Cray XC
Power Monitoring & Management

Steven J. Martin

Cray XC Power Monitoring & Management: Outline

- Cray XC Power Monitoring and Control Overview
- Enhanced Hardware Power Monitoring
- HSS ramp rate limiting
- Power cap biasing
- Off-SMW Database
- SNMP on the SMW
- Plotting updates
- Backup material from 2015…

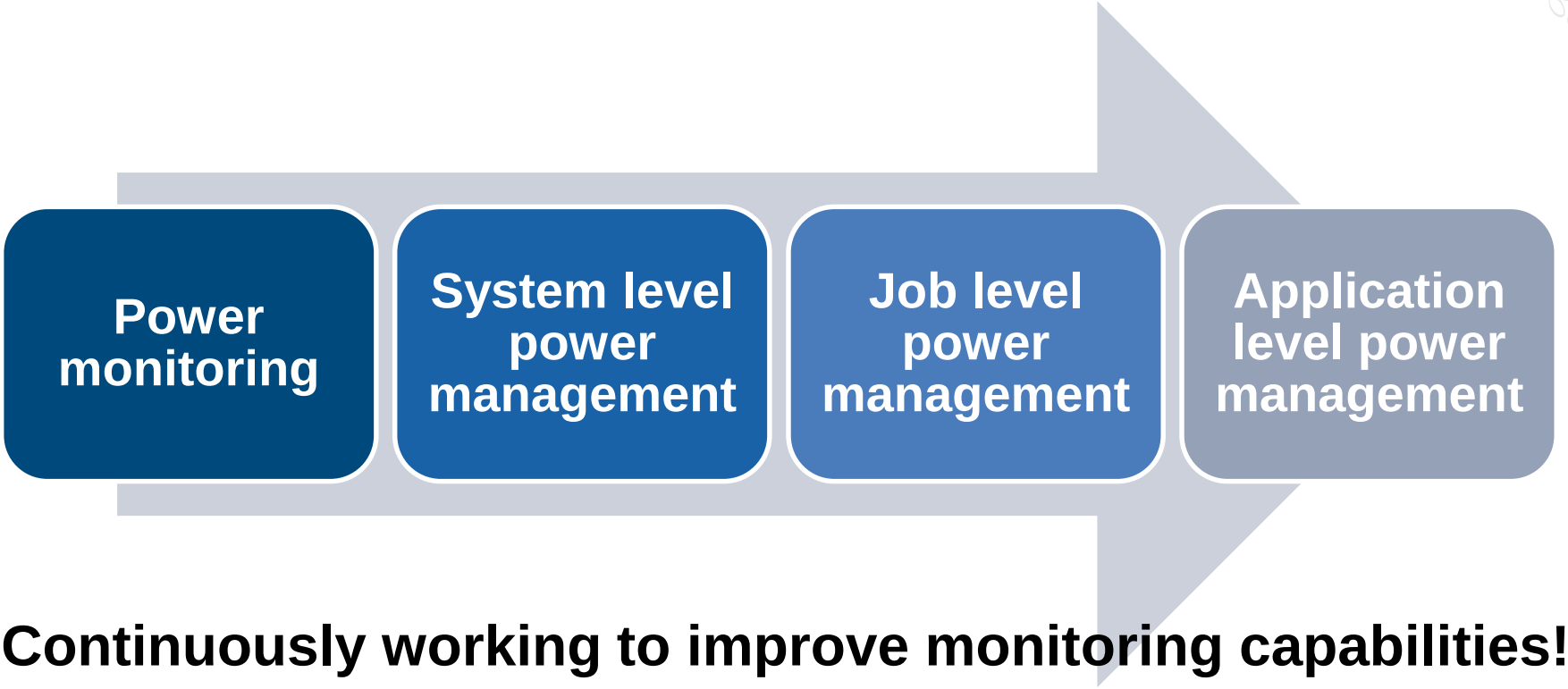
XC PM Overview

Power Management: Motivation & Philosophy

- **Motivation**
  - System procurements are increasingly constrained
    - Site power & cooling limitations
    - Cost of system power and cooling
  - Customer requirements
    - Power monitoring tools
    - Management of power consumption
    - Better performance per watt
  - Power limitations
    - 20 MW max power target for extreme-scale systems of the future

- **Philosophy**
  - Do not waste energy!
  - Measure power, so you know where it is going
  - Allow customers to affect greater power savings

Power Management: Progression



Continuously working to improve monitoring capabilities!

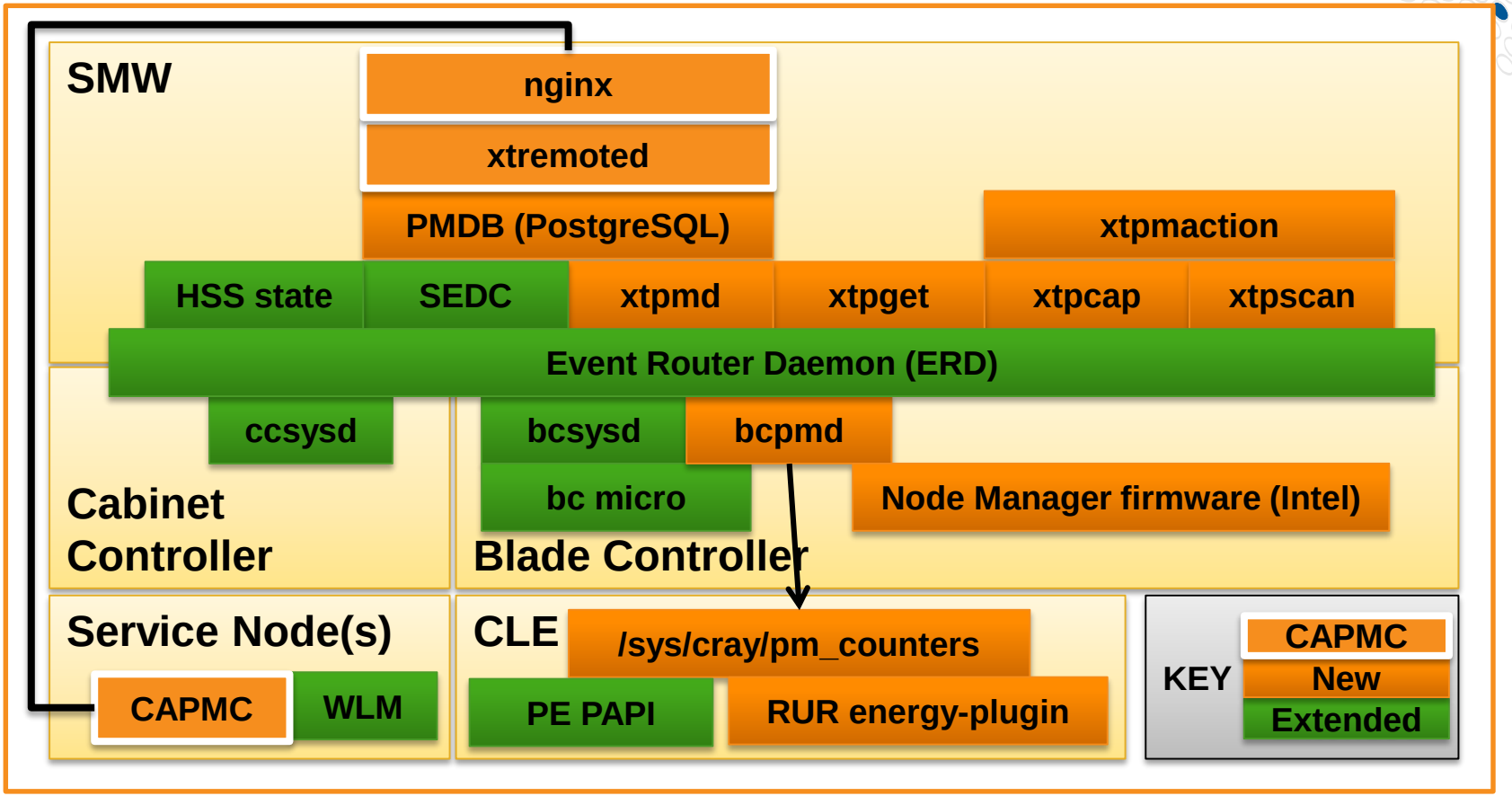
Capabilities Today on XC

Software stack
Out-of-band monitoring
In-band monitoring
Management

XC Monitoring and Control Quick List

- System Environmental Data Collection (SEDC)
- High-speed power/energy data collection and PMDB
- Turbo-boost limiting
- P-State at job launch
- Direct access to /sys/cray/pm_counters
- Resource Utilization Reporting (RUR), PAPI, & CrayPat
- Cray Advanced Platform Monitoring and Control

XC PM Software Stack



XC Out-Of-Band Monitoring

- **System Environmental Data Collection (SEDC)**
  - Cabinet-, blade-, and component-level data
  - Voltage, current, temperature, pressure, fan-speed, …
  - Flat-file data storage on SMW (deprecated)
    - Support dropping in R/R UP02
  - PMDB storage is default on R/R systems
- **System power monitoring with xtpget**
  - Command line access to real-time system power/energy data

XC Out-Of-Band Monitoring

- **High-speed power/energy data collection**
  - Cabinet, Blade, Node, and [Accelerator] data
  - Blade level data collection at 10 Hz
  - 1 Hz data into PMDB, 10Hz data into /sys/cray/pm_counters
- **Power Management Database (PMDB)**
  - Cabinet-level Power (including blower cabinets)
  - Blade-, and Node-level power and energy data
  - Job start-time, end-time, user-id, …, and node assignment data

New capabilities starting with blades supporting Intel Knights Landing processors

XC In-Band Monitoring: /sys/cray/pm_counters

- **Direct access to /sys/cray/pm_counters**
  - Unrestricted read-only access
- **Cray Resource Utilization Reporting (RUR)**
  - Application energy reporting via energy-plugin
  - Multiple reporting options
- **CrayPat & PAPI**
  - Intel RAPL counters
  - Cray custom counters

XC In-Band Monitoring

- /sys/cray/pm_counters (updated at 10Hz)

```
/sys/cray/pm_counters/accel_energy:10967358 J
/sys/cray/pm_counters/accel_power:20 W
/sys/cray/pm_counters/accel_power_cap:0 W
/sys/cray/pm_counters/energy:41006380 J
/sys/cray/pm_counters/power:55 W
/sys/cray/pm_counters/power_cap:0 W
/sys/cray/pm_counters/generation:95
/sys/cray/pm_counters/freshness:5494619
/sys/cray/pm_counters/startup:1429186804097771
/sys/cray/pm_counters/version:1
```

XC In-Band Monitoring

- /sys/cray/pm_counters (updated at 10Hz)

```
/sys/cray/pm_counters/accel_energy:10967358 J
/sys/cray/pm_counters/accel_power:20 W
/sys/cray/pm_counters/accel_power_cap:0 W
/sys/cray/pm_counters/energy:41006380 J
/sys/cray/pm_counters/power:55 W
/sys/cray/pm_counters/power_cap:0 W
/sys/cray/pm_counters/generation:95
/sys/cray/pm_counters/freshness:5494619
/sys/cray/pm_counters/startup:1429186804097771
/sys/cray/pm_counters/version:1
```

XC In-Band Monitoring

- /sys/cray/pm_counters (updated at 10Hz)

```
/sys/cray/pm_counters/accel_energy:10967358 J
/sys/cray/pm_counters/accel_power:20 W
/sys/cray/pm_counters/accel_power_cap:0 W
/sys/cray/pm_counters/energy:41006380 J
/sys/cray/pm_counters/power:55 W
/sys/cray/pm_counters/power_cap:0 W
/sys/cray/pm_counters/generation:95
/sys/cray/pm_counters/freshness:5494619
/sys/cray/pm_counters/startup:1429186804097771
/sys/cray/pm_counters/version:1
```

- Cray support via RUR, PAPI, and CrayPat

XC In-Band Monitoring

- **New descriptors**
  - Starting w/blades featuring the Intel KNL processor

```
/sys/cray/pm_counters/cpu_energy:32088906 J
/sys/cray/pm_counters/cpu_power:76 W
/sys/cray/pm_counters/memory_energy:4896643 J
/sys/cray/pm_counters/memory_power:13 W
/sys/cray/pm_counters/raw_scan_hz:10
/sys/cray/pm_counters/version:2
```

XC In-Band Monitoring

- **New descriptors**
  - Starting w/blades featuring the Intel KNL processor

```
/sys/cray/pm_counters/cpu_energy:32088906 J
/sys/cray/pm_counters/cpu_power:76 W
/sys/cray/pm_counters/memory_energy:4896643 J
/sys/cray/pm_counters/memory_power:13 W
/sys/cray/pm_counters/raw_scan_hz:10
/sys/cray/pm_counters/version:2
```

XC In-Band Monitoring

- **New descriptors**
  - Starting w/blades featuring the Intel KNL processor

```
/sys/cray/pm_counters/cpu_energy:32088906 J
/sys/cray/pm_counters/cpu_power:76 W
/sys/cray/pm_counters/memory_energy:4896643 J
/sys/cray/pm_counters/memory_power:13 W
/sys/cray/pm_counters/raw_scan_hz:10
/sys/cray/pm_counters/version:2
```

- **Future support for faster pm_counter scan rates**

XC In-Band Monitoring and Control

- **P-State at job launch**
  - Cray: aprun –p-state=<frequency>
  - Finding the optimum p-state
- **Turbo-boost limiting**
  - Boot time ability to enforce max turbo boost
  - Save energy at large scale due to variation in max turbo boost
- **Cray Advanced Platform Monitoring and Control (CAPMC)**
  - API for 3rd party Workload Manager (WLM) integration
  - WLM managed System-, node-, and job-level power monitoring, capping, and node-level power on | off control

CAPMC Enhancements (New in R/R)

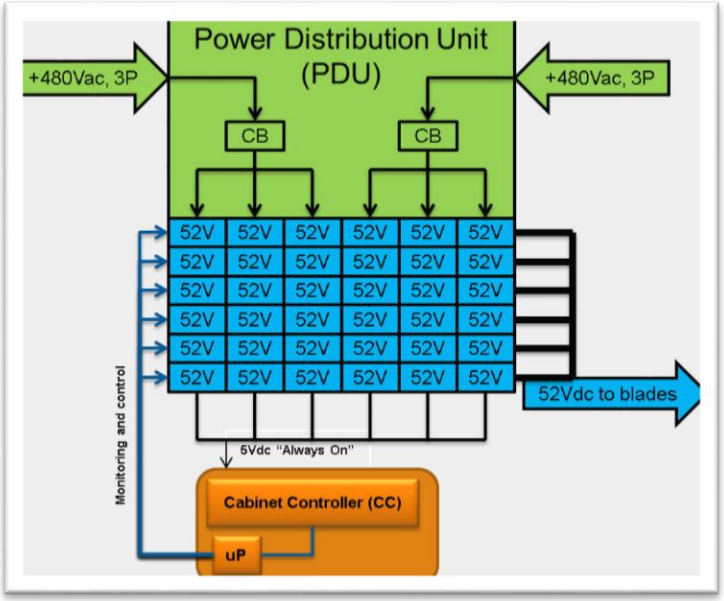
- **C-State Limiting**
  - `capmc get_sleep_state_limit_capabilities`
    - Returns all valid C-States for target node(s)
  - `capmc get_sleep_state_limit`
    - Returns the current C-State limits for target node(s)
  - `capmc set_sleep_state_limit`
    - Sets the C-State limit for the target node(s)

- **P-State Limiting**
  - `capmc get_freq_capabilities`
    - Returns all valid P-States for target node(s)
  - `capmc get_freq_limit`
    - Returns the current P-State limits for target node(s)
  - `capmc set_freq_limit`
    - Sets the P-State limits for the target node(s)

Hardware Power Monitoring

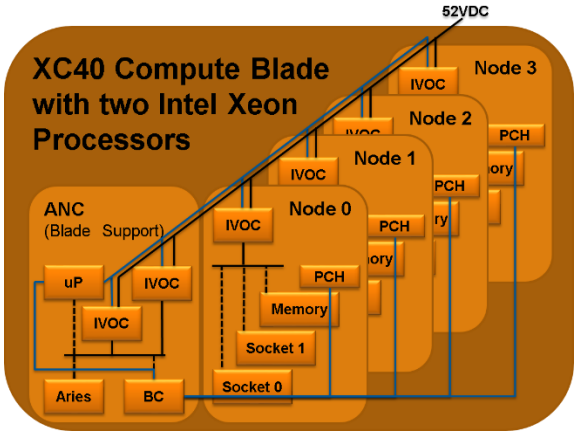
XC Cabinet Power Monitoring

- **36 bulk power supplies (rectifiers)**
  - 480 VAC to 52 VDC
  - 3000 Watts each
  - N+1 redundant
  - Hot-swap enabled
- **Cabinet controller software**
  - Sends data to PMDB



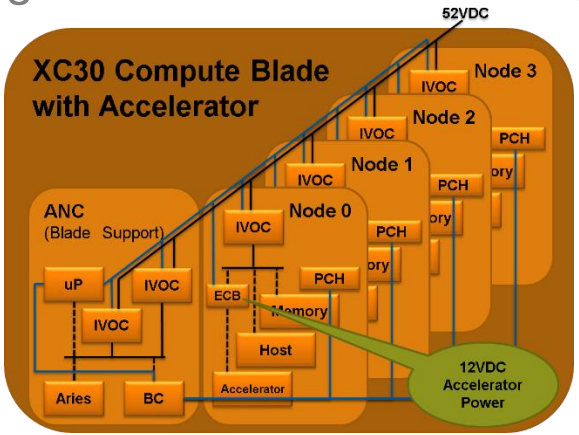
Two Socket (Xeon) Blade Power Monitoring

- **Six 52 VDC Electronic Circuit Breakers (ECBs)**
  - Monitor power into the blade (and nodes)
  - Two support Aries & blade support logic
  - Four support the nodes (one for each node)
- **Blade micro-processor (uP)**
  - Data polled at ~= 10Hz
- **Blade controller software**
  - Reports node power data into Intel chipset on demand
  - Sends power and energy data into PMDB for storages and analyses
  - Node data in-band (/sys/cray/pm_counters) at 10Hz



Accelerated (GPU/MIC) Blade Power Monitoring

- **Ten Electronic Circuit Breakers (ECBs)**
  - Two 52 VDC ECBs feed Aries & blade support logic
  - Four 52 VDC ECB, one for each node
  - Four 12 VDC ECB, one for each accelerator
- **Blade micro-processor (uP)**
  - Data polled at ~= 10Hz
- **Blade controller software**
  - Reports node power data into Intel chipset on demand
  - Sends power data to SMW for storages and analyses
  - Node & accelerator data in-band (/sys/cray/pm_counters) at 10Hz



The same infrastructure supports GPU and MIC accelerated blades!

