Energy Accounting & External Sensors Plugins

Slurm 2013 User Group

Danny Auble, SchedMD
Thomas Cadeau, Bull
Yiannis Georgiou, Bull
Martin Perry, Bull
martin.perry@bull.com
Two new plugins added in Slurm versions 2.5 and 2.6.

The **Energy Accounting Plugin** collects energy consumption data generated in-band from hardware sensors.

The **External Sensors Plugin** collects energy and temperature data generated out-of-band by an external system manager such as Nagios, or external sensors such as wattmeters.

Initial versions of each plugin provide limited functionality; may be enhanced in the future to provide additional data types and more detailed data.

Future enhancements to Slurm will allow the use the energy and temperature data collected by these plugins for resource management (allocation and scheduling decisions).
Energy & Power

• Informally, the terms energy and power are often used interchangeably, but they have distinct technical definitions.

• **Energy** is a *quantity* that represents the capacity to perform work. The standard (SI) unit of energy is the **joule**.

• **Power** is the *rate* at which energy is consumed (transferred or converted). The standard unit of power is the **watt**. 
  
  1 watt = 1 joule/second.

• Electrical energy is often expressed in units of **kilowatt-hours** (kWh). 
  
  1 kWh = 1000 watts for 3600 seconds = 3.6 megajoules.
Energy Accounting Plugin - Purpose

Plugin Name: acct_gather_energy

Purpose: To collect energy consumption data for the following uses:

- Job/step accounting – Running and total energy consumption by a job or step.
- Job/step profiling – Profile of power use by a job/step over time, per node.
- Hardware monitoring – Instantaneous power and cumulative energy consumption for each node.
acct_gather_energy Plugin - Overview

• One of a new family of acct_gather plugins that collect resource usage data for accounting, profiling and monitoring.

• Loaded by slurmd on each compute node.

• Called by jobacct_gather plugin to collect energy consumption accounting data for jobs and steps.

• Called separately via RPC from the slurmctld background thread to collect energy consumption data for nodes.

• Calls acct_gather_profile plugin to provide energy data samples for profiling.
**acct_gather_energy Plugin – Data Reporting**

- For running jobs, energy accounting data is reported by `sstat`.

- If accounting database is configured, energy accounting data is included in accounting records and reported by `sacct` and `sreport` (version 13.12).

- If `acct_gather_profile` plugin is configured, energy profiling data is reported by the method specified by the profile plugin type.

- Energy consumption data for nodes is reported by `scontrol show node`.

- Cumulative/total energy consumption is reported in units of **joules**.
- Instantaneous rate of energy consumption (power) is reported in units of **watts**.
acct_gather_energy Plugin - Versions

• Two versions of `acct_gather_energy` plugin supported:

  **acct_gather_energy/rapl**
  • Energy consumption data is collected from hardware sensors using the Running Average Power Limit (RAPL) interface.
  • Requires Intel Sandy Bridge or later Intel CPU type.
  • Linux MSR module must be loaded.

  **acct_gather_energy/ipmi**
  • Energy consumption data is collected from the Baseboard Management Controller (BMC) using the Intelligent Platform Management Interface (IPMI) protocol.
  • IPMI is a message-based, hardware-level interface specification providing for in-band and out-of-band collection of platform data.
  • Requires BMC hardware and FreeIPMI version 1.2.1 or later.

• Plugin API is described in Slurm developer documentation:
  • [http://slurm.schedmd.com/acct_gather_energy_plugins.html](http://slurm.schedmd.com/acct_gather_energy_plugins.html)
acct_gather_energy Plugin - Configuration

• In `slurm.conf`

To configure plugin:

```
AcctGatherEnergyType=acct_gather_energy/rapl or
AcctGatherEnergyType=acct_gather_energy/ipmi
```

Frequency of node energy sampling controlled by:
```
AcctGatherNodeFreq=<seconds>
```
Default value is 0, which disables node energy sampling

