Trackable RESources (TRES)

Brian Christiansen and Danny Auble
SchedMD LLC

Slurm User Group Meeting 2015
Overview

- Need
- Setup
- Transition
- Reporting
- Fairshare
- Priority
Need

● Limits on more resources other than CPU/Memory/Nodes
  ○ GRES, Licenses, etc

● Method for accounting what resources were really used
  ○ more than just cpu anyway

● Easier way to add more limits
  ○ No database alteration needed for future TRES
Setup

- All TRES are global and are defined in the slurm.conf
  - Available to all clusters
- AccountingStorageTRES
  - Used to define which TRES are to be tracked on the system. By default CPU, Energy, Memory, and Node are tracked. This will be the case whether specified or not.
- Example
  - AccountingStorageTRES=gres/gpu:tesla,license/iop1,bb/cray
Setup

- **PriorityWeightTRES**
  - A comma separated list of TRES Types and weights that sets the degree that each TRES Type contributes to the job's priority.
  - `PriorityWeightTRES=CPU=1000,Mem=2000,GRES/gpu=3000`

- **TRESBillingWeights**
  - For each partition this option is used to define the billing weights of each TRES type that will be used in calculating the usage of a job.
  - `TRESBillingWeights="CPU=1.0,Mem=0.25,GRES/gpu=2.0"`
Transition

- sacctmgr
  - [Grp|Max] [cpu|mem|node]* limits now [Grp|Max]TRES*
  - GrpTRES=cpu=500,mem=10000,nodes=100
    - (GrpCpus=500 GrpMem=10000 GrpNodes=100)
  - Old definitions still work, for legacy scripts
  - -1 still how to removing limits GrpTRES=cpu=-1,mem=-1,nodes=1000
Transition

- sacctmgr
  - New/Extended Association|QOS options (all work for any TRES)
    - GrpTRES
    - GrpTRESMins
    - GrpTRESRunMins
    - MaxTRESPerJob
    - MaxTRESPerNode
    - MaxTRESMinsPerJob
    - MaxTRESPerUser*
    - MinTRESPerJob*

*only applicable to QOS
Transition

- `scontrol/squeue/sacct`
  - Can display TRES as well
  - When a limit is violated the reason field in a job has a unique reason for each TRES type/limit combo
    - QOSGrpCpuLimit
    - QOSGrpMemLimit
    - AssocGrpCpuLimit
    - AssocGrpMemLimit
    - etc
Reporting

- Data is King!
- sreport
  - Previously would only report on CPU utilization
  - Now can report on **any** TRES (except Node)
## Reporting

- **Need more memory? Or less cpus?**

```
$ sreport -tminper cluster utilization --tres="cpu,mem" start=2015-09-01T00:00:00
```

---

**Cluster Utilization 2015-09-01T00:00:00 - 2015-09-01T23:59:59**

Use reported in TRES Minutes/Percentage of Total

<table>
<thead>
<tr>
<th>Cluster</th>
<th>TRES Name</th>
<th>Allocated</th>
<th>Down</th>
<th>PLND Down</th>
<th>Reserved</th>
<th>Idle</th>
<th>Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>compy</td>
<td>cpu</td>
<td>253440(20.00%)</td>
<td>0(0.00%)</td>
<td>0(0.00%)</td>
<td>1013760(80.00%)</td>
<td>1267200(100.00%)</td>
<td></td>
</tr>
<tr>
<td>compy</td>
<td>mem</td>
<td>4582306080(90.00%)</td>
<td>0(0.00%)</td>
<td>0(0.00%)</td>
<td>509145120(10.00%)</td>
<td>5091451200(100.00%)</td>
<td></td>
</tr>
</tbody>
</table>
Reporting

- GPUs being used?

```bash
$ sreport -tminper cluster utilization --tres="cpu,gres/gpu" start=2015-09-02T00:00:00

Cluster Utilization 2015-09-02T00:00 - 2015-09-02T23:59:59
Use reported in TRES Minutes/Percentage of Total

<table>
<thead>
<tr>
<th>Cluster</th>
<th>TRES Name</th>
<th>Allocated</th>
<th>Down</th>
<th>PLND Down</th>
<th>Reserved</th>
<th>Idle</th>
<th>Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>compy</td>
<td>cpu</td>
<td>1140480(90.00%)</td>
<td>0(0.00%)</td>
<td>0(0.00%)</td>
<td>126720(10.00%)</td>
<td>0(0.00%)</td>
<td>1267200(100.00%)</td>
</tr>
<tr>
<td>compy</td>
<td>gres/gpu</td>
<td>63360(20.00%)</td>
<td>0(0.00%)</td>
<td>0(0.00%)</td>
<td>253440(80.00%)</td>
<td>0(0.00%)</td>
<td>316800(100.00%)</td>
</tr>
</tbody>
</table>
```
## Reporting