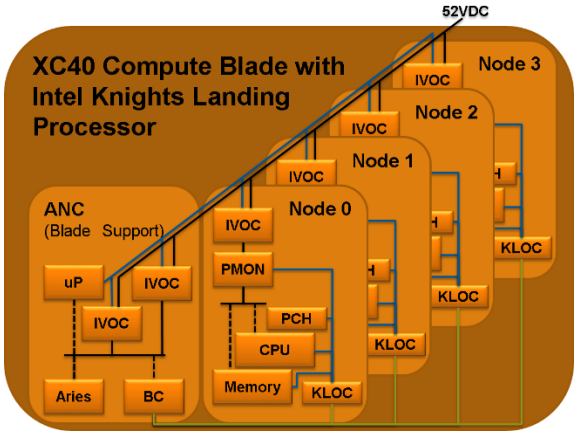
Hardware Power Monitoring

Cray XC40 blade with Intel Knights Landing (KNL) Processor, Enhanced Power Monitoring

- Paper: “Cray XC40 Power Monitoring and Control for Knights Landing”
- Presentation Wednesday, May 11th 2016 Technical Session 14C

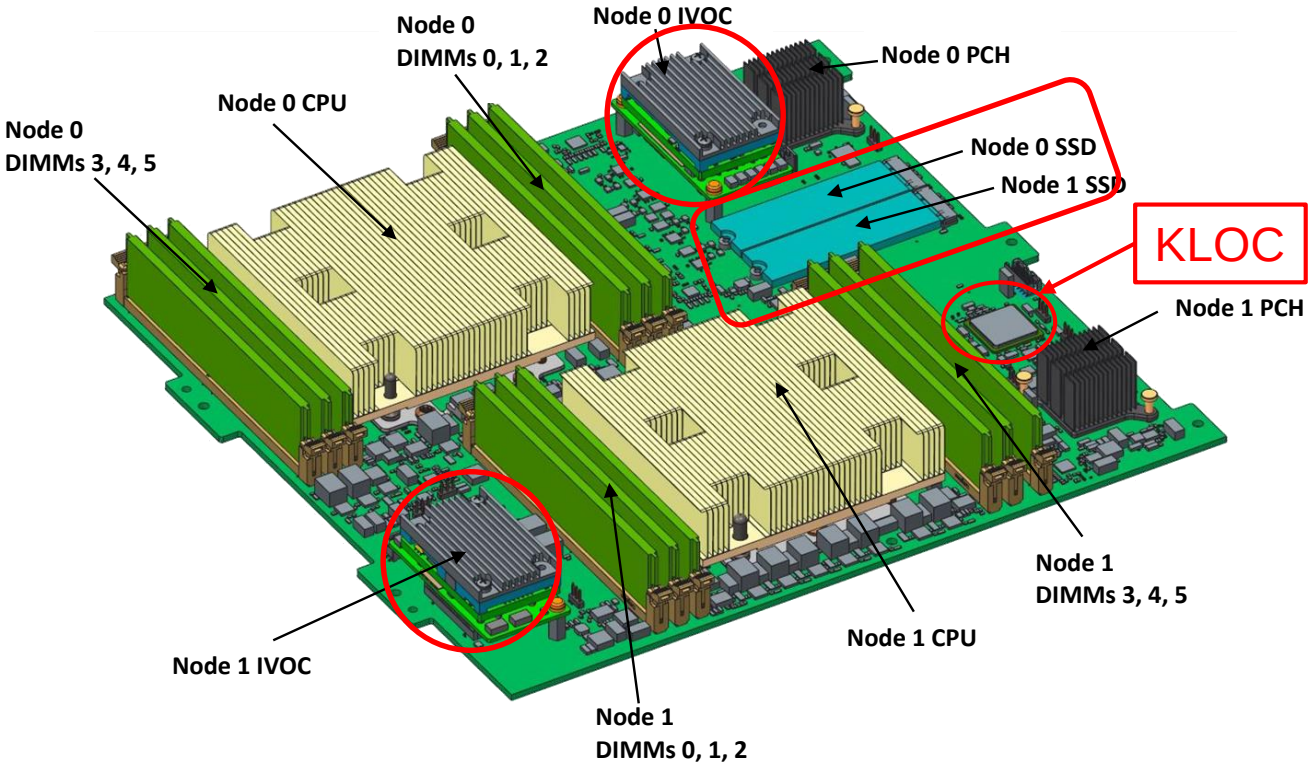
XC40 Intel KNL Processor Blade Power Monitoring

- **Six 52 VDC Electronic Circuit Breakers (ECBs)**
  - Monitor power into the blade (and nodes)
  - Two support Aries & blade support logic
  - Four support the nodes (one for each node)
- **Blade controller based telemetry collection**
  - Data polled at >= 10Hz
- **Blade controller software**
  - Reports node power data into Intel chipset on demand
  - Sends power and energy data into PMDB for storages and analyses
  - Node data in-band (/sys/cray/pm_counters) at 10Hz



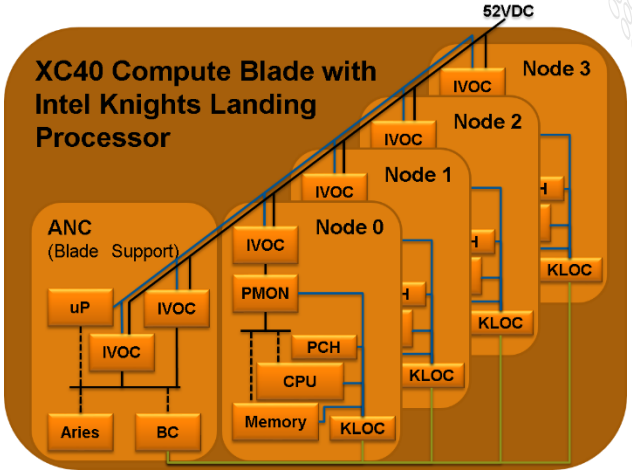
New infrastructure starting with XC40 blades with Intel KNL processors

KPDC Isometric View

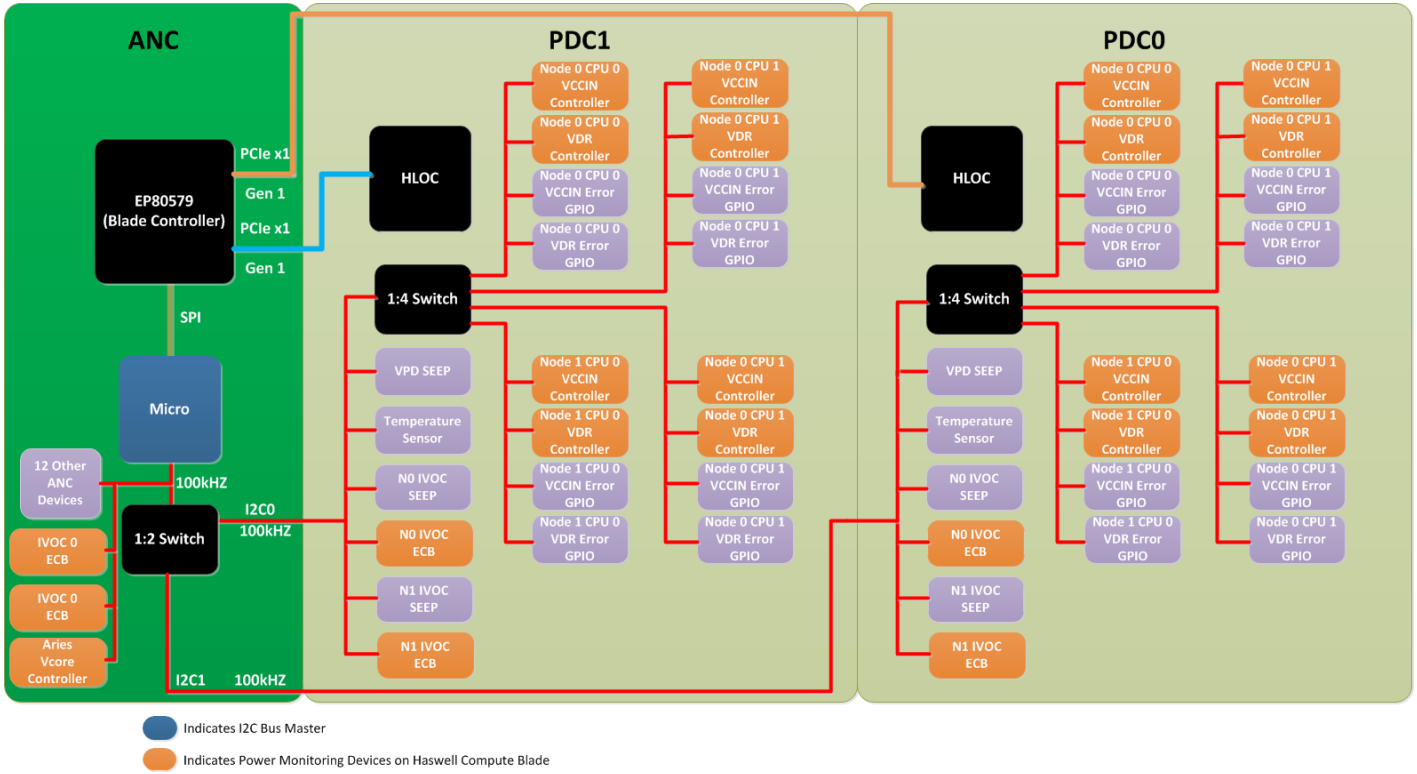


Cray XC enhanced HSS blade-level monitoring

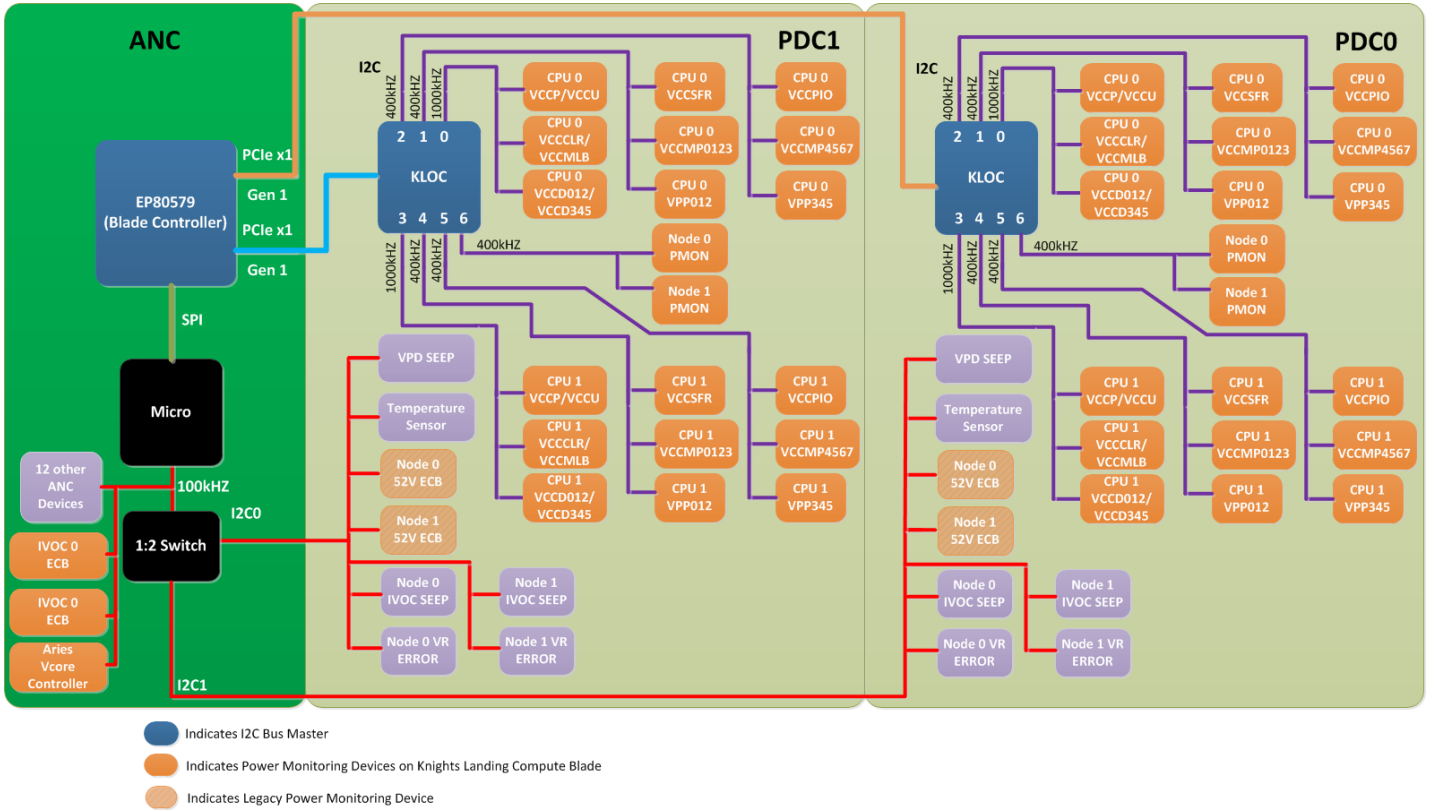
- **I2C devices connected to KLOC**
  - Six I2C masters
  - Fewer devices on each bus
  - Faster I2C clocks (400 kHz or 1MHz)
  - More I2C transactions in-flight (in parallel!)
- **Node-level power sensor (PMON)**
  - 12-bit ADC
  - 1kHz sampling rate
  - Hardware averaging filter, configured to match HSS polling rate
- **Blade Controller connected to KLOC via PCIe**



Previous XC30 and XC40 blades



Cray XC enhanced HSS blade-level monitoring



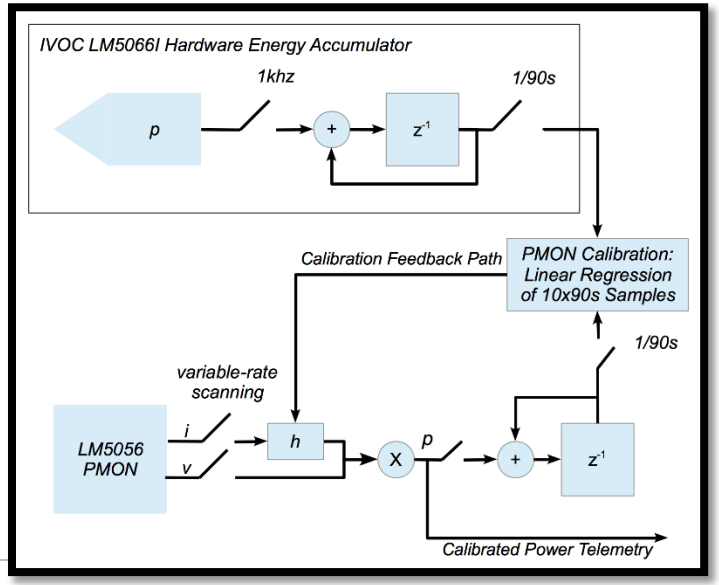
Cray XC enhanced HSS blade-level monitoring

- **PMON (LM5056)**
  - High Voltage System Power Measurement Device with PMBus
  - http://www.ti.com/lit/ds/symlink/lm5056.pdf
  - Connected to KLOC

- **IVOC (LM5066I)**
  - High Voltage System Power Management and Protection IC with PMBus
  - http://www.ti.com/lit/ds/symlink/lm5066i.pdf
  - Factory calibrated to better then +- 1% Accuracy

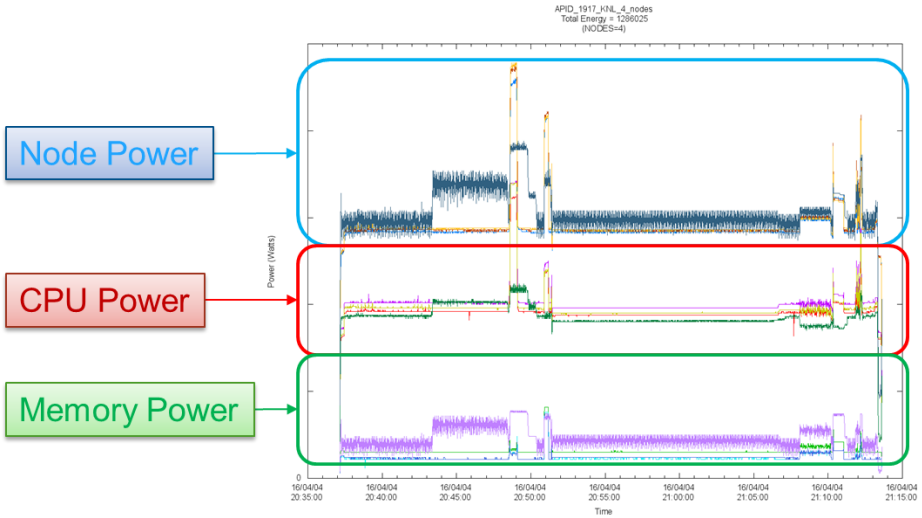
PMON Calibration

- Takes advantage of the factory calibrated TI LM5066I power sensor
- Compares LM5066I (IVOC) with LM5056 (PMON) readings
- Correcting subsequent PMON readings
- More details in our 2016 CUG paper!



