with slurm depends upon the type of MPI being used. There are three fundamentally different modes of operation used by these various MPI implementations :
  - Slurm directly launches the tasks and performs initialization of communications (use of PMI library)
  - Slurm creates a resource allocation for the job and then mpirun launches tasks using Slurm's infrastructure
  - Slurm creates a resource allocation for the job and then mpirun launches tasks using some mechanism other than Slurm, such as SSH or RSH (outside slurm's monitoring or control)
- PMI library:
  - The use of a PMI library offer tight integration with slurm and the simplest way to launch an MPI application.
  - The PMI library has been used for quite some time as a means of exchanging information needed for interprocess communication.

Running MPI jobs

srun

- OpenMPI is configured with pmi2 support (compiled with --with-pmi=). The OMPI jobs can be launched directly using the srun command.

```
> module load compiler/intel-2018.0.128 mpi/openmpi-1.8.8-intel2018.0.128
> mpicc -o hello_world hello_world.c
> srun -n 12 -N 2 --ntasks-per-node=6 hello_world
Hello world from process 4 of 12 - node101
Hello world from process 5 of 12 - node101
Hello world from process 3 of 12 - node101
Hello world from process 0 of 12 - node101
Hello world from process 2 of 12 - node101
Hello world from process 9 of 12 - node102
Hello world from process 6 of 12 - node102
Hello world from process 7 of 12 - node102
Hello world from process 10 of 12 - node102
Hello world from process 8 of 12 - node102
Hello world from process 1 of 12 - node101
Hello world from process 11 of 12 - node102
```

Running MPI jobs

mpirun

- But it is also possible to launch an MPI application through the common mpirun command. Example of a full bash script :

```
#!/bin/bash
#SBATCH -J mpi_job
#SBATCH --nodes=2
#SBATCH --tasks-per-node=6
#SBATCH --time=00:10:00
cd $SLURM_SUBMIT_DIR
module purge
module load compiler/intel-2018.0.128 mpi/openmpi-1.8.8-intel2018.0.128
mpirun -n $SLURM_NTASKS -npernode $SLURM_NTASKS_PER_NODE ./hello_world
```

Running hybrid MPI OpenMP jobs

srun

- You can use --cpus-per-task in order to set the number of OpenMP threads

```
#!/bin/bash
#SBATCH -J hybrid_mpi_openmp_job
#SBATCH --nodes=2
#SBATCH --tasks-per-node=4
#SBATCH --cpus-per-task=8
cd $SLURM_SUBMIT_DIR
module purge
module load compiler/intel-2018.0.128 mpi/openmpi-1.8.8-intel2018.0.128
[ -n "$SLURM_CPUS_PER_TASK" ] && export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
srun ./hybrid_program
```

Thanks

For more information please contact:
M+ 33 6 14655608
cedric.trivino@atos.net

Bull
atos technologies