<table>
<thead>
<tr>
<th>Rank</th>
<th>Workload Manager</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Slurm</td>
<td>Tianhe-2</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>Titan</td>
</tr>
<tr>
<td>3</td>
<td>Slurm</td>
<td>Sequoia</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>K Computer</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>Mira</td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>Trinity</td>
</tr>
<tr>
<td>7</td>
<td>Slurm</td>
<td>Piz Daint</td>
</tr>
<tr>
<td>8</td>
<td>-</td>
<td>Hazel Hen</td>
</tr>
<tr>
<td>9</td>
<td>Slurm</td>
<td>Shaheen II</td>
</tr>
<tr>
<td>10</td>
<td>Slurm</td>
<td>Stampede</td>
</tr>
</tbody>
</table>
Data Warp Overview

- A cluster-wide high-performance storage resource
- Data Warp allocations are managed by Slurm
- Two types of allocations:
  - Persistent allocations used by multiple jobs or
  - Associated with a specific job
- Data Warp allocations can exist before, during and/or after a job is allocated compute resources
  - Used to stage-in data, scratch storage, and/or stage-out data
Data Warp Workflow

**Slurmctld Daemon**
- Job submission
  - Set job expected start time
  - Allocate compute resources as available after stage-in completes
  - Job execution complete
  - Job record purged after stage-out completes

**Data Warp Plugin**
- Validate Data Warp specification
  - Allocate available BB resources to jobs expected to start soonest
  - Stage-in data
  - Stage-out data
  - Release BB resources after stage-out completes
  - Revoke and release BB

---

Data Warp Plugin

---

Slurmctld Daemon
Trackable RESources (TRES)

Method for accounting what resources are really being used
  • What happens if I used one CPU and all of memory on a node?

Accounting & Limits on more resources other than just CPU
  • Data Warp, CPU, Energy, GRES, License, Memory and Node
  • 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.

Fair Share
  • TRES contributes to the job's priority and fair share calculations
Compute nodes can be allocated to multiple jobs, but restricted to a single user

- New partition configuration parameter “ExclusiveUser=yes”
- New job option “--exclusive=user”
- Used for security and higher system utilization
Power Management Overview

• Provides mechanism to cap a cluster's power consumption
• Starts by evenly distributing power cap across all nodes, periodically lowers the cap on nodes using less power and redistributes that power to other nodes
• Configuration options to control various thresholds and change rate options
• Example with 10 nodes and 1800 watts
Example: Time 0, Initial state

180 watts
180 watts
180 watts
180 watts
180 watts
180 watts
180 watts
180 watts
180 watts
Example: Time 60 seconds

- Node one is using 110 watts, others at 180 watts
- That 110 watt node is below the 90% lower_threshold
  - 180 watts x 90% = 162 watts
- Reduce cap
  - (Max watts - Min watts) x Decrease Rate% = reduce by watts
    - (200 watts - 100 watts) x 30% = 30 watts
  - (Current watt allocation – Current watts usage) / 2 = reduce by watts
    - (180 watts – 110 watts) / 2 = 35 watts
  - Use the lessor of these two calculations
    - Node's cap is reduced from 180 watts to 150 watts.
- We now have 1650 watts available to distribute over the remaining 9 nodes
  - 183 watts per node
Example: Time 60 seconds

150 watts
183 watts
183 watts
183 watts
183 watts
183 watts
183 watts
183 watts
Example: Time 120 seconds

- Node 1 is using 110 watts
  - Reduced power by the decrease_rate
  - Now at 130 watts
- Node 2 is using 115 watts
  - Reduced power by the decrease_rate
  - Now at 163 watts
- Eight nodes at 183 watts
  - Remaining 1517 watts
  - Evenly distribute to remaining 8 compute nodes
  - 189 watts per node
Example: Time 120 seconds

- 130 watts
- 163 watts
- 189 watts
- 189 watts
- 189 watts
- 189 watts
- 189 watts
- 189 watts
Example: Time 180 seconds

- Node 1 is now consuming 128 watts
  - Over upper_threshold of 98% power utilization
    - 130 watts x 98% = 127 watts
  - Increased cap by increase_rate
    - Node 1 power cap set at 140 watts

- Node 2 is allocated a new job
  - Set power cap to the same as other nodes consuming all available power

- Remaining 1660 watts evenly distributed across 9 nodes or 184 watts per node
Example: Time 180 seconds

- 140 watts
- 184 watts
- 184 watts
- 184 watts
- 184 watts
- 184 watts
- 184 watts
- 184 watts
- 184 watts

Graph showing power levels over time.
Slurm 16.05

- Knights Landing Integration
- Job stats email
- PMIX Integration
- Deadline scheduling
- Data Warp
  - Manage multiple file systems
- scontroltop
  - User can change job priorities
  - Admin can change any job priority
- Disable memory allocation on a per partition basis
- Manage node sharing by account
- Topology aware GPU Scheduling
• Federated Cluster
  - Enable multiple clusters to act as a single cluster
  - Each cluster runs its own scheduler
  - Users can submit jobs to the federation or a specific cluster
    • The Federated Cluster solution will balance the load between clusters
  - Mix Cray and non-Cray clusters
Email questions to

jacob@schedmd.com