New Power/Energy Monitoring Enhancements

- **New for XC40 Blades supporting Intel KNL processors**
  - Aggregate sensors for cpu and memory telemetry
    - Abstract, stable interfaces this and planned future blades
- **New PMDB telemetry**
  - 16 new sensor IDs
  - Data in pmdb.bc_data
  - 1Hz default rate



New Power/Energy Monitoring Enhancements

- **New for XC40 Blades supporting Intel KNL processors**
  - Aggregate sensors for cpu and memory telemetry
    - Abstract, stable interfaces this and planned future blades
- **New PMDB telemetry**
  - 16 new sensor IDs
  - Data in pmdb.bc_data
  - 1Hz default rate

```
ID | Sensor Description    | Unit
---+-----------------------+-----
36 | Node 0 CPU Power      |  W
37 | Node 0 CPU Energy     |  J
44 | Node 1 CP
45 | Node 1 CP
52 | Node 2 CP
53 | Node 2 CP
60 | Node 3 CP
61 | Node 3 CP
```

```
68 | Node 0 Memory Power   |  W
69 | Node 0 Memory Energy  |  J
76 | Node 1 Memory Power   |  W
77 | Node 1 Memory Energy  |  J
84 | Node 2 Memory Power   |  W
85 | Node 2 Memory Energy  |  J
92 | Node 3 Memory Power   |  W
93 | Node 3 Memory Energy  |  J
```

New Power/Energy Monitoring Enhancements

- **SEDC blade-level power telemetry**
- **Data sent by default to PMDB**
  - Into the pmdb.bc_sedc table
  - Once every 30 seconds
  - Maximum reporting rate of 1Hz
- **Voltage, Current, and Thermal**
  - Data also collected

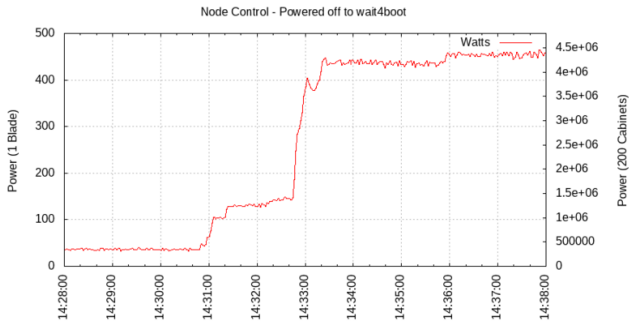
ID	Name
2128	BC_P_NODE0_VCCLR_POUT
2129	BC_P_NODE0_VCCMLB_POUT
2130	BC_P_NODE0_VCCD012_POUT
2131	BC_P_NODE0_VCCD345_POUT
2132	BC_P_NODE0_VCCP_POUT
2133	BC_P_NODE0_VCCU_POUT
2134	BC_P_NODE0_VCCSFR_POUT
2135	BC_P_NODE0_VCCMP0123_POUT
2136	BC_P_NODE0_VPP012_POUT
2137	BC_P_NODE0_VCCPIO_POUT
2138	BC_P_NODE0_VCCMP4567_POUT
2139	BC_P_NODE0_VPP345_POUT

HSS Ramp Rate Limiting

HSS Ramp Rate Limiting: Motivation

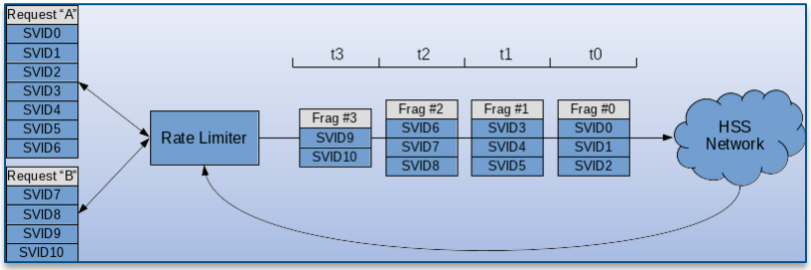
- Changes in power state often target large portions of HPC systems
- Resulting in multi-megawatt power fluctuations in a matter of seconds
- At large sites, this rapid power fluctuation may over-stress the power grid
  - Potential to cause cascading power failures affecting other power grid customers
- Sites causing such power events may end up paying more for their power
- Solution: The ability to smooth out large power fluctuations over time
  - Ramp Rate Limiting

The bulk of the transition, +3MW, occurs in 20 sec (+9MW / minute)



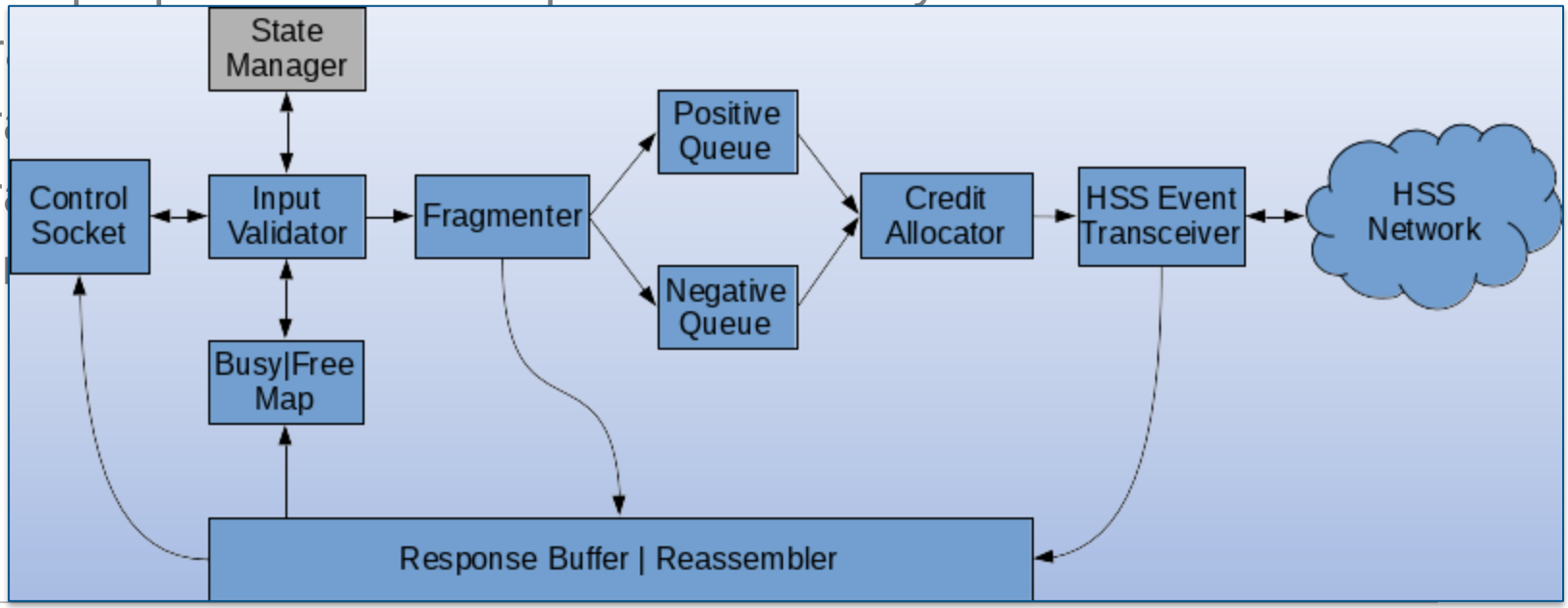
HSS Ramp Rate Limiting: Proposed Solution

- **Involves new process that:**
  - Accepts power-related requests of arbitrary size
  - Fragments them into smaller units
  - Transmits fragments at a controlled rate
  - Track fragment responses (and/or) timeouts
  - Single response to caller after all fragments complete or timeout



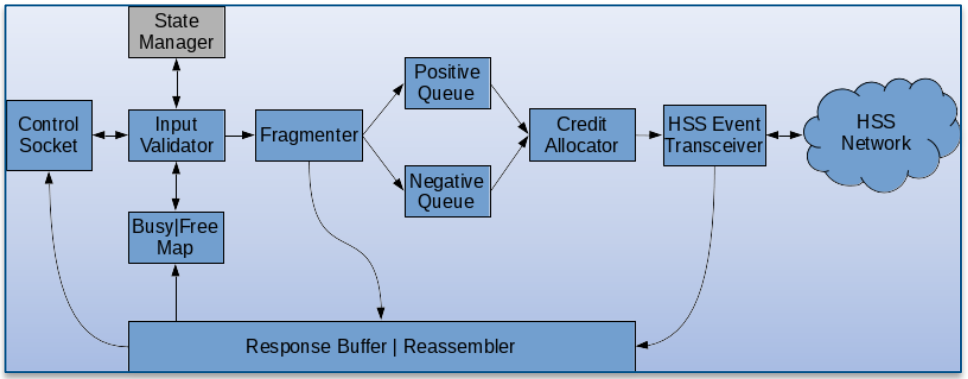
HSS Ramp Rate Limiting: Proposed Solution

- **Involves new process that: (In More Detail)**
  - Accepts power-related requests of arbitrary size
  - Fr
  - Tr
  - Tr
  - Si



HSS Ramp Rate Limiting: xtpowerd

- **New xtpowerd implements**
  - Input Validator
  - Fragmenter
  - Credit Allocator
  - Tracks component transition states (Busy | Free Map)
  - Reassembles responses
  - Handles timeouts
  - Mitigates conflicts



What System Admin Need to Know / Do

- **Configuration for this feature is held in an ini file:**
  - /opt/cray/hss/default/etc/xtpowerd.ini
- **Values can be changed on the fly!**
  - Editing the file
  - Then send xtpowerd a SIGHUP!
- **This allows for on the fly configuration**
  - Along with syncing configuration for SMW HA

```
[xtpowerd]
# Only the following two settings require administrator configuration. The
# default establishes a 2MW/minute limit when enabled. NOTE: Any trailing spaces
# on integer values prevent the parser from interpreting the values!

# ramp_limited: true or false respectively meaning enabled or disabled
ramp_limited=false
# ramp_limit: integer, watts/minute (default: 2000000, for ±2MW per minute)
#ramp_limit=2000000

# The following two power band parameters are informational only, and are not
# enforced by xtpowerd. To better understand how these values can be used,
# please review the Cray CAPMC API documentation.
#
# Note: Even though these values are not enforced, power_band_min cannot be
#       greater than power_band_max.

# power_band_min: integer, watts (default: 0, for unbounded)
# NOTE: This value is informational only, and is not enforced.
#power_band_min=0

# power_band_max: integer, watts (default: 0, for unbounded)
# NOTE: This value is informational only, and is not enforced.
#power_band_max=0

# The following settings may be tuned if desired, but provided defaults should work.

# window_size: integer, credit allocation ±window limit override or zero to use
# automatically computed value
#window_size=0

# The cost of each operation represents the power delta, in watts, which results
# on a per component basis from each operation. WARNING: These values should
# only be edited by Cray service personnel.
#[cost]
#ec_l0_node_up=80
#ec_l0_node_dwn=-125
#ec_boot=-50
#ec_node_halt=55
#ec_l1_cab_up=4080
#ec_l1_cab_dwn=-4080
#ec_l1_slot_up=85
#ec_l1_slot_dwn=-85
#ec_node_cmd=85
#ec_l0_reset=85
```

What System Admin Need to Know / Do

- **Configuration for this feature is held in an ini file:**
  - /opt/cray/hss/default/etc/xtpowerd.ini
- **Values can be changed on the fly!**
  - Editing the file
  - Then send xtpowerd a SIGHUP!
- **This**
  - Alc

```
[xtpowerd]
# Only the following two settings require administrator configuration. The
# default establishes a 2MW/minute limit when enabled. NOTE: Any trailing spaces
# on integer values prevent the parser from interpreting the values!

# ramp_limited: true or false respectively meaning enabled or disabled
ramp_limited=false
# ramp_limit: integer, watts/minute (default: 2000000, for ±2MW per minute)
#ramp_limit=2000000
…
```

What System Admin Need to Know / Do

- **Configuration for this feature is held in an ini file:**
  - /opt/cray/hss/default/etc/xtpowerd.ini
- **Values can be changed on the fly!**
  - Editing the file
  - Then send xtpowerd a SIGHUP!

```
root@smw: pkill -SIGHUP xtpowerd
```

- **This**
  - Alc

```
[xtpowerd]
# Only the following two settings require administrator configuration. The
# default establishes a 2MW/minute limit when enabled. NOTE: Any trailing spaces
# on integer values prevent the parser from interpreting the values!

# ramp_limited: true or false respectively meaning enabled or disabled
ramp_limited=true
# ramp_limit: integer, watts/minute (default: 2000000, for ±2MW per minute)
#ramp_limit=2000000
…
```

Power Cap Bias Capability

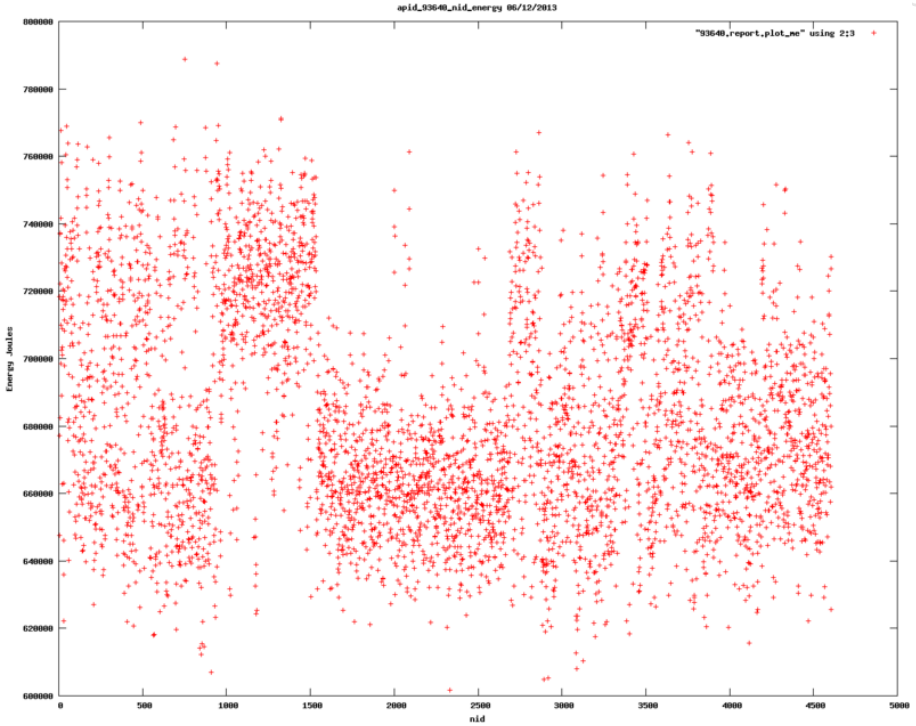
New in R/R

Power Cap Bias: Problem/Motivation

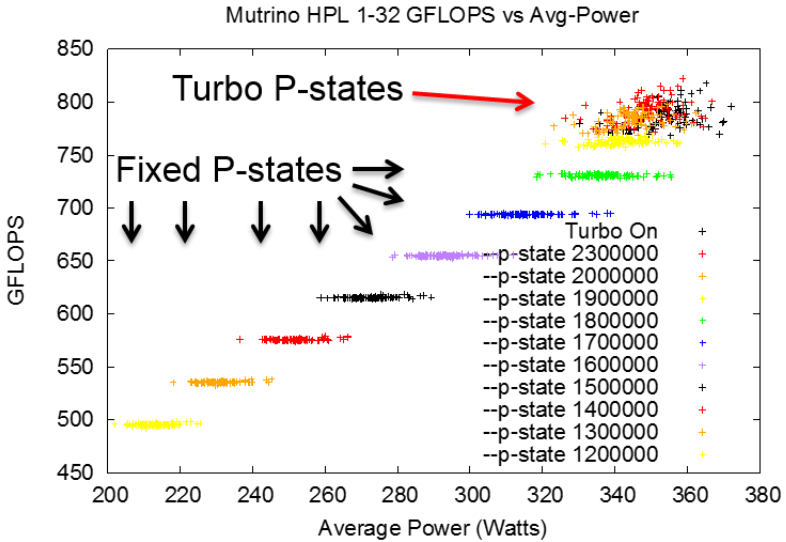
- **Due to manufacturing process variation:**
  - For a given performance level, per node energy usage will vary
- **Power capping all nodes in a parallel job to an equal value…**
  - Effectively forces non-uniform performance
- **This is not what we want when running HPC systems!**

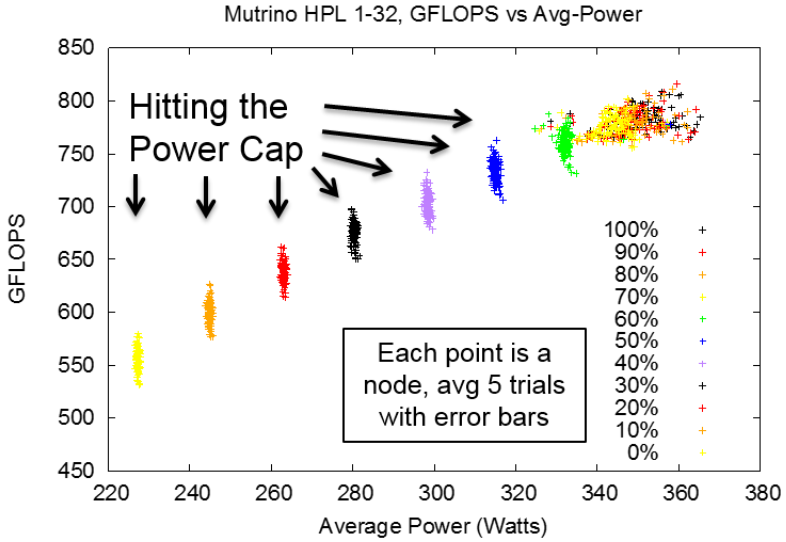
Power Cap Bias: Problem/Motivation

- **Scatter plot (nid,energy)**
  - NID number (X axis)
  - Job energy (Y axis)
- **Approximately 4500 nodes**
  - memtester for ~40 min runtime
- **Range of ~80 watts**
  - Data from June 2013



Power Cap Bias: Problem/Motivation





Sandia test data from XC40 Haswell blades with Intel Xeon E5-2698V3 processors

Power Cap Bias: Solution

- **Select code (or kernel) for system workload characterization**
  - System-level characterization
- **Run code on all nodes in the system uncapped**
  - Collecting average power data for each node
  - Using average power data, compute power bias factor for each node
  - Save per-node bias factor into the PMDB
- **When a power cap is request for a group of nodes:**
  - The requested value is adjusted (per-node) using the bias factor
  - On average all nodes are capped at the requested value

Power Cap Bias: Solution - Modified

- **Characterize code/application on all nodes**
  - Perhaps at maximum fixed non-turbo frequency
  - Collect average power data for each node
  - Save average power data for all nodes for future use

- **Running the selected code/application under a power cap**
  - Generate update bias data for all node assigned to the job
  - Update PMDB with bias data to all node assigned to the job
  - Launch job with power cap bias data customized for:
    - The specific code/application and nodes assigned to the application!

