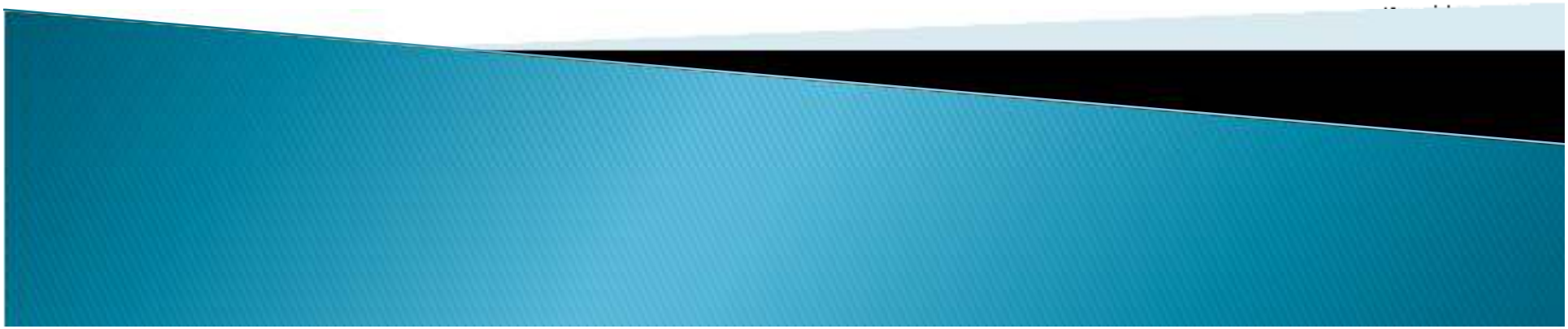
SLURM :– Resource Manager & Scheduler



Outline

- Introduction
- Commands & Running Jobs
- Configuration
- Scheduling
- Accounting

Simple Linux Utility for Resource Management

- Slurm is a resource manager. As there are a lot of resource within a cluster like : CPU-Cores, Memory banks, GPU accelerator cards managing which becomes a tedious task for a user and a system administrator.

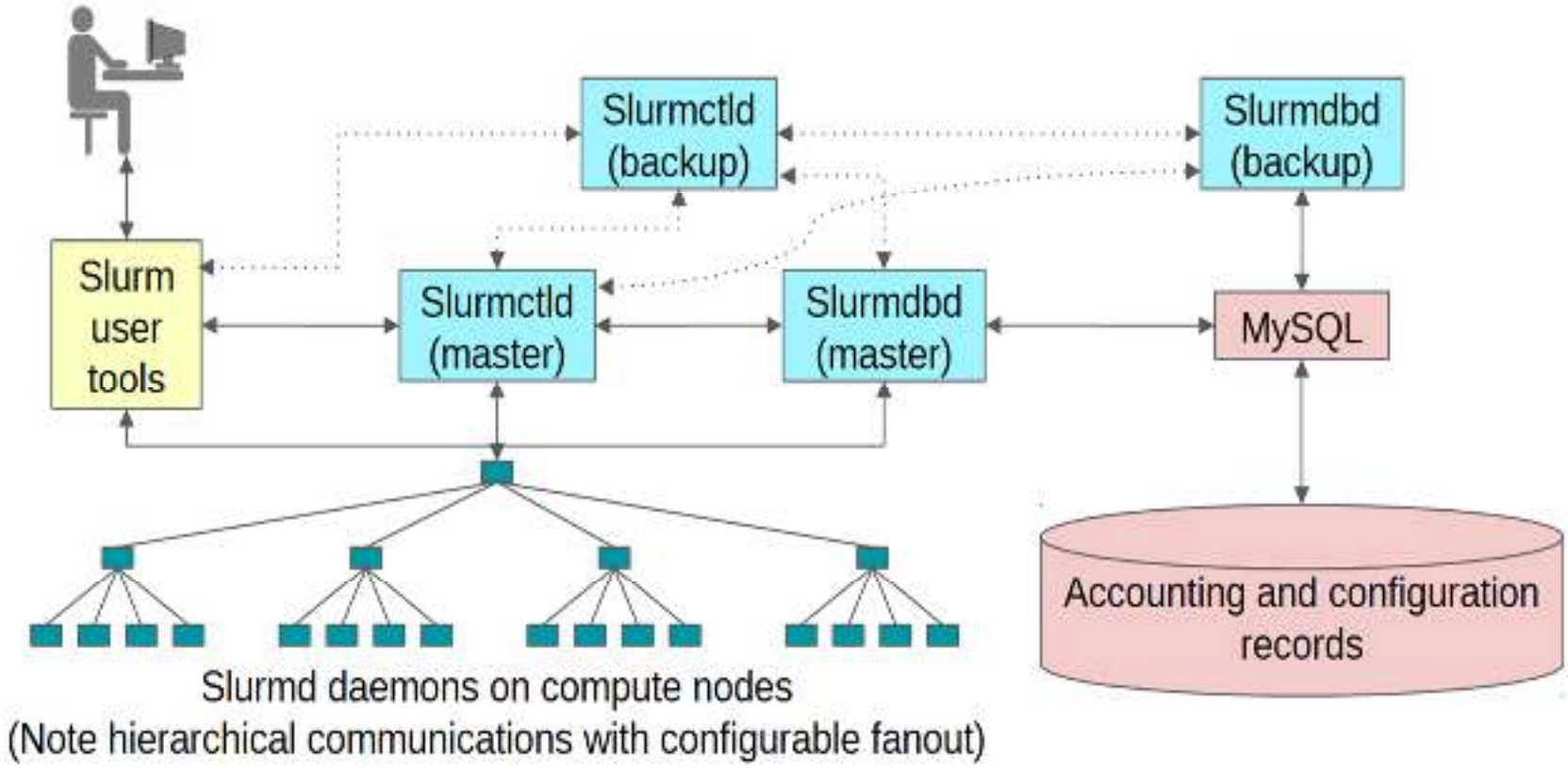
- Resource manager with in “slurm” tool helps to manage and represent resources to the users in a simplest way.