Collection of energy accounting data for jobs/steps requires:
```
JobAcctGatherType=jobacct_gather/linux or
JobAcctGatherType=jobacct_gather/cgroup
```
Frequency of job accounting sampling controlled by:
```
JobAcctGatherFrequency=task=<seconds>
```
Default value is 30 seconds

• In `acct_gather.conf` (new config file), for `acct_gather_energy/ipmi` only:

```
EnergyIPMIFrequency
EnergyIPMICalcAdjustment
EnergyIPMIPowerSensor
EnergyIPMIUsername
EnergyIPMIPassword
```
acct_gather_energy Plugin – Major Limitations

• The granularity of IPMI and RAPL data is node. Therefore, energy accounting and profiling data is reliable only for jobs/steps using unshared whole node allocation (select/linear, --exclusive). Future enhancements may support finer granularity (socket, core) for acct_gather_energy/rapl.

• RAPL energy data includes CPU, DRAM and cache energy consumption only. IPMI energy data includes all energy consumption by each node.

• Poor precision of energy accounting measurements for short jobs with few samples (depends on configured values of JobAcctGatherFrequency and EnergyIPMIFrequency).

• Asynchronous IPMI calls to eliminate potential delays.
External Sensors Plugin - Purpose

**Plugin Name:** ext_sensors

**Purpose:** To collect environmental-type data from external sensors or sources for the following uses:

- Job/step accounting – Total energy consumption by a completed job or step (no energy data while job/step is running).

- Hardware monitoring – Instantaneous power and cumulative energy consumption for nodes; instantaneous temperature of nodes.

- Future work will add additional types of environmental data, such as energy and temperature data for network switches, cooling system, etc. Environmental data may be used for resource management.
ext_sensors Plugin - Overview

- Loaded by `slurmctld` on management node.

- Collects energy accounting data for jobs and steps independently of the `acct_gather` plugins.
  - Called by slurmctld request handler when step starts.
  - Called by slurmctld step manager when step completes.

- Since energy use by jobs/steps is measured only at completion (i.e., no sampling), does not support power profiling or energy reporting for running jobs/steps (sstat).

- Called separately from the `slurmctld background` thread to sample energy consumption and temperature data for nodes.
ext_sensors Plugin – Data Reporting

• If accounting database is configured, energy data is included in accounting records and reported by `sacct` and `sreport` (in version 13.12).

• Energy consumption data for nodes is reported by `scontrol show node`.

• Cumulative/total energy consumption reported in `joules`.
  • Instantaneous energy consumption rate (power) for nodes reported in `watts`.
  • Node temperature reported in `celsius`.
ext_sensors Plugin - Versions

• One version of ExtSensorsType plugin currently supported:
  
  • ext_sensors/rrd
    External sensors data is collected using RRD. RRDtool is GNU-licensed software that creates and manages a linear database used for sampling or logging. The database is populated with energy data using out-of-band IPMI collection.

• Plugin API is described in Slurm developer documentation:
  • http://slurm.schedmd.com/ext_sensorsplugins.html
ext_sensors Plugin - Configuration

- In `slurm.conf`

  To configure plugin:
  
  ```
  ExtSensorsType=ext_sensors/rrd
  ```

  Frequency of node energy sampling controlled by:
  
  ```
  ExtSensorsFreq=<seconds>
  ```
  Default value is 0, which disables node energy sampling

  Collection of energy accounting data for jobs/steps requires:
  
  ```
  JobAcctGatherType=jobacct_gather/linux or cgroup
  ```

- In `ext_sensors.conf` (new configuration file)

  ```
  JobData=energy Specify the data types to be collected by the plugin for jobs/steps.
  NodeData=energy|temp Specify the data types to be collected by the plugin for nodes.
  SwitchData=energy Specify the data types to be collected by the plugin for switches.
  ColdDoorData=temp Specify the data types to be collected by the plugin for cold doors.
  MinWatt=<number> Minimum recorded power consumption, in watts.
  MaxWatt=<number> Maximum recorded power consumption, in watts.
  MinTemp=<number> Minimum recorded temperature, in celsius.
  MaxTemp=<number> Maximum recorded temperature, in celsius.
  EnergyRRA=<name> Energy RRA name.
  TempRRA=<name> Temperature RRA name.
  EnergyPathRRD=<path> Pathname of energy RRD file.
  TempPathRRD=<path> Pathname of temperature RRD file.
  ```
ext_sensors Plugin – Major Limitations