Bias Factor Computation

1) Consider a database table named 'bias_data' containing per node energy data.

```
nid | energy
----+-------
1   | 770000
2   | 720000
3   | 800000
```

2) Calculate the bias factor:

"select nid, energy / (select avg(energy) from bias_data) as bias from bias_data;"

```
nid | bias
----+-----
1   | 1.01
2   | 0.94
3   | 1.05
```

3) Calculated per node caps, assuming a 300 watt power cap target:

```
nid | cap
----+----
1   | 303
2   | 283
3   | 314
```

Power Cap Bias: Solution

- **CAPMC node bias factor applets:**
  - Power cap bias data stored in PMDB
  - get_power_bias
  - set_power_bias
  - clr_power_bias
  - set_power_bias_data
  - compute_power_bias

- **See S-2553-11,chapter 3 for details**

Power Cap Bias: pbutil

- **New script pbutil**
  - Stands for “power bias utility”
  - Collects and publishes power cap bias factor data into PMDB
  - Leverages CAPMC “power_bias” functionality
- **The *pbutil* is run from a login node as root**

```
loginx:~> su -
loginx:~ # module load capmc
loginx:~ # pbutil –help
```

- **The capmc module must be loaded**
  - On the login node where *pbutil* is being executed

Power Cap Bias

- **Power cap bias feature is new with R/R**
- **The *pbutil* script:**
  - Enables testing and research use-cases
- **CAPMC 'power bias' applets enable WLM innovation**
  - Workload management software value-add
- **We are looking for your feedback**

PMDB Tuning

R/R UP01

PostgreSQL Configuration Tuning

- **As storage size increases**
  - It is important to tune PostgreSQL configuration
  - See “Tune the PMDB” section of S-0043
  - 1 TB dedicated storage now default on new systems
- **Currently a manual process**
  1. Update /var/lib/pgsql/data/postgresql.conf per S-0043
  2. Followed by “systemctl restart postgresql”
- **PostgreSQL configuration autotuning planned**
  - Targeted at SMW 8.0.UP02 release

PMDB Tuning: New *xtpmdbconfig* Options

- **-E, --estimate:**
  - Estimate storage requirements for current configuration.
- **-A, --autoconfig:**
  - Automatically configure PMDB sizing for available storage on the partition holding the database data directory
- **-d, --dryrun:**
  - Print an autoconfig proposal
- **-z, --storage-size:**
  - Specify a storage size for PMDB
- **-r, --report:**
  - Report statistics about the database (timespans of stored data)

xtpmdbconfig -r

```
smw:~> xtpmdbconfig -r
Showing 3 bc_data attributes
--------------------
Earliest timestamp              = 2016-04-20 02:12:48.706779-05:00
Latest timestamp                = 2016-05-03 13:01:42.625458-05:00
Timespan                        = 13 days, 10:48:53.918679

Showing 3 bc_sedc_data attributes
--------------------
Earliest timestamp              = 2016-03-30 23:21:03.442809-05:00
Latest timestamp                = 2016-05-03 13:01:47.654076-05:00
Timespan                        = 33 days, 13:40:44.211267

Showing 3 cc_data attributes
--------------------
Earliest timestamp              = 2016-03-31 07:00:26.438189-05:00
Latest timestamp                = 2016-05-03 13:01:44.441754-05:00
Timespan                        = 33 days, 6:01:18.003565

Showing 3 cc_sedc_data attributes
--------------------
Earliest timestamp              = 2016-04-10 01:46:19.374969-05:00
Latest timestamp                = 2016-05-03 13:01:50.658069-05:00
Timespan                        = 23 days, 11:15:31.283100
```

Off-SMW Database Support

(New) R/R UP01
Motivation
Data Output Plugin API
Event Router Daemon Enhanced Network Topology

Motivation

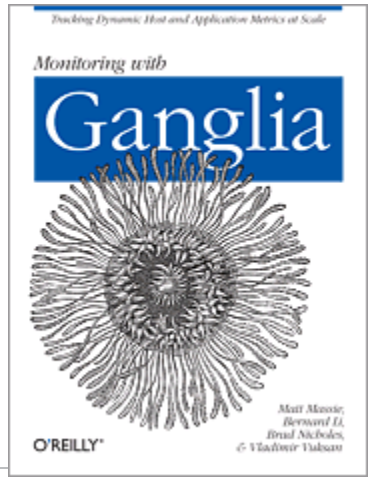
- **Offer customers an interface for streaming telemetry**
  - 'The Cray' is not an island
  - Customers want to use their own data center monitoring systems
- **Separate monitoring and control**
  - Multiple security domains
  - Resource utilization
- **Isolate PMDB**
  - From other critical HSS infrastructure.

Data Output Plugin API

On-SMW or Off-SMW

Data Output Plugins

- **Send data 'off system'**
  - Customers have preferences
  - No 'one size fits all' solution
  - Enabling connections into
    - Redis, InfluxDB, Monasca, Ganglia, etc…
- **Implemented as shared library**
  - Loaded by 'xtpmd_plugd'
  - Shared memory communication with xtpmd
    - Shared memory access hidden from plugin
  - 'xtpmd_plugd' calls library method per data sample



Data Output Plugins

- **Plugins run in own process**
  - Inspired by modern web browsers
  - Automatic restart on crash
    - Can run “unsupervised” (development mode)
- **Can’t corrupt shared memory**
  - Segfault if shared memory write attempted
- **Can’t interfere with other plugins**
  - Slow plugins won’t block others
  - Data is dropped

Data Output Plugins

- **Plugin development**
  - One header file
    - /opt/cray/hss/default/pm/xtpmd_api/xtpmd_plugin.h
  - Example plugin source code
  - No 3rd party dependencies

- **Demo plugin (shipping with SMW 6.0 UP01)**
  - Writes to CSV file
    - /opt/cray/hss/default/pm/xtpmd_api/xtpmd_plugin_csv.c
  - Functionally, but not very useful
  - Intended to jumpstart customer plugin development

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Objective**
  - Tap into cabinet level SEDC data
  - Stream comma separated value (CSV) data through FIFO
  - Read CSV data into your application framework

- **Required tooling (for this CUG2016 demo)**
  - C compiler
  - GLib-2 development headers
    - This demo uses ‘g_time_val_to_iso8601()’ & ‘g_malloc0()’
    - Glib2 not a hard requirement for all plugin development

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Entry Point**
  - Called by xtpmd_plugd
  - Initialize internal state
- **“demo_ctx_t”**
  - Plugin-specific data type
  - Maintains private state
- **“ivar”**
  - Instance parameters

```
/****** CUG2016 Plugin Demo - Stream CSV to FIFO ******/
int init_sedc_cc_ctx(int version, pmd_data_opts_t *opts)
{
    /* XTPMD and API header plugin version must match */
    if(version != XTPMD_PLUGIN_VERSION) return -1;

    /* Allocate internal state */
    demo_ctx_t *ctx = g_malloc0(sizeof(demo_ctx_t));
    ctx->log = stdout;

    /* Write to FIFO if configured or standard out */
    char *ivar = plugin_get_instance_var("fifo");
    if(ivar) {
        FILE* f = fopen(ivar, "w");
        if(f) ctx->log = f;
        plugin_free_instance_var(ivar);
    }

    /* Assign callbacks and private state data */
    opts->recv_double = stub_recv_double;
    opts->flush = stub_flush;
    opts->user_data = ctx;
    return 0;
}
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Receive Function**
  - Called once per sample

```
/****** CUG2016 Plugin Demo - Stream CSV to FIFO ******/
static void demo_recv_double(void *user_data,
    uint64_t ts, int source, int id, double value)
{
    /* Private state available in callback functions */
    demo_ctx_t *ctx = (demo_ctx_t*)user_data;

    /* Timestamp is microseconds since the unix epoch */
    GTimeVal tv = {
        .tv_sec  = ts / 1000000,
        .tv_usec = ts % 1000000,
    };

    /* Convert timestamp to string */
    char *tvs = g_time_val_to_iso8601(&tv);

    /* Write CSV data to FIFO */
    fprintf(ctx->log,
            "%s,%d,%d,%f\n", tvs, source, id, value);
    g_free(tvs);
}
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Flush Function**
  - Called ~1 per second
  - Can be unimplemented

```
/****** CUG2016 Plugin Demo - Stream CSV to FIFO ******/
static void demo_flush(void *user_data)
{
    /* Implement plugin specific 'flush' logic */
    stub_ctx_t *ctx = (stub_ctx_t*)user_data;
    fflush(ctx->log);
}
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Build Plugin**
  - Simple Makefile

```
# CUG2016 Plugin Demo – Stream CSV to FIFO
# Makefile
OBJ = demo_plugin.o
LIB = demo_plugin.so

CFLAGS  += -O2 -fPIC -I/opt/cray/hss/default/pm/xtpmd_api/
CFLAGS  += $(shell pkg-config --cflags glib-2.0)
LDFLAGS += $(shell pkg-config --libs glib-2.0)

$(LIB) : $(OBJ)
        $(CC) -shared -o $(LIB) $(OBJ) $(LDFLAGS)
```

- **Configure Instance**
  - Path to FIFO
  - Shared object

```
# CUG2016 Plugin Demo – Stream CSV to FIFO
# /opt/cray/hss/default/etc/xtpmd_plugins.ini
...
[demo]
fifo=/home/crayadm/demo/sedc_stream
object=/home/crayadm/demo/demo_plugin.so
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Test Plugin**
  - Run ‘xtpmd_plugd’ unsupervised
    - `xtpmd_plugd [shm-key] [shm-size] [instance] [ini-file]`
  - Create FIFO
    - `mkfifo /home/crayadm/demo/sedc_stream`
  - Locate shared memory region

```
crayadm@smw:~/> ipcs

...

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x0052e2c1 0          postgres   600        48         24
0x7a060809 65537      crayadm    600        4194304    1

...
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Test Plugin**
  - Run ‘xtpmd_plugd’ unsupervised

```
crayadm@smw:~/> xtpmd_plugd 0x7a060809 4194304 \
                            demo /opt/cray/hss/default/etc/xtpmd_plugins.ini
SHM Key: 0x7a060809, SHM Len: 4194304, Plugin: /home/crayadm/demo/demo_plugin.so
```

  - Read streaming CSV data from FIFO

```
crayadm@smw:~/> cat sedc_stream
2016-04-29T18:51:39.577903Z,402653184,1079,51.900000
2016-04-29T18:51:39.577903Z,402653184,1080,51.910000
2016-04-29T18:51:39.577903Z,402653184,1081,51.930000
2016-04-29T18:51:39.577903Z,402653184,1082,51.920000
2016-04-29T18:51:39.577903Z,402653184,1083,51.910000
...
```

CUG2016 Plugin Demo – Stream CSV to FIFO

- **Deploy Plugin**
  - Run ‘xtpmd_plugd’ supervised via xtpmd
    - Edit ‘xtpmd_plugins.ini’
    - Add “demo” to semicolon delimited “instances” list

```
# /opt/cray/hss/default/etc/xtpmd_plugins.ini
[plugins]
instances=rabbit;demo
...
[demo]
fifo=/home/crayadm/demo/sedc_stream
object=/home/crayadm/demo/demo_plugin.so
...
```

  - Restart xtpmd
  - Write streaming CSV reader init script

Source ID to Hostname Mapping

- **Plugin API does not provide ID mapping functions**
  - Query database for static source to hostname list
  - Source ID number may be used as hashtable key

```
#!/bin/sh
# generate-source-map.sh
psql pmdb pmdbuser <<- EOT
  COPY (
    select cname2source(name) as source, name
    from sm.expand('rt_l1', 's0')
  ) TO STDOUT with CSV;
  COPY (
    select cname2source(name) as source, name
    from sm.expand('rt_l0', 's0')
  ) TO STDOUT with CSV;
  COPY (
    select cname2source(name) as source, name
    from sm.expand('rt_node', 's0')
  ) TO STDOUT with CSV;
EOT
```

```
crayadm@smw:~/> ./generate-source-map.sh
402653184,c0-0
134217728,c0-0c0s0
134218240,c0-0c0s1
134218752,c0-0c0s2
134219264,c0-0c0s3
134219776,c0-0c0s4
134220288,c0-0c0s5
134220800,c0-0c0s6
134221312,c0-0c0s7
134221824,c0-0c0s8
...
```

PMDB Sensor ID to Name Mapping

- **Plugin API does not provide ID mapping functions**
  - Query database for static ID to sensor name list
  - Sensor ID number may be used as hashtable key

```
#!/bin/sh
# generate-sensor-map.sh
psql pmdb pmdbuser <<- EOT
  COPY (
    select sensor_id, sensor_name, sensor_units
    from pmdb.sensor_info
  ) TO STDOUT with CSV;
EOT
```

```
crayadm@smw:~/> ./generate-sensor-map.sh
0,Cabinet Power,W
1,Cabinet Energy,J
2,Cabinet Voltage,mV
3,Cabinet Current,A
8,Cabinet Blower Power,W
16,HSS Power,W
17,HSS Energy,J
18,HSS Voltage,mV
19,HSS Current,mA
32,Node 0 Power,W
33,Node 0 Energy,J
34,Node 0 Voltage,mV
...
```

SEDC Scan ID to Sensor Name Mapping

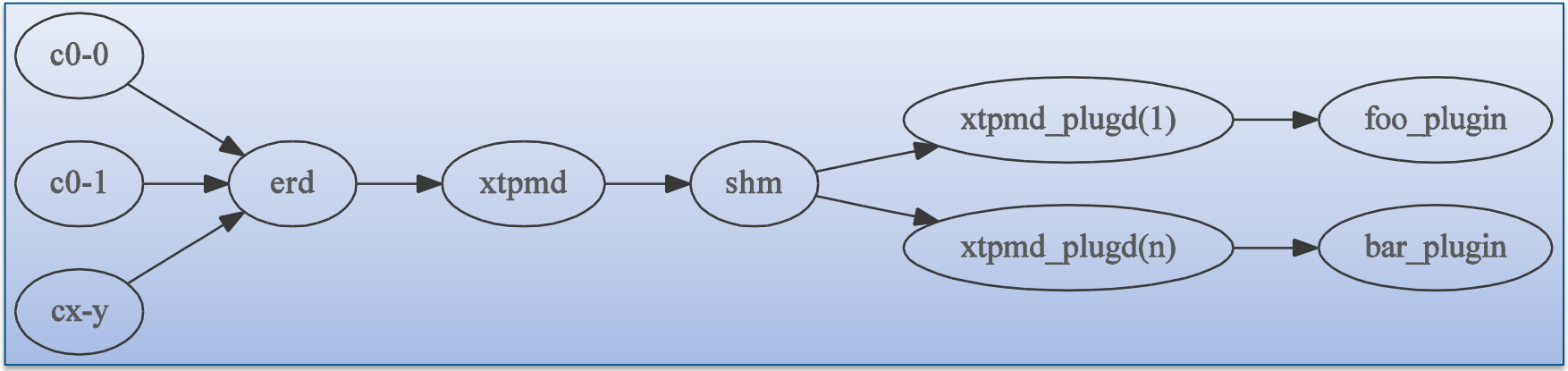
- **Plugin API does not provide ID mapping functions**
  - Query database for static ID to SEDC name list
  - SEDC Scan ID may be used as hashtable key

```
#!/bin/sh
# generate-sedc-map.sh
psql pmdb pmdbuser <<- EOT
  COPY (
    select sensor_id, sensor_name, sensor_units
    from pmdb.sedc_scanid_info
  ) TO STDOUT with CSV;
EOT
```