- Slurm will also function as a Job schedular.
- A scheduler checks the available resources within a cluster and manages which jobs run where and when.
- Allocating resources to each users for optimal utilization of system resources.
- Provides multiple algorithm, which provides different ways to initiate jobs on the resources.

SLURM Principles

- **Architecture Design:**
- One central controller daemon (**slurmctld**) on a management node.
- A daemon on each computing node (**slurmd**).
- One central daemon for the accounting database (**slurmdbd**)
- SLURM may be aware of network topology and use it in node selection.

SLURM Architecture



SLURM Principles …

- **Principal Concepts:**
- A general purpose **plug-in mechanism**(provides different behavior for features such as scheduling policies, process tracking, etc).
- **Partitions** represent group of nodes with specific characteristics (similar resources, priority, job limits, access controls, etc).
- **Job steps** which are sets of tasks within a job.

Basic CPU Management Steps SLURM

- SLURM uses four basic steps to manage CPU resources for a job/step:
- Selection of Nodes
- Allocation of CPUs from Selected Nodes
- Distribution of Tasks to Selected Nodes
- Optional Distribution and Binding of Tasks to Allocated CPUs within a Node (Task Affinity)

User & Admin Commands

- **sinfo** display characteristics of partitions
- **squeue** display jobs and their state
- **scancel** cancel a job or set of jobs.
- **scontrol** display and changes characteristics of jobs, nodes, partitions.
- **sstat** how status of running jobs.
- **sview** graphical view of cluster. Display and change characteristics of jobs, nodes, partitions.

sinfo

- Displays node and partition information
- Options permit you to filter, sort, and output information in almost any way desired
- [user@n260 ~]$ **sinfo**

```
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
    lsf        up      infinite    2       drain* n[100,110]
    lsf        up      infinite    224     alloc  n[1-96,111-238]
    lsf        up      infinite    14       Idle  n[97-99,101-109,239-240]
    devel*     up      60:00       12       alloc n[241-252]
    devel*     up      60:00       4        down* n[253-256]
```

squeue

- Displays job and job step information
- Options permit you to filter, sort, and output information in almost any way desired.

```
[user@n260 ~]$ squeue
JOBID  PARTITION  NAME USER ST TIME NODES NODELIST
16000    lsf          xc1@37 alice R 6:46:04 96 n[1-96]
16306    lsf            xc1@37 brian R 4:03:53 128 n[111-238]
16721 devel        fall cheryl R 20:07 8 n[241-248]
16745 devel       winter david R 6:40 4 n[249-252]
16752 devel       season edith PD 0:00  6
```

squeue – Job Step Example

```
[user@n260 ~]$ squeue -s
STEPID PARTITION USER TIME NODELIST
16000.0 lsf      alice 6:48:04 n1
16000.1 lsf      alice 6:48:03 n[1-96]
16306.0 lsf      brian 4:05:54 n111
16306.1 lsf      brian 4:05:53 n[111-238]
16721.0          devel cheryl 22:07 n[241-248]
16721.1          devel cheryl 22:06 n[241-248]
16721.2          devel cheryl 22:05 n[241-248]
16745.0          devel david 8:40 n[249-252]
```