• The granularity of RRD energy data is node. Therefore, energy accounting data is reliable only for jobs/steps using unshared whole node allocation (select/linear, --exclusive).

• Potential for inaccuracy due RRD energy sampling interval.
Plugin Configuration Cases

• For node energy monitoring:
  
  AcctGatherEnergyType=acct_gather_energy/ipmi or rapl
  AcctGatherNodeFreq=<seconds>
  or
  ExtSensorsType=ext_sensors/rrd
  ExtSensorsFreq=<seconds>

• For job/step energy accounting:
  
  JobAcctGatherType=jobacct_gather/linux or cgroup
  AcctGatherEnergyType=acct_gather_energy/ipmi or rapl
  JobAcctGatherFrequency=task=<seconds>
  or
  JobAcctGatherType=jobacct_gather/linux or cgroup
  ExtSensorsType=ext_sensors/rrd

• For job/step power profiling:
  
  AcctGatherEnergyType=acct_gather_energy/ipmi or rapl
  AcctGatherProfileType=acct_gather_profile/hdf5
  JobAcctGatherFrequency=energy=<seconds>

Use of the acct_gather_energy/ipmi or acct_gather_profile plugins requires acct_gather.conf.
Use of the ext_sensors plugin requires ext_sensors.conf.
Use of the jobacct_gather/cgroup plugin requires cgroup.conf.
Command line option acctg-freq may be used to override any value from JobAcctGatherFrequency.
Configuration and Use Examples
Example 1 – Node energy monitoring using acct_gather_energy/rapl

[sulu] (slurm) mnp> scontrol show config
...
AcctGatherEnergyType = acct_gather_energy/rapl
AcctGatherNodeFreq = 30 sec
...

[sulu] (slurm) mnp> scontrol show node n15
NodeName=n15 Arch=x86_64 CoresPerSocket=8
   CPUAlloc=0 CPUErr=0 CPUTot=32 CPULoad=0.00 Features=(null)
   Gres=(null)
   NodeAddr=drak.usrnd.lan NodeHostName=drak.usrnd.lan
   OS=Linux RealMemory=1 AllocMem=0 Sockets=4 Boards=1
   State=IDLE ThreadsPerCore=1 TmpDisk=0 Weight=1
   CurrentWatts=121 LowestJoules=69447 ConsumedJoules=8726863
   ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
Example 2 – Energy accounting using acct_gather_energy/rapl

```
[sulu] (slurm) mnp> scontrol show config
...
JobAcctGatherType     = jobacct_gather/linux
JobAcctGatherFrequency = task=10
AcctGatherEnergyType   = acct_gather_energy/rapl
AccountingStorageType  = accounting_storage/slurmd
...
```

```
[sulu] (slurm) mnp> srun test/memcputest 100 10000 &
[1] 20712
[sulu] (slurm) mnp> 100 Mb buffer allocated
```

```
[sulu] (slurm) mnp> squeue

  JOBID PARTITION     NAME     USER  ST       TIME  NODES NODELIST(REASON)
120  drak-only      memcpute  slurm  R       0:03      1  n15

[sulu] (slurm) mnp> sstat -j 120 -o ConsumedEnergy
ConsumedEnergy
--------------
   2149

[sulu] (slurm) mnp> sstat -j 120 -o ConsumedEnergy
ConsumedEnergy
--------------
   2452

[sulu] (slurm) mnp> sstat -j 120 -o ConsumedEnergy
ConsumedEnergy
--------------
   2720
```

```
[sulu] (slurm) mnp> Finished: j = 10001, c = 2990739969

[1]+  Done  srun test/memcputest 100 10000

[sulu] (slurm) mnp> sacct -j 120 -o ConsumedEnergy
ConsumedEnergy
--------------
   3422
```
Example 3 – Energy accounting using acct_gather_energy/ipmi