```
crayadm@smw:~/> ./generate-sedc-map.sh
991,CC_T_MCU_TEMP,degC
992,CC_T_PCB_TEMP,degC
993,CC_V_VCC_5_0V,V
994,CC_V_VCC_5_0V_FAN1,V
995,CC_V_VCC_5_0V_SPI,V
996,CC_V_VDD_0_9V,V
997,CC_V_VDD_1_0V_OR_1_3V,V
998,CC_V_VDD_1_2V,V
999,CC_V_VDD_1_2V_GTP,V
1000,CC_V_VDD_1_8V,V
...
```

Streaming Data Plugin

- **System telemetry flows into xtpmd**
  - Lock free single writer / multiple reader semantics
  - Many plugins may be running at once



Event Router Daemon (ERD) Network Topology – Enhanced

Event Router Daemon (ERD)

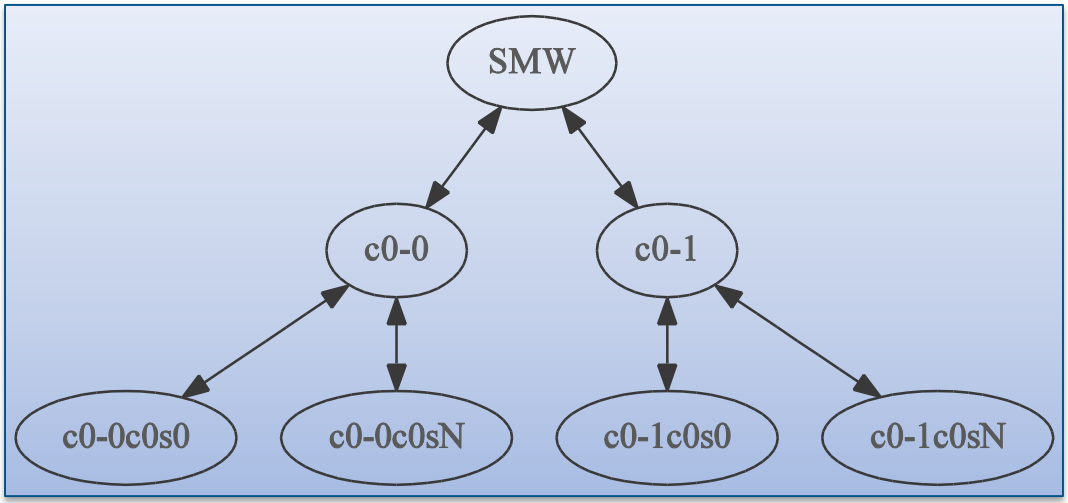
- **Message bus in HSS**
  - Hierarchical message passer
    - Publish / Subscribe paradigm
  - Unreliable message delivery (like UDP)
    - No receiver acknowledgement
    - Messages may be dropped in transit
- **Primary functions**
  - Command and Control
    - Initiated from System Management Workstation (SMW)
  - System monitoring
    - Data push from below

ERD Network Topology – Traditional

- **Single Root Node**
  - SMW
  - Command and Control
- **Intermediate Nodes**
  - Cabinet Controllers (CC)
  - Communicate upstream to SMW
- **Leaf Nodes**
  - Blade Controllers (BC)
  - Communicate upstream to parent CC

ERD Network Topology – Traditional

- **CCs communicate one upstream host**
  - SMW presents a bottleneck
    - Limited compute power
    - Limited storage capacity

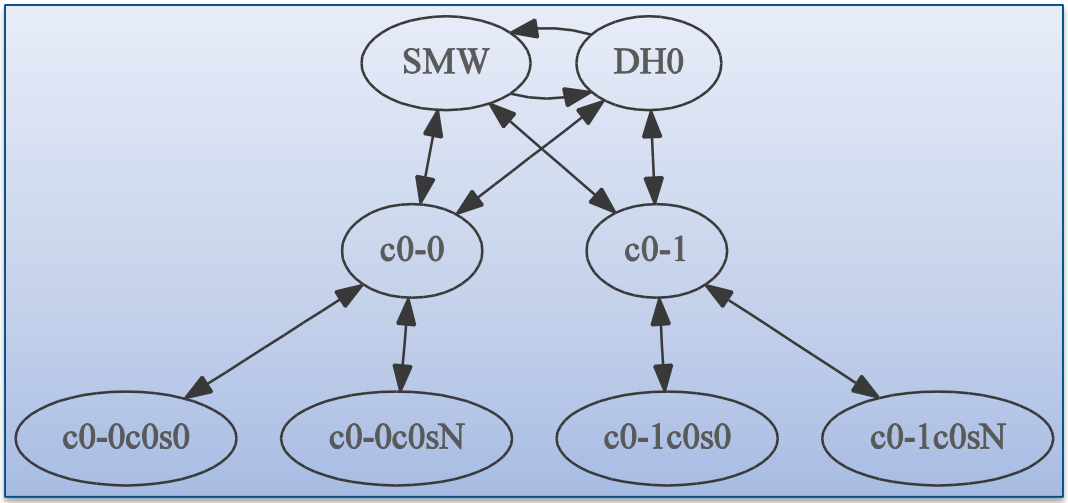


ERD Network Topology – Enhanced

- **Multiple root nodes**
  - SMW, Database, etc…
  - Communicate among themselves
- **Intermediate nodes**
  - Cabinet Controllers
  - Communicate upstream to one or more root nodes
  - Selective routing of upstream traffic
- **Leaf nodes**
  - Blade Controllers
  - Communicate upstream to parent CC

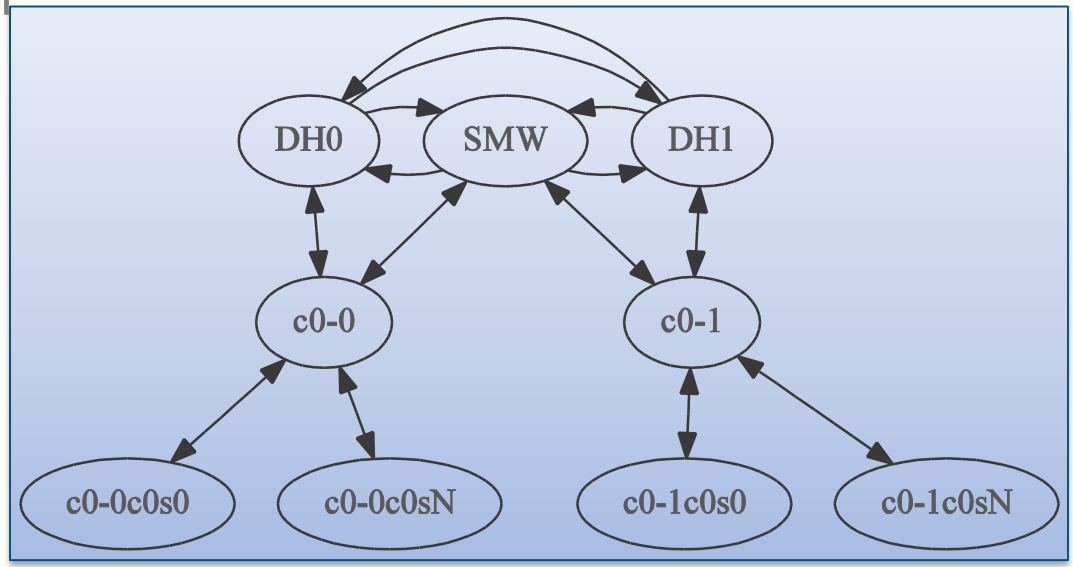
ERD Network Topology – Enhanced

- **CCs communicate with all upstream hosts**
  - Moves bottleneck from SMW to single DB node
    - Relieves network pressure on SMW
  - No horizontal scaling



ERD Network Topology – Enhanced

- **CCs communicate with subset of upstream hosts**
  - SMW handles command and control
  - Distributed data collection
  - Horizontal scaling!



Simple Network Management Protocol (SNMP)

(NEW) R/R support for SNMP on the SMW

SNMP on Cray XC systems

- **Simple Network Management Protocol (SNMP)**

- **System-level power statistics available via SNMP:**
  - Instant power (W)
  - Peak power (W)
  - Average power (W)
  - Accumulated energy (J)

SNMP on the SMW

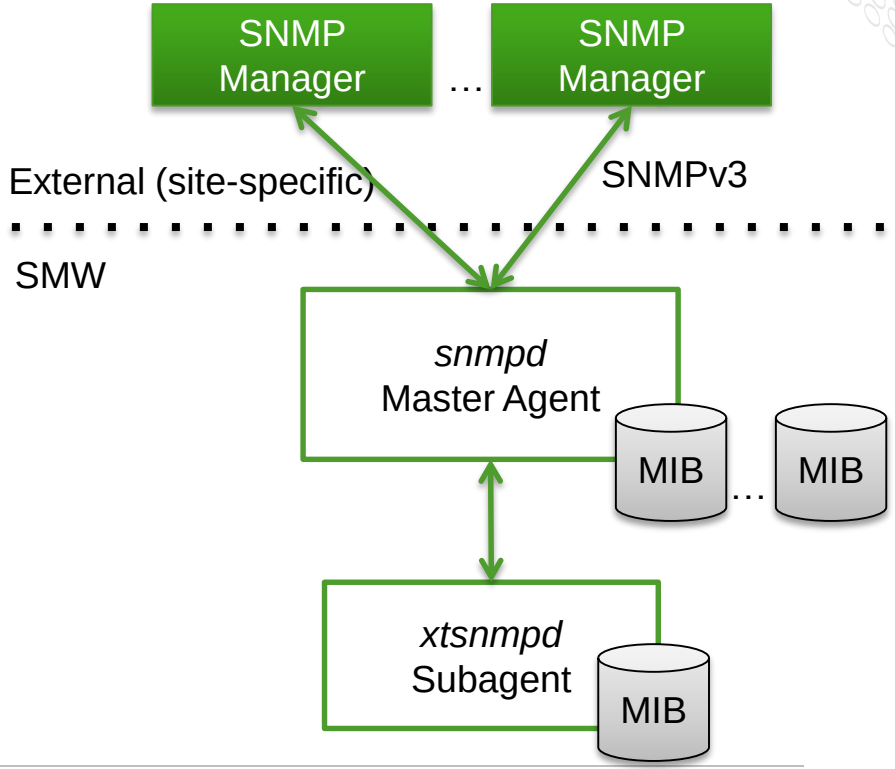
- **Cray-specific Management Info**
  - Base (MIB) files available on SMW
- **Configuration utility: xtsnmpd_setup**
- **Leverages Net-SNMP as the master SNMP service**

SNMP on the SMW Details

- **For security reasons, SNMP v3 connections only**
  - SNMP v1 and v2c connections are not supported and disallowed
  - Configured automatically by the xtsnmpd_setup utility
- **Authentication by User-based Security Model (USM)**
- **Transmission encryption using AES-128**
- **MIB files for use by external SNMP clients:**
  - Available under /opt/cray/hss/default/etc/snmp/mibs on the SMW
  - CRAY-SMI.txt describes the structure of the Cray “enterprise”
  - CRAY-XC-MIB.txt provides all Cray XC-series objects

xtsnmpd – A New HSS Daemon

- Subagent under the Net-SNMP snmpd daemon
- snmpd responds for SUSE- and Net-SNMP- distributed (generic) SNMP objects
- Subagent (xtsnmpd) responds for Cray-specific objects



Xtsnmpd_setup

- Must have root privileges to run xtsnmpd_setup

```
crayadm@smw:~> type xtsnmpd_setup
xtsnmpd_setup is /opt/cray/hss/default/bin/xtsnmpd_setup

crayadm@smw:~> xtsnmpd_setup
ERROR: This script must be run as root.
Exiting...
```

xtsnmpd_setup --help

```
smw:~ # xtsnmpd_setup --help
usage: xtsnmpd_setup [-h] [--on] [--off] [--create_user CREATE_USER]
                     [--new_auth_pw NEW_AUTH_PW] [--new_priv_pw NEW_PRIV_PW]
                     [--remove_user REMOVE_USER] [--auth_pw AUTH_PW]
                     [--priv_pw PRIV_PW] [--user USER]

Cray SNMP daemon configuration script

optional arguments:
  -h, --help                show this help message and exit
  --on                      enable the Cray SNMP daemon.
  --off                     disable the Cray SNMP daemon.
  --create_user CREATE_USER create an SNMPv3 user.
  --new_auth_pw NEW_AUTH_PW authentication password for new SNMPv3 user.
  --new_priv_pw NEW_PRIV_PW privacy/encryption password for new SNMPv3 user.
  --remove_user REMOVE_USER remove an SNMPv3 user.
  --auth_pw     AUTH_PW     authentication password for removing SNMPv3 user.
  --priv_pw     PRIV_PW     privacy/encryption password for removing SNMPv3 user.
  --user USER               user id for remove_user action (not necessarily the
                            user id being removed!)
```

Xtsnmpd_setup

- **Toggle SNMP on and off:**

```
xtsnmpd_setup --on | --off
```

- **Add user named ‘craysnmp’ with random passwords:**

```
xtsnmpd_setup --create_user craysnmp --new_auth_pw
‘9scpIsJMvb-m5S4’ --new_priv_pw '6w@zqEX3eYpe.mN'
```

- **Remove user ‘baduser’ using craysnmp’s credentials:**

```
xtsnmpd_setup --remove_user baduser --user craysnmp --
auth_pw '9scpIsJMvb-m5S4' --priv_pw '6w@zqEX3eYpe.mN'
```

- **Note: User add/remove requires snmpd restart**
  - Which will make SNMP objects unavailable for up to 60 seconds.

SNMP Client Example

```
# Query Cray-provided MIB for system power/energy stats
   whitebox:~> snmpwalk -v 3 -u craysnmp -l authPriv -x AES -a MD5 -A
   '9scpIsJMvb-m5S4'  -X '6w@zqEX3eYpe.mN' example-smw cray
   CRAY-XC-MIB::crayXCSysInstantPower.0 = Gauge32: 197079
   CRAY-XC-MIB::crayXCSysAvgPower.0 = Gauge32: 196205
   CRAY-XC-MIB::crayXCSysPeakPower.0 = Gauge32: 198886
   CRAY-XC-MIB::crayXCSysAccumEnergy.0 = Counter64: 2591835295

# Query Net-SNMP-provided MIB giving SMW uptime
   whitebox:~> snmpget -v 3 -u craysnmp -l authPriv -x AES -a MD5 -A
   '9scpIsJMvb-m5S4'  -X '6w@zqEX3eYpe.mN' example-smw sysUpTime.0
   DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (70391) 0:11:43.91
```

Plotting Data

New Example Script (Redwood 6.0 UP01)

cray_pmdb_report_system_power.py

- **System power from PMDB (pmdb.cc_data)**

  /opt/cray/hss/default/pm/script_examples/cray_pmdb_report_system_power.py

- **Replaces older unreleased bash/awk script:**

  smw: …/stevem/gen_system_power_plot -s <STIME> -e <ETIME>

cray_pmdb_report_system_power.py

- **System power from PMDB (pmdb.cc_data)**

  /opt/cray/hss/default/pm/script_examples/cray_pmdb_report_system_power.py

```
smw:~> cray_pmdb_report_system_power.py --help
usage: cray_pmdb_report_system_power.py [-h] [-a] [-d] [-v] [-s START_TIME][-e END_TIME] [-p]

Reports total system power from pmdb.cc_data. With no options specified, this script will query
the most recent 10 minutes of data, and will only report the calculated total system power.

optional arguments:
  -h, --help            show this help message and exit
  -a, --all_data        Query for all possible system data. Overrides '-s' and '-e'
  -d, --debug           Display performance metrics while the script runs.
  -v, --verbose         Displays per-cabinet data in summary report
  -s START_TIME, --start START_TIME     (Start time as 'YYYY-MM-DD HH:MM:SS‘)
  -e END_TIME, --end END_TIME           (End time as 'YYYY-MM-DD HH:MM:SS‘)
  -p, --plot            Generate plot of system power in PNG format
```

cray_pmdb_report_system_power.py

- Syst

/opt/cra py

```
smw:~> cray_p
usage: cray_p

Reports total
the most rece

optional argu
  -h, --help
  -a, --all_d
  -d, --debug
  -v, --verbo
  -s START_TI
  -e END_TIME
  -p, --plot
```

TIME] [-p]

t will query
m power.

```
smw:~> cray_pmdb_report_system_power.py -av
System Power Summary Report:
[Data Summary]
    First recorded      : 2016-03-19 19:42:04-05:00
    Last recorded       : 2016-04-15 11:28:51-05:00
    Time span           : 26 days, 15:46:47
    No. Cabinets        : 2
    No. Blower Cabinets : 1
[Total System Power]
    Power Type |            Max |         Median |        Average
    -----------------------------------------------------------
    DC         |      143351.00 |       39464.00 |       55300.99
    AC         |      150518.55 |       41437.20 |       58066.04
[Per Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       65162.00 |       14650.00 |       22833.98
    c1-0       |       71190.00 |       15024.00 |       22242.05
[Per Blower Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       13200.00 |        9790.00 |       10224.97
```

cray_pmdb_report_system_power.py

- Syst
  /opt/cra py
- Rep
  smw IME>

```
smw:~> cray_pmdb_report_system_power.py -av
System Power Summary Report:
[Data Summary]
    First recorded      : 2016-03-19 19:42:04-05:00
    Last recorded       : 2016-04-15 11:28:51-05:00
    Time span           : 26 days, 15:46:47
    No. Cabinets        : 2
    No. Blower Cabinets : 1
[Total System Power]
    Power Type |            Max |         Median |        Average
    -----------------------------------------------------------
    DC         |      143351.00 |       39464.00 |       55300.99
    AC         |      150518.55 |       41437.20 |       58066.04
[Per Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       65162.00 |       14650.00 |       22833.98
    c1-0       |       71190.00 |       15024.00 |       22242.05
[Per Blower Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       13200.00 |        9790.00 |       10224.97
```

cray_pmdb_report_system_power.py

- Syst
  /opt/cra py
- Rep
  smw IME>

```
smw:~> cray_pmdb_report_system_power.py -av
System Power Summary Report:
[Data Summary]
    First recorded      : 2016-03-19 19:42:04-05:00
    Last recorded       : 2016-04-15 11:28:51-05:00
    Time span           : 26 days, 15:46:47
    No. Cabinets        : 2
    No. Blower Cabinets : 1
[Total System Power]
    Power Type |            Max |         Median |        Average
    -----------------------------------------------------------
    DC         |      143351.00 |       39464.00 |       55300.99
    AC         |      150518.55 |       41437.20 |       58066.04
[Per Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       65162.00 |       14650.00 |       22833.98
    c1-0       |       71190.00 |       15024.00 |       22242.05
[Per Blower Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       13200.00 |        9790.00 |       10224.97
```

cray_pmdb_report_system_power.py

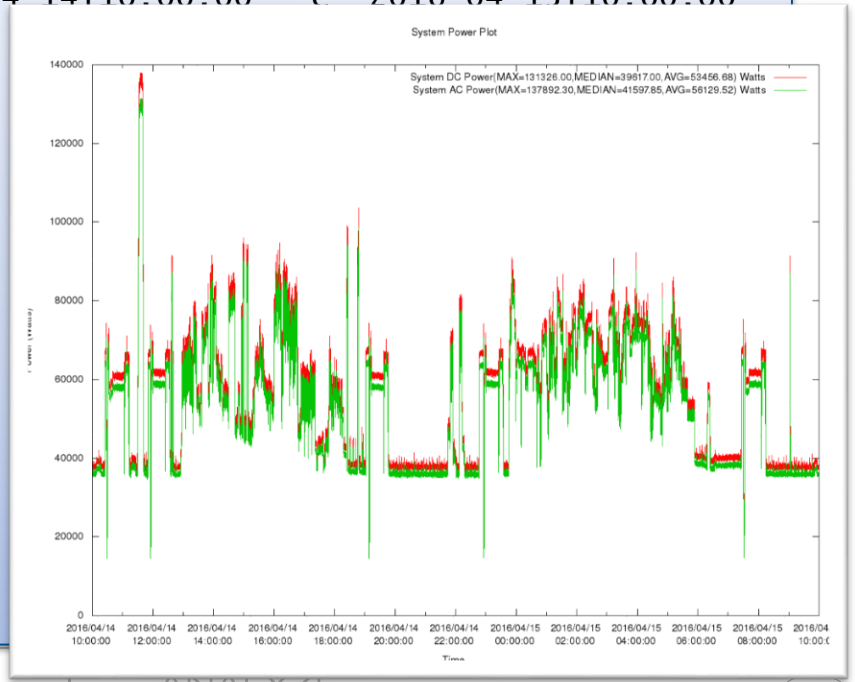
- Syst
  /opt/cra py
- Rep
  smw IME>