scancel

- Send specified signal to a job and/or job step.
- By default, sends SIGKILL to terminate job.
- Filters can be used to specify user, program
- name, partition, job state, etc.

```
[user@n16 ~]$ scancel 12345
[root@n16 root]# scancel --interactive --user=brian
Cancel job id=13601 name=summer partition=pdebug [y/n]? y
Cancel job id=13777 name=NewJob partition=pdebug [y/n]? n
```

scontrol

- Administrative tool to set and get configuration information
  - Can be useful to users who want to see full state information
  - without fancy filtering or formatting

```
[root@n16 root]# scontrol ping
Slurmctld(primary/backup) at n11/n12 are UP/UP
[root@n16 root]# scontrol show partition pdebug
PartitionName=pdebug TotalNodes=64 TotalCPUs=128 RootOnly=NO
Default=NO Shared=NO State=UP MaxTime=30
MinNodes=1 MaxNodes=UNLIMITED AllowGroups=(null)
Nodes=xc[40-103] NodeIndecies=0,63,-1
[root@n16 root]# scontrol show job 70573
JobId=70573 UserId=david(789) Name=winter JobState=RUNNING
Priority=4294895192 Partition=pdebug BatchFlag=0
AllocNode:Sid=mcr39:4277 TimeLimit=30
StartTime=02/03-14:00:49 EndTime=02/03-14:30:49
NodeList=xc[64-79] NodeListIndecies=64,79,-1
ReqProcs=0 MinNodes=0 Shared=0 Contiguous=0
MinProcs=0 MinMemory=0 Features=(null) MinTmpDisk=0
ReqNodeList=(null) ReqNodeListIndecies=-1
```

Configuration

slurm.conf

- Management policies
- Scheduling policies
- Allocation policies
- Node definition
- Partition definition
- Present on controller and all compute nodes

slurmdbd.conf

- Type of persistent storage (DB)
- Location of storage
- Admin choices

topology.conf

- Switch hierarchy

Others:

- plugstack.conf, gres.conf, cgroup.conf, ...

Configuration (slurm.conf)

- **Management Policies:**
- Location of controllers, backups, logs, state info
- Authentication
- Cryptographic tool
- Accounting
- Logging
- Process tracking

Configuration (slurm.conf) ...

```
# Sample configf or SLURM Users Group
# Management Policies
ClusterName=rod
ControlMachine=sulu
SlurmUser=slurm
SlurmctldPort=7012
SlurmdPort=7013
AuthType=auth/munge
CryptoType=crypto/munge
# Location of logs and state info
StateSaveLocation=/app/slurm/rbs/tmp_slurm/rbs-slurm/tmp
SlurmdSpoolDir=/app/slurm/rbs/tmp_slurm/rbs-slurm/tmp/slurmd.%n.spool
SlurmctldPidFile=/app/slurm/rbs/tmp_slurm/rbs-slurm/var/run/slurmctld.pid
SlurmdPidFile=/app/slurm/rbs/tmp_slurm/rbs-slurm/var/run/slurmd.%n.pid
SlurmctldLogFile=/app/slurm/rbs/tmp_slurm/rbs-slurm/slurmctld.log
SlurmdLogFile=/app/slurm/rbs/tmp_slurm/rbs-slurm/slurmd.%n.log.%h
# Accounting
AccountingStorageType=accounting_storage/slurmdbd
AccountingStorageEnforce=limits
AccountingStorageLoc=slurm3_db
AccountingStoragePort=8513
AccountingStorageHost=sulu
```

Configuration (slurm.conf) ...

- **Scheduling policies**
- Priority
- Preemption
- Backfill

 # Scheduling policies SchedulerType=sched/builtin FastSchedule=1
PreemptType=preempt/partition_prio Preempt Mode=GANG,SUSPEND

Configuration (slurm.conf)

- **Allocation policies**
- Entire nodes or 'consumable resources'
- Task Affinity (lock task on CPU)
- Topology (minimum number of switches)

Allocaton Policies

SelectType=select/cons_res

SelectTypeParameters=CR_Core

TaskPlugin=task/cgroup

Configuration (slurm.conf)

- **Partition definition**
- Set of nodes
- Sharing
- Priority/preemption

 # Partition Definitions

PartitionName=all Nodes=trek[0-63] Shared=NO Default=YES
PartitionName=P2 Nodes=trek[0-63] Shared=NO Priority=2
PreemptMode=CANCEL

PartitionName=P3 Nodes=trek[0-63] Shared=NO Priority=3
PreemptMode=REQUEUE

PartitionName=P4 Nodes=trek[0-63] Priority=1000
AllowGroups=vip

PartitionName=MxThrdNodes=trek[32-63] Shared=NO

Why use multiple partitions

- Provide different capabilities for different groups of users.
- Provides multiple queue for priority (with different preemption behavior)
- Group machines with same features (hyperthreading)
- Provide sharing.

Thank You