

Lenovo/LRZ DP01 & DP02



Felip Moll
felip.moll@schedmd.com
SchedMD, LLC

Requirements



- DP01 - New XCC Energy Plugin
- DP02 - Avoid user ability to modify `--cpu-freq` on job submission

DP01 - Outline



- Requirements
- Plugin design
- How it works
- Example
- Future work/improvements

DP01 - Requirements



- Lenovo + LRZ need a plugin which issues OEM RAW commands to Lenovo XCC IPMI driver
- 'Fast accurate power meter' provides better precision (samples @ 50hz)
- New firmware in Lenovo SD650 servers
- XCC interface is not supported by Freeipmi
- RAW Commands need to be used in place of standard sensor reading which are already in Slurm IPMI plugin

DP01 - Requirements

Single Energy Command

Command	Raw Command String
Get 1 energy reading (main command used by BSC & LRZ)	<p>0x3A 0x32 4 2 0 0 0</p> <p>Response (LSB is first for each field): a7 00 75 e8 77 01 e4 02 9d a2 dd 5a 3b 01</p> <p>Bytes 0:1 =index/location of FPGA FIFO. These 2 bytes are for debug (e.g. 00 a7 =FIFO index 167)</p> <p>Incremental energy reading since the energy counter was last reset: Bytes 2:5 =Joules (e.g. 01 77 e8 75 =24,635,509 J) Bytes 6:7 =mJ (e.g. 02 e4 =740mJ) Total energy =24,635,509.740 Joules</p> <p>Timestamp associated with energy reading: Bytes 8:11 =seconds (e.g. 5a dd a2 9d =1,524,474,525 sec) Bytes 12:13 =mS (e.g. 01 3b =315 mS) Total time =1,524,474,525.315 seconds</p>

- With the single energy command, a 10mS sample rate cannot be maintained when running the ipmi command out-of-band. This is due to the network overhead associated with the out-of-band command.
- For this command, in-band execution should be used when sampling at a 10mS rate. Running in-band under linux requires the openipmi driver & the ipmitool program.

DP01 - Design

- For a job, need to provide:
 - Minimum registered power
 - This will be the minimum of the highest watermarks over all the power measurements among the allocated nodes.
 - Maximum registered power
 - This will be the highest watermark over all the power measurements among the allocated nodes.
 - Average power
 - Average of the highest watermarks.
 - Current power
 - Current consumption of the job/task
 - Consumed energy
 - Total joules consumed of the job

DP01 - Design



- In Slurm, metrics are gathered for each STEP of a Job.
 - A step can contain multiple tasks (each process with a pid)
 - A step will be run in one node
- For this plugin (and energy based ones), shared nodes will not provide accurate results, ideally nodes must be exclusively used.
- Metrics aggregation for the entire Job are done at the end

DP01 - Design

- XCC provides energy measurements in Joules
- Slurm XCC Plugin will report power and energy to slurmd/slurmctld
- Same concept than current ipmi plugin
 - Multiple tasks on one node, report the same value for power
 - Will not sum joules twice if two tasks are in a node
- Usage of FreeIPMI library versus FreeIPMIMonitoring API
 - freeipmi-devel, same package than IPMI