[root@cuzco108 bin]# scontrol show config
...
JobAcctGatherType = jobacct_gather/linux
JobAcctGatherFrequency = task=10
AcctGatherEnergyType = acct_gather_energy/ipmi
AccountingStorageType = accounting_storage/slurmdb
...
[root@cuzco108 bin]# cat /usr/local/slurm2.6/etc/acct_gather.conf

EnergyIPMIFrequency=10
#EnergyIPMICalcAdjustment=yes
EnergyIPMIPowerSensor=1280

[root@cuzco108 bin]# srun -w cuzco113 memcputest 100 10000 &
[1] 26138
[root@cuzco108 bin]# 100 Mb buffer allocated

[root@cuzco108 bin]# squeue
  JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
  101 exclusive memcpute root  R           0:04      1 cuzco113
[root@cuzco108 bin]# sstat -j 101 -o ConsumedEnergy
ConsumedEnergy
-------------
  570

[root@cuzco108 bin]# sstat -j 101 -o ConsumedEnergy
ConsumedEnergy
-------------
  1.74K
Example 3 – continued

[root@cuzco108 bin]# Finished: j = 10001, c = 2990739969

[1]+  Done  srun -w cuzco113 memcputest 100 10000
[root@cuzco108 bin]# sacct -j 101 -o ConsumedEnergy
ConsumedEnergy
-------------
  1.74K
Example 4 – Node energy and temperature monitoring using ext_sensors/rrd

```
[root@cuzco0 ~]# scontrol show config
...
ExtSensorsType = ext_sensors/rrd
ExtSensorsFreq = 10 sec
...

[root@cuzco108 slurm]# cat /usr/local/slurm2.6/etc/ext_sensors.conf
#
# External Sensors plugin configuration file
#
JobData=energy
NodeData=energy,temp

EnergyRRA=1
EnergyPathRRD=/BCM/data/metric/%n/Power_Consumption.rrd

TempRRA=1
TempPathRRD=/BCM/data/metric/%n/Temperature.rrd

MinWatt=4
MaxWatt=200

[root@cuzco0 ~]# scontrol show node cuzco109

NodeName=cuzco109 Arch=x86_64 CoresPerSocket=4
CPUAlloc=0 CPUErr=0 CPUSum=8 CPULoad=0.00 Features=(null)
Gres=(null)
NodeAddr=cuzco109 NodeHostName=cuzco109
OS=Linux RealMemory=24023 AllocMem=0 Sockets=2 Boards=1
State=IDLE ThreadsPerCore=1 TmpDisk=0 Weight=1
BootTime=2013-09-03T17:39:00 SlurmdStartTime=2013-09-10T22:58:10
CurrentWatts=0 LowestJoules=0 ConsumedJoules=0
ExtSensorsJoules=4200 ExtSensorsWatts=105 ExtSensorsTemp=66
```
Example 5 – Energy accounting comparison using ext_sensors/rrd and acct_gather_energy/ipmi

The accuracy/consistency of energy measurements may be inaccurate if the run time of the job is short and allows for only a few samples. This effect should be reduced for longer jobs.

