● What is a TRES
● Job Usage
● Limits
● Priority
● Billing TRES
● Reporting
TRES

- Usage Tracking and Limits
  - Previously only cpu and energy were tracked and limited.

- Trackable RESource (TRES)
  - BB (burst buffers)
  - Billing
  - CPU
  - Energy
  - FS (filesystem)
  - GRES
  - IC (interconnect)
  - License
  - Mem (Memory)
  - Node
  - Pages
  - VMem (Virtual Memory/Size)
TRES

- Configuration
  - slurm.conf
    - AccountingStorageTRES=
  - Default: Billing, CPU, Energy, Memory, Node, FS/Disk, Pages and VMem
    - Can't unset
    - AccountingStorageTRES=gres/gpu:tesla,license/iop1,bb/cray
  - Slurmctld pushes TRES to Slurmdbd
    - sacctmgr show tres
<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>mem</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>energy</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>node</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>billing</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>fs</td>
<td>disk</td>
<td>6</td>
</tr>
<tr>
<td>vmem</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>pages</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>gres</td>
<td>gpus</td>
<td>1001</td>
</tr>
<tr>
<td>gres</td>
<td>gpus:tesla</td>
<td>1002</td>
</tr>
<tr>
<td>license</td>
<td>ansys</td>
<td>1003</td>
</tr>
<tr>
<td>gres</td>
<td>gpus:k80</td>
<td>1004</td>
</tr>
<tr>
<td>ic</td>
<td>ofed</td>
<td>1005</td>
</tr>
<tr>
<td>fs</td>
<td>lustre</td>
<td>1006</td>
</tr>
<tr>
<td>license</td>
<td>jobsize</td>
<td>1007</td>
</tr>
</tbody>
</table>
TRES: Job Usage

- **scontrol show jobs**
  - ReqTRES until running, AllocTRES after running
    - e.g. Requesting 1 cpu on a 2 threaded core with CR_CORE
  - Billing TRES not figured out until allocation

- **sacct -o**
  - ReqTRES
  - AllocTRES
  - TRESUsage<In|Out><Ave|Min|Max|Tot>
  - TRESUsage<In|Out><Min|Max>Node
  - TRESUsage<In|Out><Min|Max>Task
TRES: Limits
TRES: Limits

- **Association, User**
  - <Grp|Max>TRESMins
  - <Grp|Max>TRESRunMins
  - <Grp|Max>TRES

- **QOS**
  - GrpTRES
  - <Grp|Max>TRESMins
  - MaxTRESPerAccount
  - MaxTRESPerJob
  - MaxTRESPerNode
  - MaxTRESPerUser
  - MinTRES
TRES: Limits

- set/clear with sacctmgr
  - sacctmgr modify user bob set grptres=cpu=10,memory=200
  - sacctmgr modify user bob set grptres=cpu=-1,memory=-1

- Good to know
  - Limits stored in SlurmDBD
  - Usage tracked and stored (state files) in slurmd
  - Limits enforced by slurmd

- scontrol show assoc_mgr
  - [<users|qos|accounts>=<name1>[,...,<nameN>]]
  - flags=<users,assoc,qos>
$ scontrol show assoc users=brian accounts=stuff flags=assoc

ClusterName=lappy Account=stuff UserName= Partition= Priority=0 ID=3
  SharesRaw/Norm/Level/Factor=2147483647/0.00/5/0.00
  UsageRaw/Norm/Efctv=20612.12/1.00/1.00
  ParentAccount=root(1) Lft=86 DefAssoc=No
  GrpJobs=N(40) GrpJobsAccrue=N(29)
  GrpSubmitJobs=N(69) GrpWall=N(69.59)
  GrpTRES=cpu=500(80),mem=N(16000),energy=N(0),node=N(10),billing=N(160)
  GrpTRESMins=cpu=1000(171),mem=N(34350),energy=N(0),node=N(69),billing=N(343),...
  GrpTRESRunMins=cpu=N(80),mem=N(16000),energy=N(0),node=N(40),billing=N(160),...