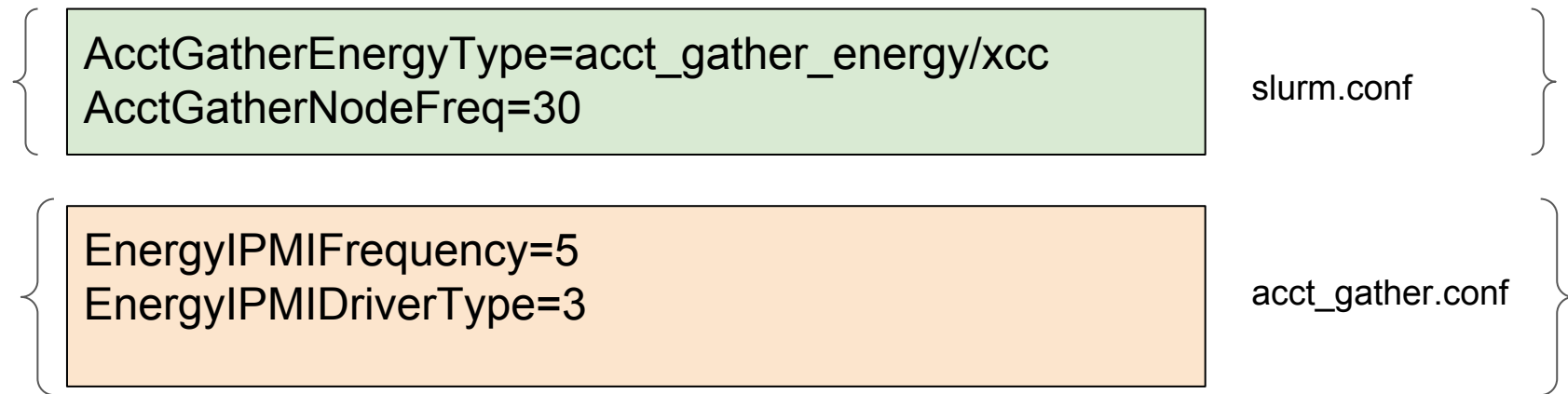
DP01 - Design

- No out-of-band communication, XCC requires in-band
- Code for version 18.08.5
- Usage of current TRES Fields:

TRESUsageInTot=energy=10089 <----- Energy Total
TRESUsageOutAve=energy=175 <----- Avg Power
TRESUsageOutMax=energy=180 <----- Max Power
TRESUsageOutMin=energy=138 <----- Min Power
TRESUsageOutTot=energy=175 <----- Curr Power

DP01 - Design

- This plugin can be enabled in slurm.conf
- Sample configuration



DP01 - How it works 1/5

- slurmd starts and initiates a new thread (i.e. 759076):
 - `_thread_update_node_energy()`

```
slurmd(759070) --{slurmd} (759076)
```

- This thread initializes an `acct_gather_energy_t` struct named *xcc_sensor*, shared among slurmd threads representing the recent measurements.

DP01 - How it works 2/5



- This thread will poll the ipmi every EnergyIPMIFrequency
 - And call: `_read_ipmi_values()` → `ipmi_raw_cmd()` [freeipmi library]
 - Then will calculate the current power
 - Given that we can calculate all the required information
 - Min/max/average power
 - Consumed joules
 - Poll time
 - This information is used as a 'cache', and consulted by others through RPCs to slurmd.

DP01 - How it works 3/5

- After the slurmd thread is created, a job can come in which will **fork** a new slurmstepd.
- slurmstepd needs to communicate with slurmd through API
 - `slurm_get_node_energy(NULL, delta, &sensor_cnt, &energy)`
- When the slurmstepd issues this RPC, slurmd will create a new thread to serve it.

```
slurmd(759070)-+-{slurmd}(759076) ← Initial thread, set global xcc_sensor
                `-{slurmd}(759071) ← New thread serving the RPC
                                   (needs to initialize a new ipmi_ctx due to
                                   freeipmi implementation), shares xcc_sensor
```

```
slurmstepd(14511)-+-sleep(14516) ←New fork(), doesn't share
                    |-{slurmstepd}(14514) xcc_sensor
                    `-{slurmstepd}(14515)
```

How it works 4/5

- The xcc_sensor data will be sent through the API.
 - An acct_gather_energy_t object is sent
 - A task maintains its total consumed energy, which is less than the total node consumed energy. Starts to count when first RPC is issued.
- An RPC may ask slurmd to update xcc_sensor info. depending on a delta factor → ENERGY_DATA_JOULES_TASK vs ENERGY_DATA_STRUCT

How it works 5/5

- Finally, slurmd periodically (every AcctGatherNodeFreq) issues an RPC REQUEST_ACCT_GATHER_UPDATE which returns an acct_gather_energy_t struct
 - scontrol show node
- Task's TRESUsage[IN/OUT]* are filled by common_jag.c
 - sstat and sacct

Future work/notes

- Overflow management needs to be determined, we don't know when the XCC BMC will overflow, we need feedback from Lenovo developers.
- EnergyIPMIDriverType=X flag could be modified to recognize a string and not a number, in order to clarify the settings for this parameter.
- The delta factor for `_get_joules_task()` is 10 seconds, this will avoid a slurmstepd task to ask for new ipmi readings too often.

DP02 - Disable cpu-freq



- LRZ needs to disable --cpu-freq switch for users.
- A job_submit.lua plugin has been developed which will wrap job submissions and throw an error if the user tries to specify the frequency.
- This plugin can be installed as documented in Slurm documentation for job submit lua plugin.
- Can be modified just editing the code.

DP02 - Disable cpu-freq



- The error will be similar to:

Requesting specific CPU Governor is not supported in this cluster

or

Requesting specific CPU Frequency is not supported in this cluster

- If debug is set in slurm.conf, a debug message in slurmctld log will be shown.