- **Which GPUs are being used most?**

```
$ sreport -tminper cluster utilization --tres="gres/gpu:k40,gres/gpu:k80" start=2015-09-02T00:00:00
```

---

Cluster Utilization 2015-09-02T00:00:00 - 2015-09-02T23:59:59
Use reported in TRES Minutes/Percentage of Total

<table>
<thead>
<tr>
<th>Cluster</th>
<th>TRES Name</th>
<th>Allocated</th>
<th>Down</th>
<th>PLND Down</th>
<th>Reserved</th>
<th>Idle</th>
<th>Reported</th>
</tr>
</thead>
<tbody>
<tr>
<td>compy</td>
<td>gres/gpu:k40</td>
<td>63360 (20.00%)</td>
<td>0 (0.00%)</td>
<td>0 (0.00%)</td>
<td>0 (0.00%)</td>
<td>253440 (80.00%)</td>
<td>316800 (100.00%)</td>
</tr>
<tr>
<td>compy</td>
<td>gres/gpu:k80</td>
<td>190080 (60.00%)</td>
<td>0 (0.00%)</td>
<td>0 (0.00%)</td>
<td>0 (0.00%)</td>
<td>126720 (40.00%)</td>
<td>316800 (100.00%)</td>
</tr>
</tbody>
</table>
Fairshare

- Previously only total cpus was accounted for in fairshare
- If a job used 1 CPU and all the memory on the machine the job was only charged for 1 CPU when it really used the whole node
- Now, any TRES can be accounted for in fairshare
  - TRESBillingWeights
Fairshare

- TresBillingWeights configured per partition
- Billing weights are specified as a comma-separated list of <TRES Type>=<TRES Billing Weight> pairs
- TRESBillingWeights=CPU=1.0,Mem=0.25,GRES/gpu=2.0
  - Mem is weighted per gigabyte
- Two methods of calculating billable TRES
  - MAX_TRES
  - SUM of TRES
Fairshare

- **SUM of TRES**
  - Default
  - SUM(<TRES> + <TRES Weight>, …)
  - Good if you want to account for what you are using

- **MAX_TRES**
  - PriorityFlags=MAX_TRES
  - MAX(Node TRES) + SUM(Global TRES)
  - Good if you want to account if any one resource is blocking other jobs from running on a node
Fairshare

- TRESBillingWeights=CPU=1.0,Mem=0.25
- 16CPU, 64GB nodes

**SUM of TRES:**

<table>
<thead>
<tr>
<th></th>
<th>CPUs</th>
<th>Mem GB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job1</td>
<td>(1 <em>1.0) + (60</em>0.25) = (1 + 15) = 16</td>
<td></td>
</tr>
<tr>
<td>Job2</td>
<td>(16*1.0) + (1 *0.25) = (16+.25) = 16.25</td>
<td></td>
</tr>
<tr>
<td>Job3</td>
<td>(16<em>1.0) + (60</em>0.25) = (16+ 15) = 31</td>
<td></td>
</tr>
</tbody>
</table>

**MAX_TRES:**

<table>
<thead>
<tr>
<th></th>
<th>CPUs</th>
<th>Mem GB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job1</td>
<td>MAX((1 <em>1.0), (60</em>0.25)) = 15</td>
<td></td>
</tr>
<tr>
<td>Job2</td>
<td>MAX((15*1.0), (1 *0.25)) = 15</td>
<td></td>
</tr>
<tr>
<td>Job3</td>
<td>MAX((16<em>1.0), (64</em>0.25)) = 16</td>
<td></td>
</tr>
</tbody>
</table>
Priority

- PriorityWeightTRES
  - List of TRES Types and weights
  - PriorityWeightTRES=CPU=1000,Mem=2000,GRES/gpu=3000

- Control how much a TRES contributes to the job’s priority

- Node TRES (i.e. CPU, Mem, GRES, Node) are normalized against total TRES configured in a partition

- Global TRES (i.e. license, bb) are normalized against the global amount in the system
Priority

- Ex. If a partition has 80 cpus and a job uses 8, then the priority factor is .1 (or 10%)
- AccountingStorageTRES=cpu,mem,gres/gpu
- PriorityWeightTRES=cpu=1000,gres/gpu=3000

```
$ sprio
    JOBID  PRIORITY  AGE  FAIRSHARE   TRES
      3    625      0    500        cpu=125
      5    600      0    500        cpu=100
      6    812      0    500   cpu=12,gres/gpu=300
```
Questions?