```
smw:~> cray_pmdb_report_system_power.py -av
System Power Summary Report:
[Data Summary]
    First recorded      : 2016-03-19 19:42:04-05:00
    Last recorded       : 2016-04-15 11:28:51-05:00
    Time span           : 26 days, 15:46:47
    No. Cabinets        : 2
    No. Blower Cabinets : 1
[Total System Power]
    Power Type |            Max |         Median |        Average
    -----------------------------------------------------------
    DC         |      143351.00 |       39464.00 |       55300.99
    AC         |      150518.55 |       41437.20 |       58066.04
[Per Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       65162.00 |       14650.00 |       22833.98
    c1-0       |       71190.00 |       15024.00 |       22242.05
[Per Blower Cabinet Power (DC Power in Watts)]
    Cabinet    |            Max |         Median |        Average
    -----------------------------------------------------------
    c0-0       |       13200.00 |        9790.00 |       10224.97
```

cray_pmdb_report_system_power.py

- System power from PMDB (pmdb.cc_data)

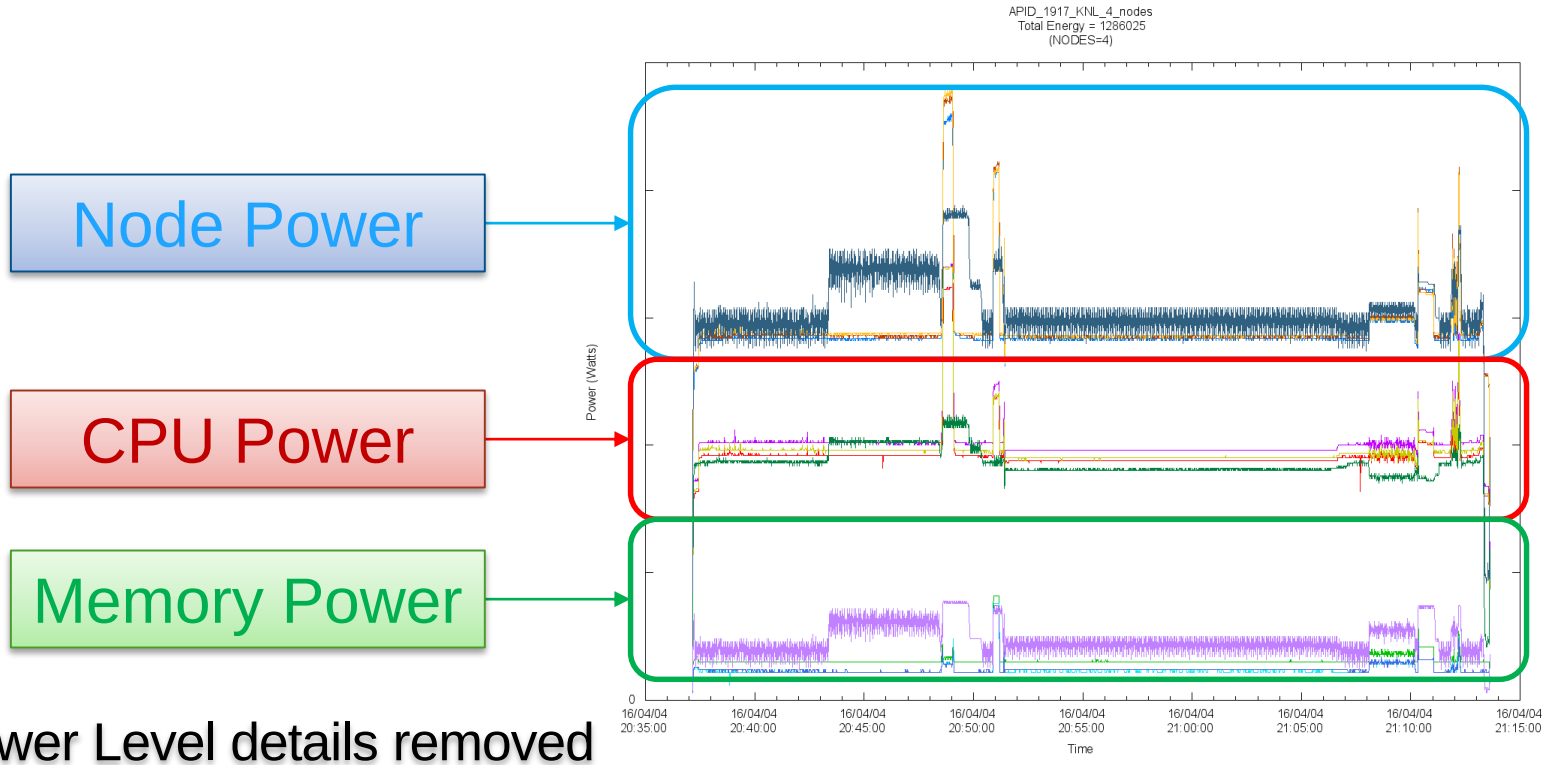
```
smw:~> cray_pmdb_report_system_power.py -p -s "2016-04-14T10:00:00" -e "2016-04-15T10:00:00"
Collecting CC Power Data
Processing Data
Plotting System Power Data
* file: system_power_plot.png
System Power Summary Report:
[Data Summary]
First recorded      : 2016-04-14 10:00:00-05:00
Last recorded       : 2016-04-15 09:59:59-05:00
Time span           : 23:59:59
No. Cabinets        : 2
No. Blower Cabinets : 1
[Total System Power]
Power Type |         Max |      Median |     Average
-----------------------------------------------------
DC         |   131326.00 |    39617.00 |    53456.68
AC         |   137892.30 |    41597.85 |    56129.52
```



Plotting Data

Cray XC40 blade with Intel Knights Landing (KNL) Processor

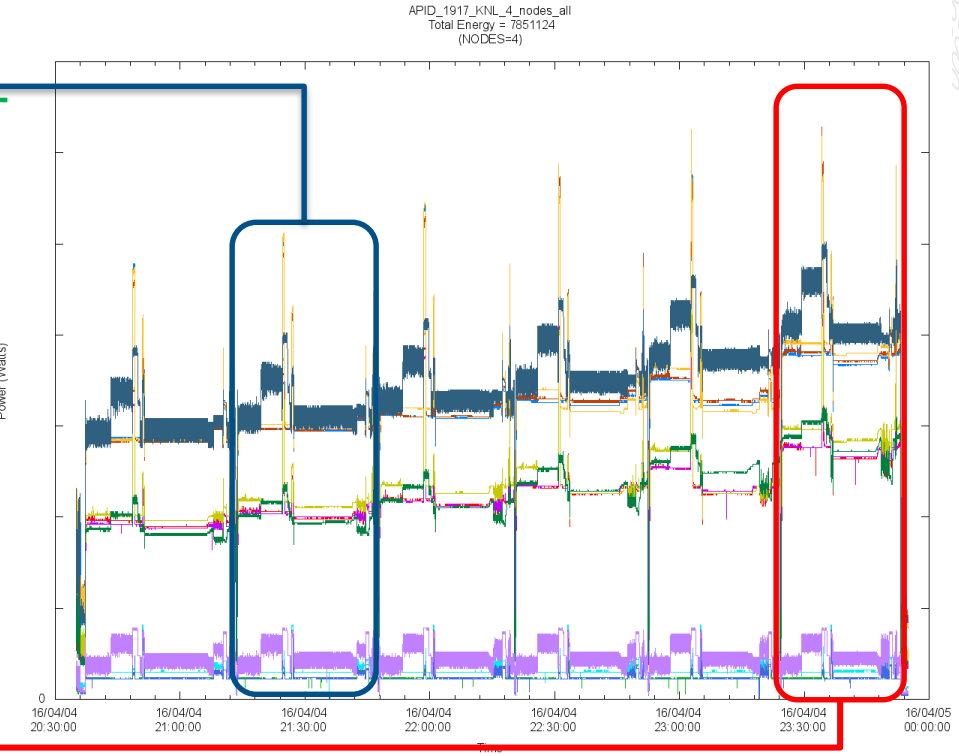
APID 1917 Test Power Profile (4-Node Test)



Power Level details removed

Summary Plot Six 4-Node Runs

APID	Joules	Runtime
1917	1286025	00:36:28.99
1918	1257640	00:34:07.17
1919	1268345	00:32:42.08
1920	1298037	00:31:58.58
1921	1333215	00:31:43.99
1922	1353328	00:28:52.91



Power Level details removed

Plotting Data

Content from 2015 CUG tutorial

Verbose Text Output

```
c0-0 CC_T_COMP_AMBIENT_TEMP0(MAX = 23.800000 MEDIAN = 23.400000 AVG = 23.297600)
c0-0 CC_T_COMP_AMBIENT_TEMP1(MAX = 22.800000 MEDIAN = 22.100000 AVG = 22.120000)
c0-1 CC_T_COMP_AMBIENT_TEMP0(MAX = 22.300000 MEDIAN = 21.500000 AVG = 21.555200)
…
c9-1 CC_T_COMP_WATER_TEMP_OUT(MAX = 22.400000 MEDIAN = 19.800000 AVG = 19.830400)
System DC Power (MAX = 1541254 , AVG = 328265 , MEDIAN = 303892) Watts
    c0-0 CAB_DC_POWER(MAX = 74210 MEDIAN = 12519 AVG = 13644) Watts
    c0-1 CAB_DC_POWER(MAX = 74307 MEDIAN = 12336 AVG = 13523) Watts
    c1-0 CAB_DC_POWER(MAX = 73331 MEDIAN = 12465 AVG = 13617) Watts
    …
    c9-1 CAB_DC_POWER(MAX = 74563 MEDIAN = 12367 AVG = 13582) Watts
```

Cabinet

Verbose Text Output

```
c0-0 CC_T_COMP_AMBIENT_TEMP0(MAX = 23.800000 MEDIAN = 23.400000 AVG = 23.297600)
c0-0 CC_T_COMP_AMBIENT_TEMP1(MAX = 22.800000 MEDIAN = 22.100000 AVG = 22.120000)
c0-1 CC_T_COMP_AMBIENT_TEMP0(MAX = 22.300000 MEDIAN = 21.500000 AVG = 21.555200)
…
c9-1 CC_T_COMP_WATER_TEMP_OUT(MAX = 22.400000 MEDIAN = 19.800000 AVG = 19.830400)
System DC Power (MAX = 1541254 , AVG = 328265 , MEDIAN = 303892) Watts
    c0-0 CAB_DC_POWER(MAX = 74210 MEDIAN = 12519 AVG = 13644) Watts
    c0-1 CAB_DC_POWER(MAX = 74307 MEDIAN = 12336 AVG = 13523) Watts
    c1-0 CAB_DC_POWER(MAX = 73331 MEDIAN = 12465 AVG = 13617) Watts
    …
    c9-1 CAB_DC_POWER(MAX = 74563 MEDIAN = 12367 AVG = 13582) Watts
```

Sensor Name

Verbose Text Output

```
c0-0 CC_T_COMP_AMBIENT_TEMP0(MAX = 23.800000 MEDIAN = 23.400000 AVG = 23.297600)
c0-0 CC_T_COMP_AMBIENT_TEMP1(MAX = 22.800000 MEDIAN = 22.100000 AVG = 22.120000)
c0-1 CC_T_COMP_AMBIENT_TEMP0(MAX = 22.300000 MEDIAN = 21.500000 AVG = 21.555200)
…
c9-1 CC_T_COMP_WATER_TEMP_OUT(MAX = 22.400000 MEDIAN = 19.800000 AVG = 19.830400)
System DC Power (MAX = 1541254 , AVG = 328265 , MEDIAN = 303892) Watts
    c0-0 CAB_DC_POWER(MAX = 74210 MEDIAN = 12519 AVG = 13644) Watts
    c0-1 CAB_DC_POWER(MAX = 74307 MEDIAN = 12336 AVG = 13523) Watts
    c1-0 CAB_DC_POWER(MAX = 73331 MEDIAN = 12465 AVG = 13617) Watts
    …
    c9-1 CAB_DC_POWER(MAX = 74563 MEDIAN = 12367 AVG = 13582) Watts
```

Maximum, Median, Average sensor readings for the time window data was collection

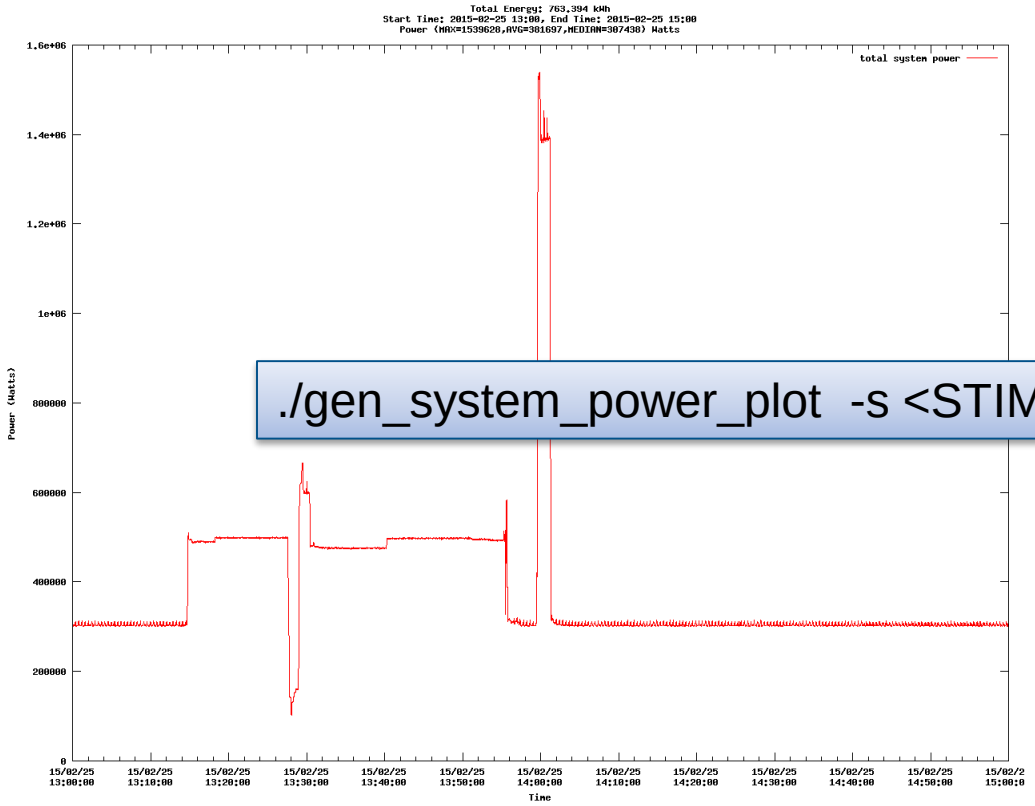
Verbose Text Output