ClusterName=lappy Account=stuff UserName=brian(1003) Partition= Priority=0 ID=4
  SharesRaw/Norm/Level/Factor=1/0.20/5/0.20
  UsageRaw/Norm/Efctv=20612.12/1.00/1.00
  ParentAccount= Lft=91 DefAssoc=Yes
  GrpJobs=N(40) GrpJobsAccrue=N(29)
  GrpSubmitJobs=N(69) GrpWall=N(69.59)
  GrpTRES=cpu=N(80),mem=N(16000),energy=N(0),node=N(10),billing=N(160),...
  GrpTRESMins=cpu=N(171),mem=N(34350),energy=N(0),node=N(69),billing=N(343),...
  GrpTRESRunMins=cpu=N(80),mem=N(16000),energy=N(0),node=N(40),billing=N(160),...

MaxJobs= MaxJobsAccrue= MaxSubmitJobs= MaxWallPJ=
TRES: Priority
TRES: Priority

- **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`

- By default, normalized against Partition's on-node resources (e.g. cpu, memory, gres) and against global resources (e.g. licenses, bb)

- In 19.05, `PriorityFlags=NO_NORMAL_TRES` was added to not normalize
  - NO_NORMAL_ALL
  - NO_NORMAL_ASSOC
  - NO_NORMAL_PART
  - NO_NORMAL_QOS
TRES: 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

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PRIORITY</th>
<th>AGE</th>
<th>FAIRSHARE</th>
<th>TRES</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>625</td>
<td>0</td>
<td>500</td>
<td>cpu=125</td>
</tr>
<tr>
<td>5</td>
<td>600</td>
<td>0</td>
<td>500</td>
<td>cpu=100</td>
</tr>
<tr>
<td>6</td>
<td>812</td>
<td>0</td>
<td>500</td>
<td>cpu=12, gres/gpu=75</td>
</tr>
</tbody>
</table>
TRES: Billing

- **Billing TRES**
  - Billing is a value that represents multiple TRES
  - Previously, only cpu was accounted for in fairshare.
    - Jobs only cpu usage even if used 1 cpu and all the memory on the node
  - Added as a TRES in 17.11
    - Limits and Usage

- **Calculated on a per-partition basis**
  - TRESBillingWeights="CPU=1.0,Mem=0.25G,GRES/gpu=2.0"

- **Two methods of calculating billable TRES**
  - MAX_TRES
  - SUM of TRES
TRES: Billing

- **SUM of TRES**
  - Default
  - SUM(<TRES> + <TRES Weight>, …)

- **MAX_TRES**
  - PriorityFlags=MAX_TRES
  - MAX(Node TRES) + SUM(Global TRES)
TRES: Billing

- 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>
The Billing TRES is calculated from a partition's TRESBillingWeights. Though TRES weights on a partition may be defined as doubles, the Billing TRES values for a job are stored as integers. This is not the case when calculating a job's fairshare where the value is treated as a double.
**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>0(0.00%)</td>
<td>1013760(80.00%)</td>
<td>1267200(100.00%)</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>0(0.00%)</td>
<td>5091451200(100.00%)</td>
</tr>
</tbody>
</table>
GPUs being used?

```
$ sreport -tminper cluster utilization --tres="cpu,gres/gpu" 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>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>0(0.00%)</td>
<td>253440(80.00%)</td>
<td>316800(100.00%)</td>
</tr>
</tbody>
</table>
```
Which GPUs are being used most?

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

<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>316800(100.00%)</td>
<td></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>316800(100.00%)</td>
<td></td>
</tr>
</tbody>
</table>
In sreport, the "Reported" Billing TRES is calculated from the largest Billing TRES of each node multiplied by the time frame. For example, if a node is part of multiple partitions and each has a different TRESBillingWeights defined the Billing TRES for the node will be the highest of the partitions. If TRESBillingWeights is not defined on any partition for a node then the Billing TRES will be equal to the number of CPUs on the node.
Questions?

- [https://slurm.schedmd.com/tres.html](https://slurm.schedmd.com/tres.html)