The following example shows that the ext_sensors/rrd and acct_gather_energy/ipmi plugins produce very similar energy consumption results for a MPI benchmark job using 4 nodes and 32 CPUs, with a run time of ~9 minutes.
Example 5 – continued

acct_gather_energy/ipmi

[root@cuzco108 bin]# scontrol show config | grep acct_gather_energy
AcctGatherEnergyType = acct_gather_energy/ipmi

[root@cuzco108 bin]# srun -n32 --resv-ports ./cg.D.32 &

[root@cuzco108 bin]# squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
122 exclusive  cg.D.32     root  R       0:02      4  cuzco[109,111-113]

[root@cuzco108 bin]# sacct -o "JobID%5,JobName,AllocCPUS,NNodes%3,NodeList%22,State,Start,End,Elapsed,ConsumedEnergy%9"

<table>
<thead>
<tr>
<th>JobID</th>
<th>JobName</th>
<th>AllocCPUS</th>
<th>NNodes</th>
<th>NodeList</th>
<th>State</th>
<th>Start</th>
<th>End</th>
<th>Elapsed</th>
<th>ConsumedEnergy</th>
</tr>
</thead>
</table>

ext_sensors/rrd

[root@cuzco108 bin]# scontrol show config | grep ext_sensors
ExtSensorsType = ext_sensors/rrd

[root@cuzco108 bin]# srun -n32 --resv-ports ./cg.D.32 &

[root@cuzco108 bin]# squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
128 exclusive  cg.D.32     root  R       0:02      4  cuzco[109,111-113]

[root@cuzco108 bin]# sacct -o "JobID%5,JobName,AllocCPUS,NNodes%3,NodeList%22,State,Start,End,Elapsed,ConsumedEnergy%9"

<table>
<thead>
<tr>
<th>JobID</th>
<th>JobName</th>
<th>AllocCPUS</th>
<th>NNodes</th>
<th>NodeList</th>
<th>State</th>
<th>Start</th>
<th>End</th>
<th>Elapsed</th>
<th>ConsumedEnergy</th>
</tr>
</thead>
</table>
The following slides illustrate the basic data collection architecture for each plugin version
acct_gather_energy/ipmi - Accounting Data Collection Architecture

Function call/return
RPC calls

jobacct_gather plugin

 слurmd: request handler

request handler

RPC calls

Function call/return

_slot_ipmi_id_run

_get_joules_task

_get_data

_get_joules_task

_RPC calls

Function call/return

_jobacct_gather_p_poll_data

_RPC calls

Function call/return

_thread_ipmi_run

FreeIPMI

RPC calls

Function call/return

rpc_acct_gather_energy

REQUEST_ACCT_GATHER_ENERGY

RESPONSE_ACCT_GATHER_ENERGY

RPC calls

Function call/return

slurm_get_node_energy

RPC calls

Function call/return

slurm_api

RPC calls

Function call/return

slurm_get_node_energy

RPC calls

Function call/return

slurm_api

acct_gather_energy/ipmi - Node Data Collection Architecture

acct_gather_energy/ipmi plugin

- update_nodes_energy
- get_data
- _get_joules_task

_thread_ipmi_run

FreeIPMI

slurmd: request handler

- rpc_acct_gather_update

REQUEST_ACCT_GATHER_UPDATE
RESPONSE_ACCT_GATHER_UPDATE

slurmctld: controller

- update_nodes_acct_gather_data
- _slurmctld_background

_thread_ipmi_id_run

thread_ipmi_id_run
acct_gather_energy/rapl - Accounting Data Collection Architecture

jobacct_gather_plugin

jobacct_gather_p_poll_data

acct_gather_energy/rapl plugin

get_data

_get_joules_task

RAPL API
acct_gather_energy/rapl - Node Data Collection Architecture

acct_gather_energy/rapl plugin

- get_data
- update_nodes_energy
- _get_joules_task

slurmd: request handler

- rpc_acct_gather_update

REQUEST_ACCT_GATHER_UPDATE

slurmctld: controller

- update_nodes_acct_gather_data
- _slurmctld_background

RAPL API

ext_sensors/rrd - Accounting Data Collection Architecture

The RRD database provides time-based platform data. Energy accounting values are calculated from the start and end timestamps of jobs/steps.
ext_sensors/rrd - Node Data Collection Architecture

- ext_sensors/rrd plugin
  - update_component_data
  - _update_node_data

- slurmcld: controller
  - _slurmctld_background

- RRDTool