```
c0-0 CC_T_COMP_AMBIENT_TEMP0(MAX = 23.800000 MEDIAN = 23.400000 AVG = 23.297600)
c0-0 CC_T_COMP_AMBIENT_TEMP1(MAX = 22.800000 MEDIAN = 22.100000 AVG = 22.120000)
c0-1 CC_T_COMP_AMBIENT_TEMP0(MAX = 22.300000 MEDIAN = 21.500000 AVG = 21.555200)
…
c9-1 CC_T_COMP_WATER_TEMP_OUT(MAX = 22.400000 MEDIAN = 19.800000 AVG = 19.830400)
System DC Power (MAX = 1541254 , AVG = 328265 , MEDIAN = 303892) Watts
    c0-0 CAB_DC_POWER(MAX = 74210 MEDIAN = 12519 AVG = 13644) Watts
    c0-1 CAB_DC_POWER(MAX = 74307 MEDIAN = 12336 AVG = 13523) Watts
    c1-0 CAB_DC_POWER(MAX = 73331 MEDIAN = 12465 AVG = 13617) Watts
    …
    c9-1 CAB_DC_POWER(MAX = 74563 MEDIAN = 12367 AVG = 13582) Watts
```

Maximum, Median, Average: System power (Cabinet DC + Blower Power)

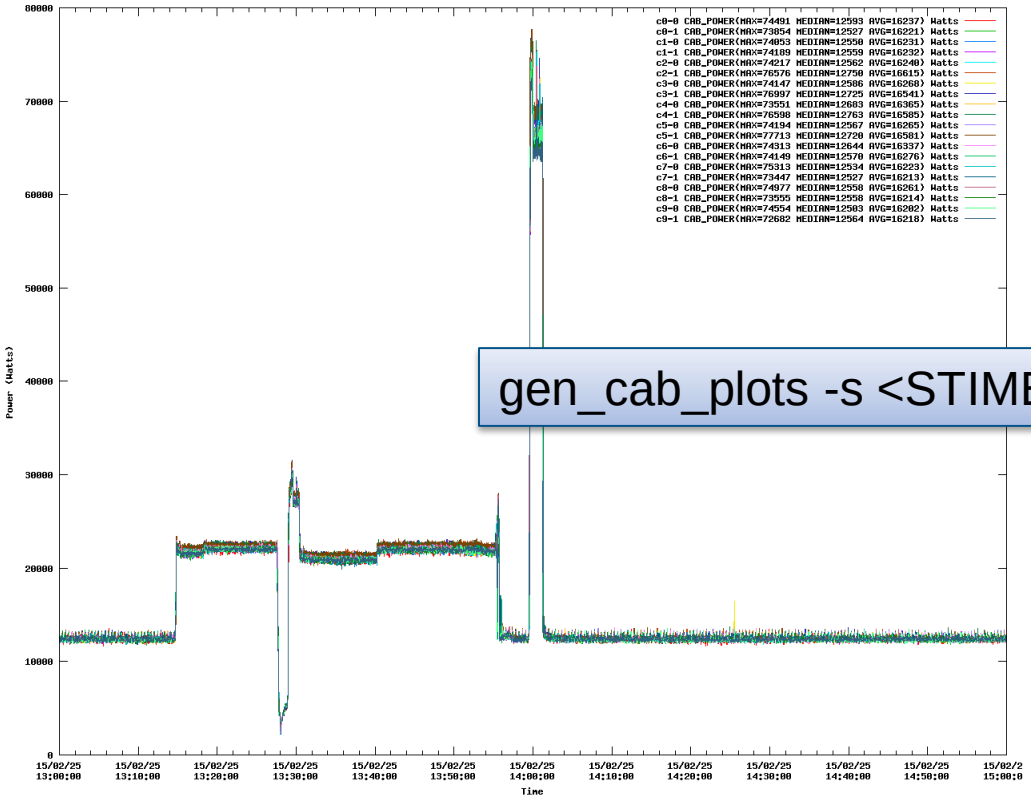
- Needs to be scaled to estimate AC power

System Power DC, 20 Cabinets Including Blowers



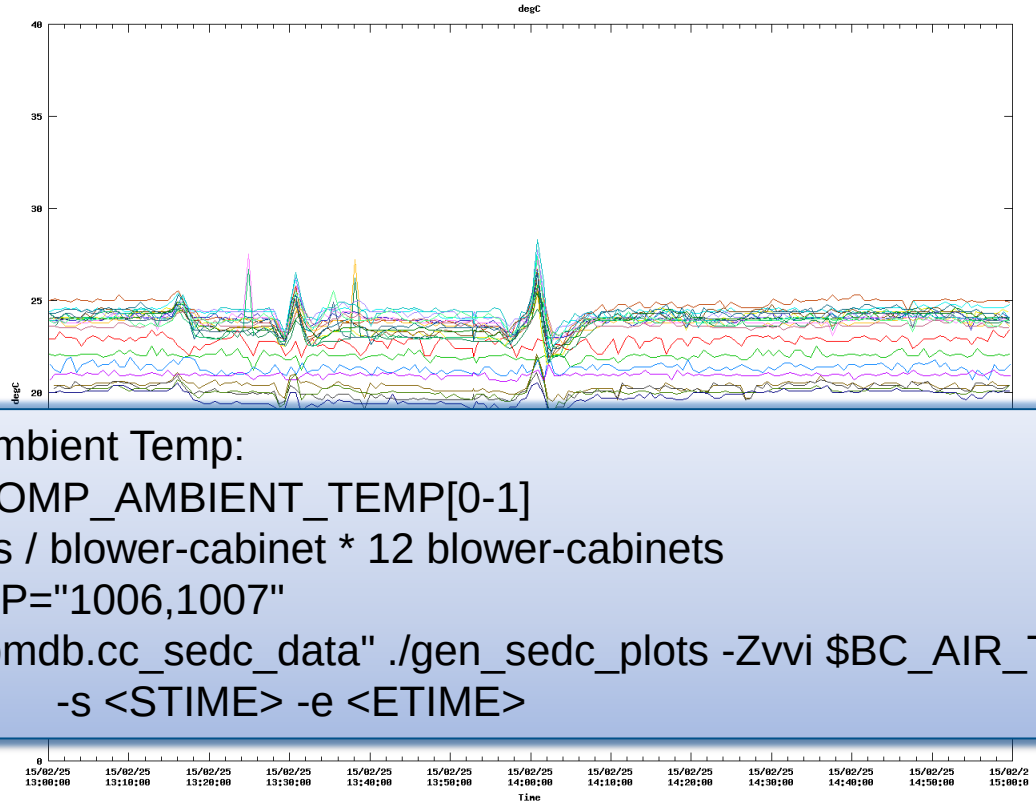
./gen_system_power_plot -s <STIME> -e <ETIME>

Cabinet Power (20 Cabinets)



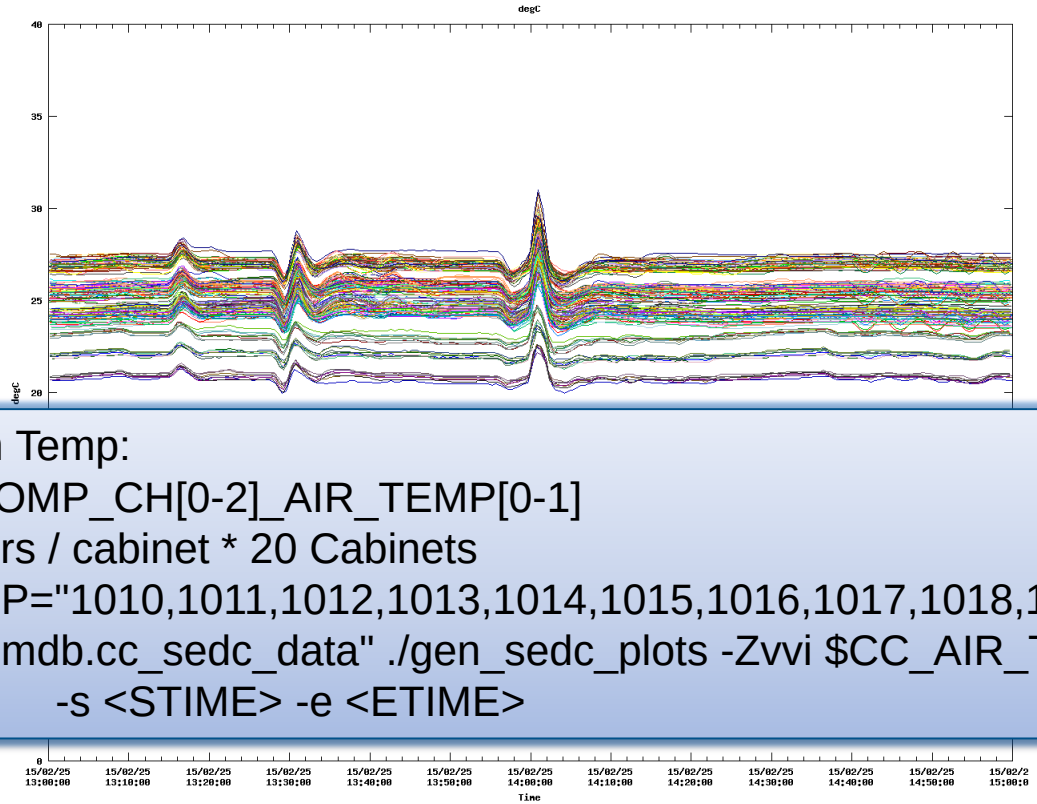
gen_cab_plots -s <STIME> -e <ETIME>

Blower Cab Ambient Temp



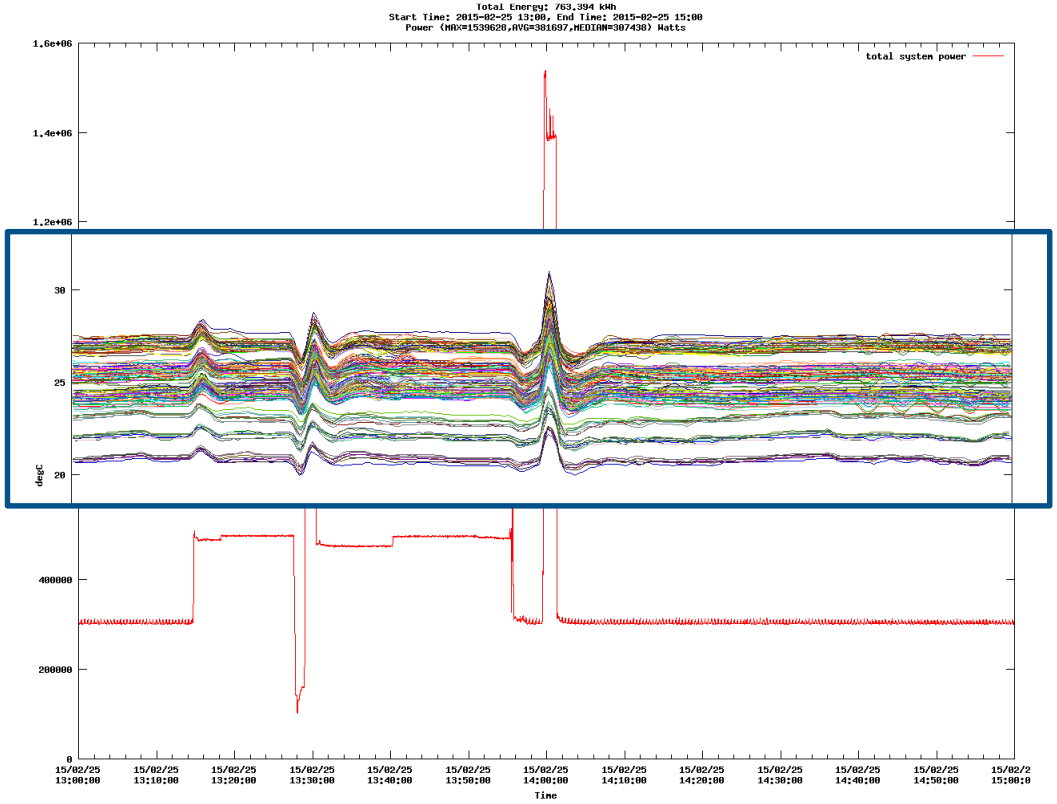
- Blower Cab Ambient Temp:
  - CC_T_COMP_AMBIENT_TEMP[0-1]
  - 2 sensors / blower-cabinet * 12 blower-cabinets
- BC_AIR_TEMP="1006,1007"
- PMDB_DB="pmdb.cc_sedc_data" ./gen_sedc_plots -Zvvi $BC_AIR_TEMP \
  -s <STIME> -e <ETIME>

CC Air Stream Temp



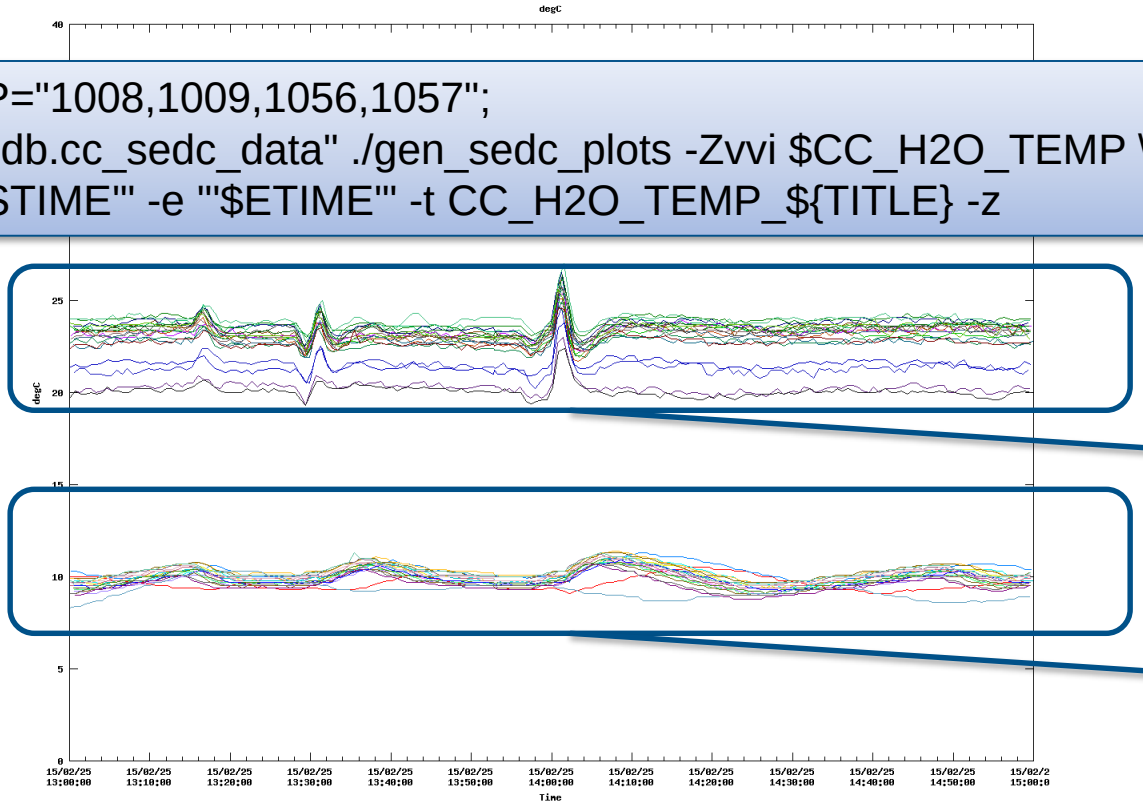
- CC Air Stream Temp:
  - CC_T_COMP_CH[0-2]_AIR_TEMP[0-1]
  - 12 sensors / cabinet * 20 Cabinets
- CC_AIR_TEMP="1010,1011,1012,1013,1014,1015,1016,1017,1018,1019,1020,1021";
- PMDB_DB="pmdb.cc_sedc_data" ./gen_sedc_plots -Zvvi $CC_AIR_TEMP \
  -s <STIME> -e <ETIME>

CC Air Stream Temp & System Power Plots



Water Temp In & Out

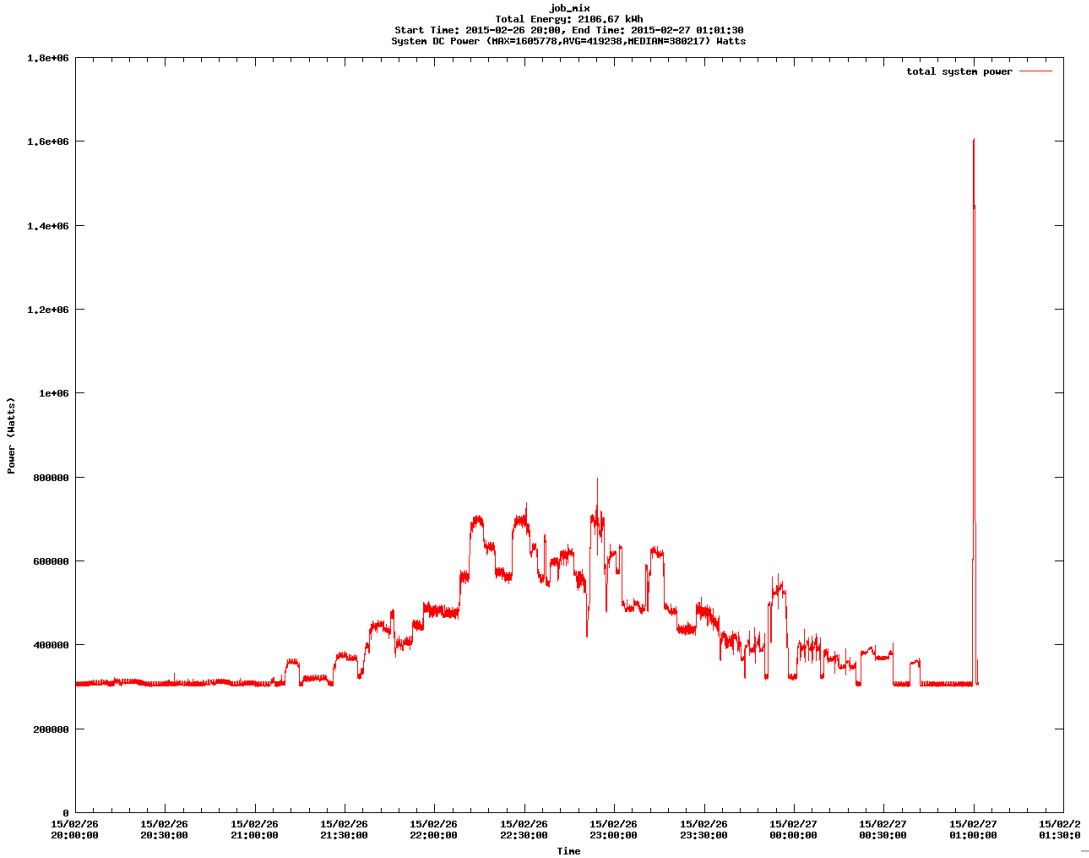
```
CC_H2O_TEMP="1008,1009,1056,1057";
PMDB_DB="pmdb.cc_sedc_data" ./gen_sedc_plots -Zvvi $CC_H2O_TEMP \
        -s "'$STIME'" -e "'$ETIME'" -t CC_H2O_TEMP_${TITLE} -z
```



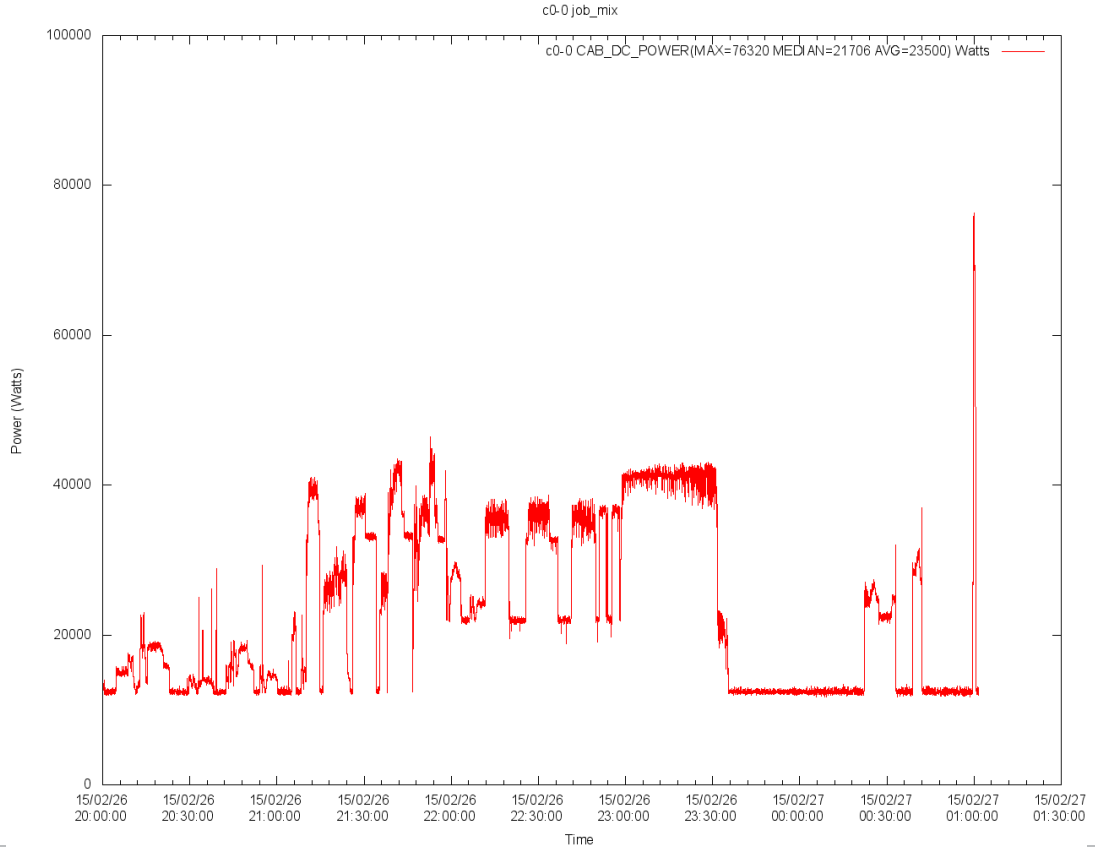
Outlet Water

Inlet Water

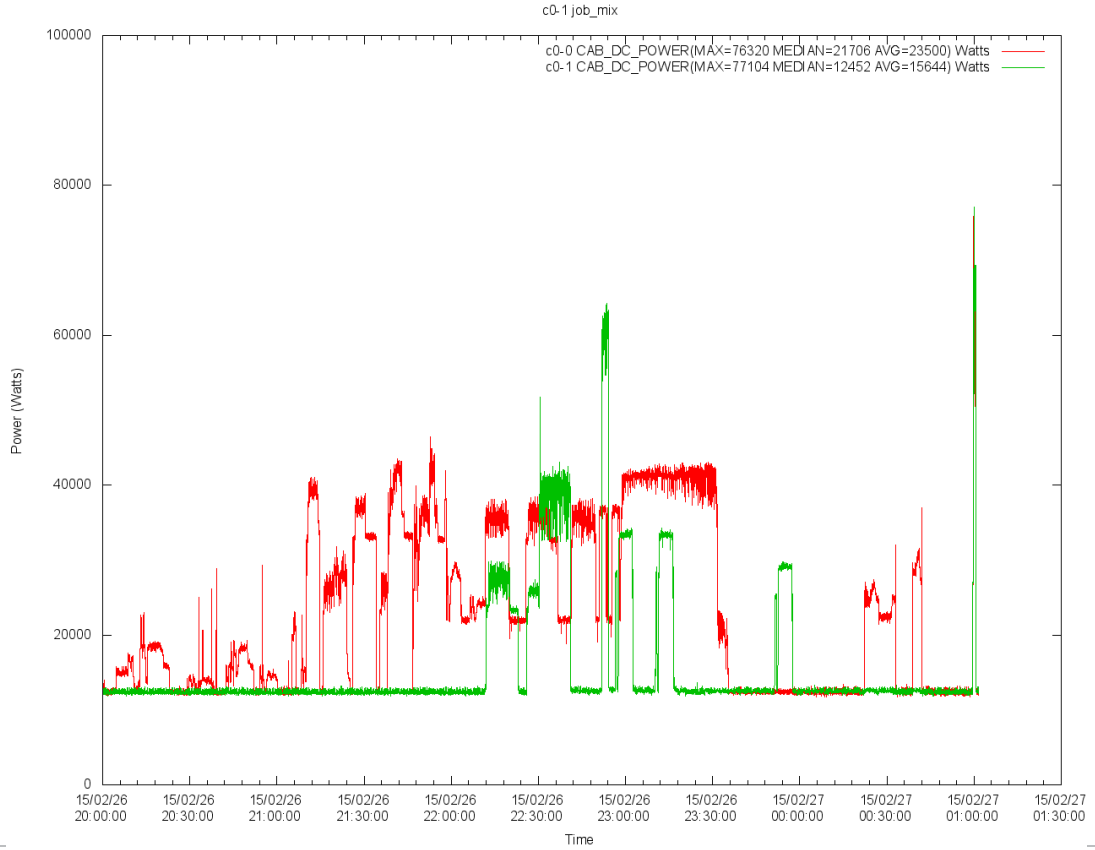
Job Mix: System Power



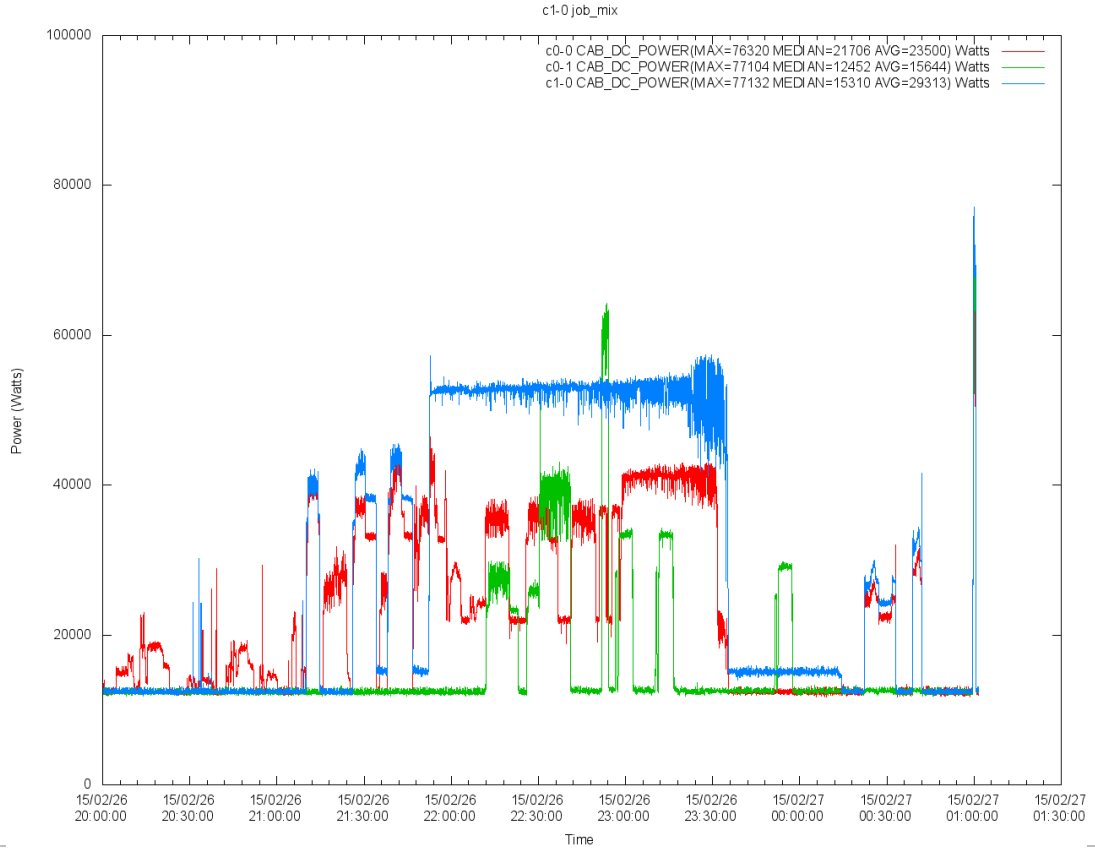
Job Mix: Cabinet Power



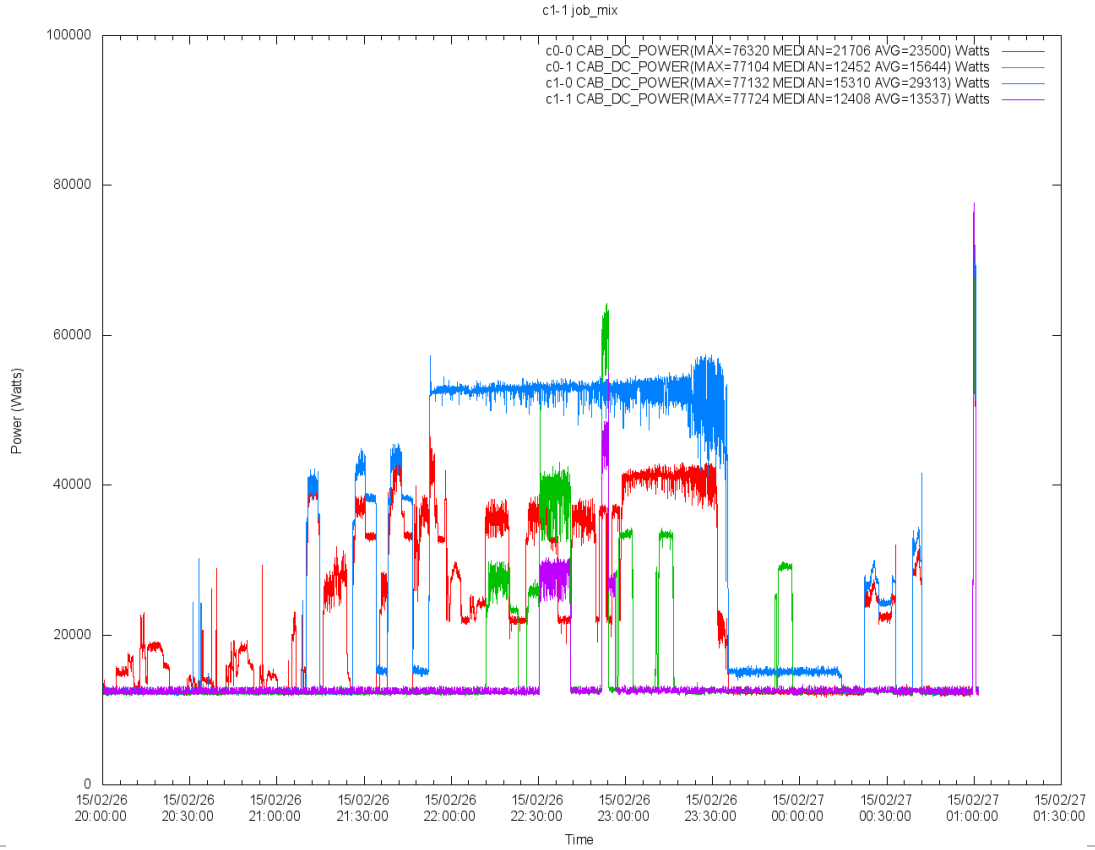
Job Mix: Cabinet Power



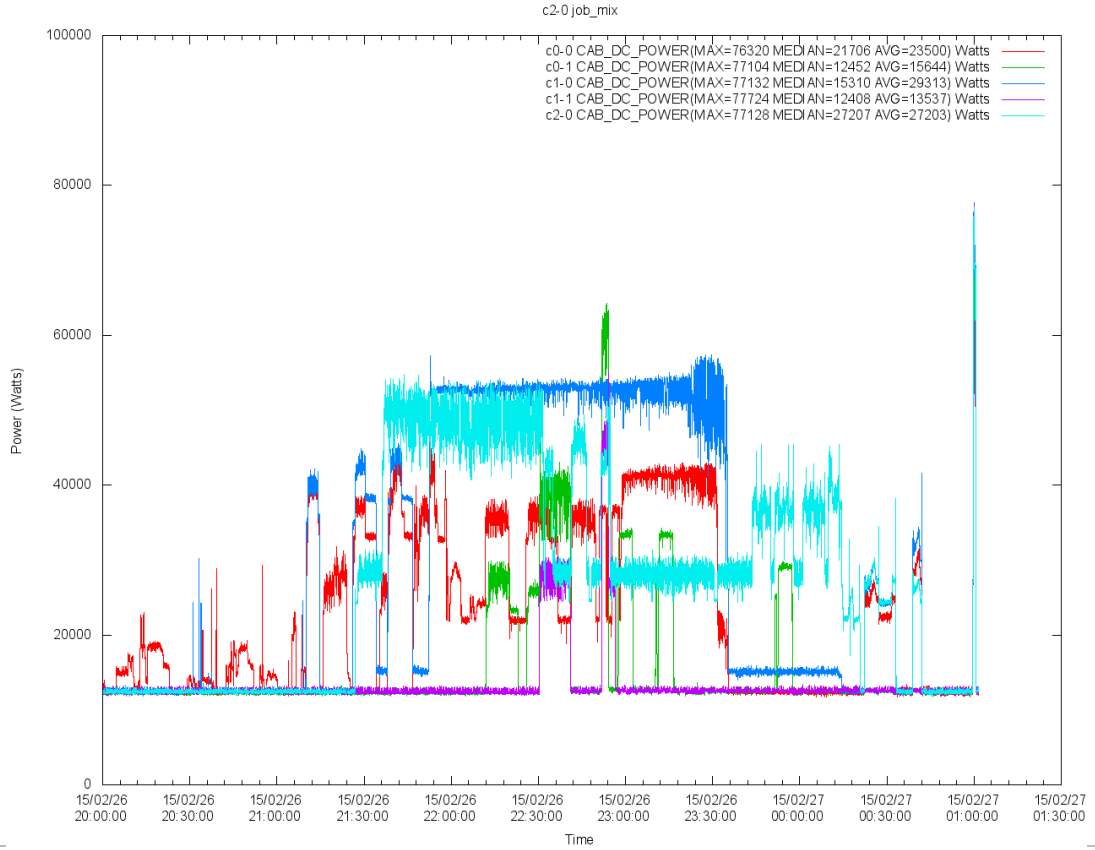
Job Mix: Cabinet Power



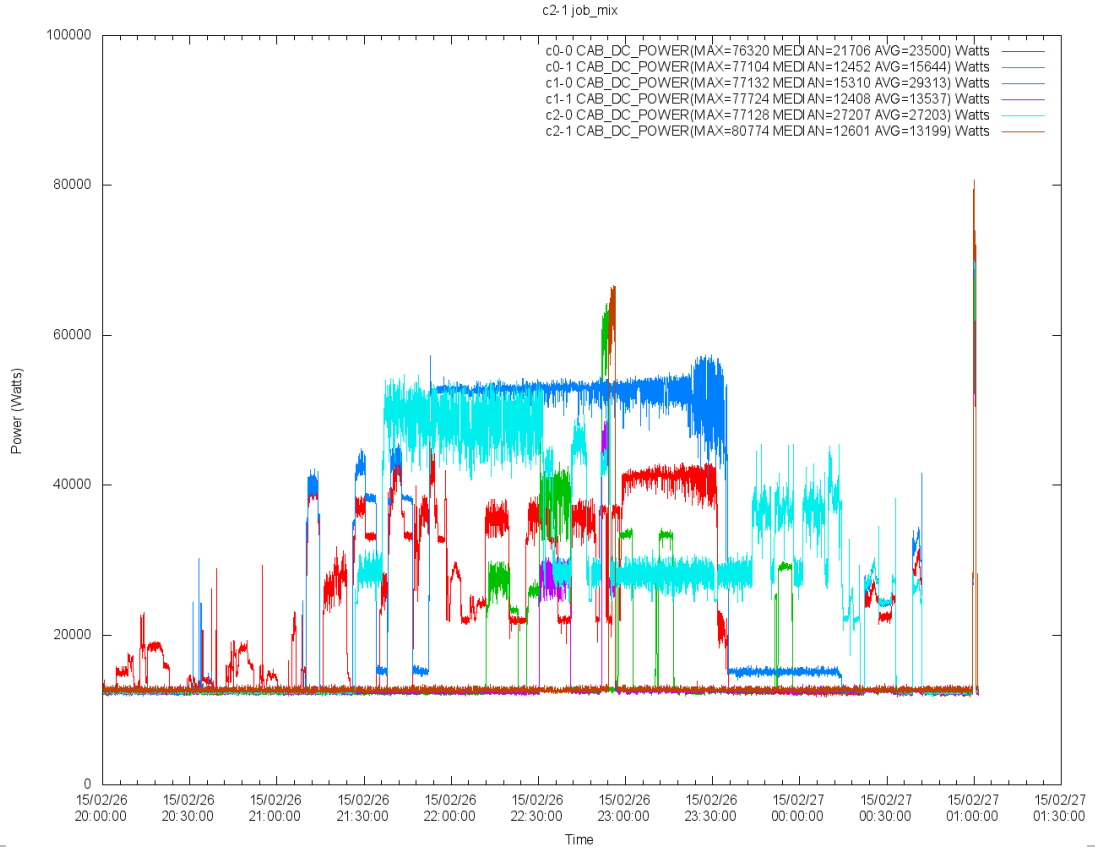
Job Mix: Cabinet Power



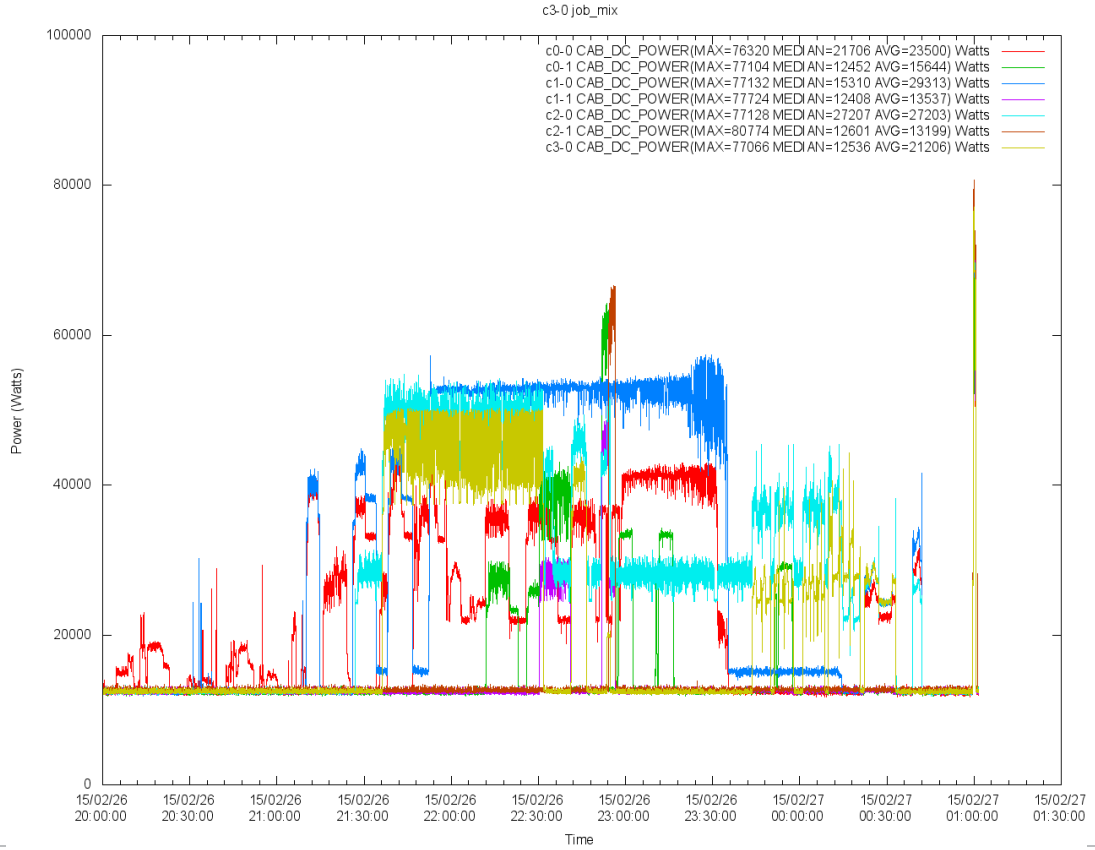
Job Mix: Cabinet Power



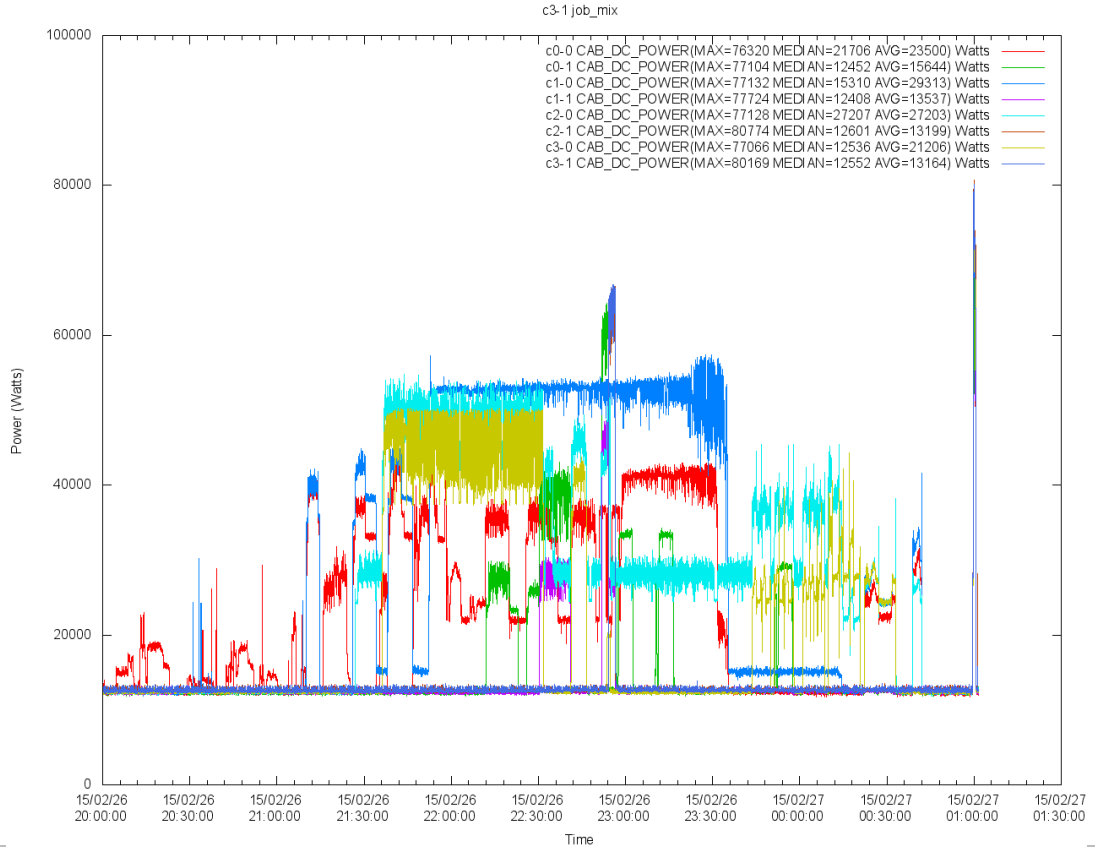
Job Mix: Cabinet Power



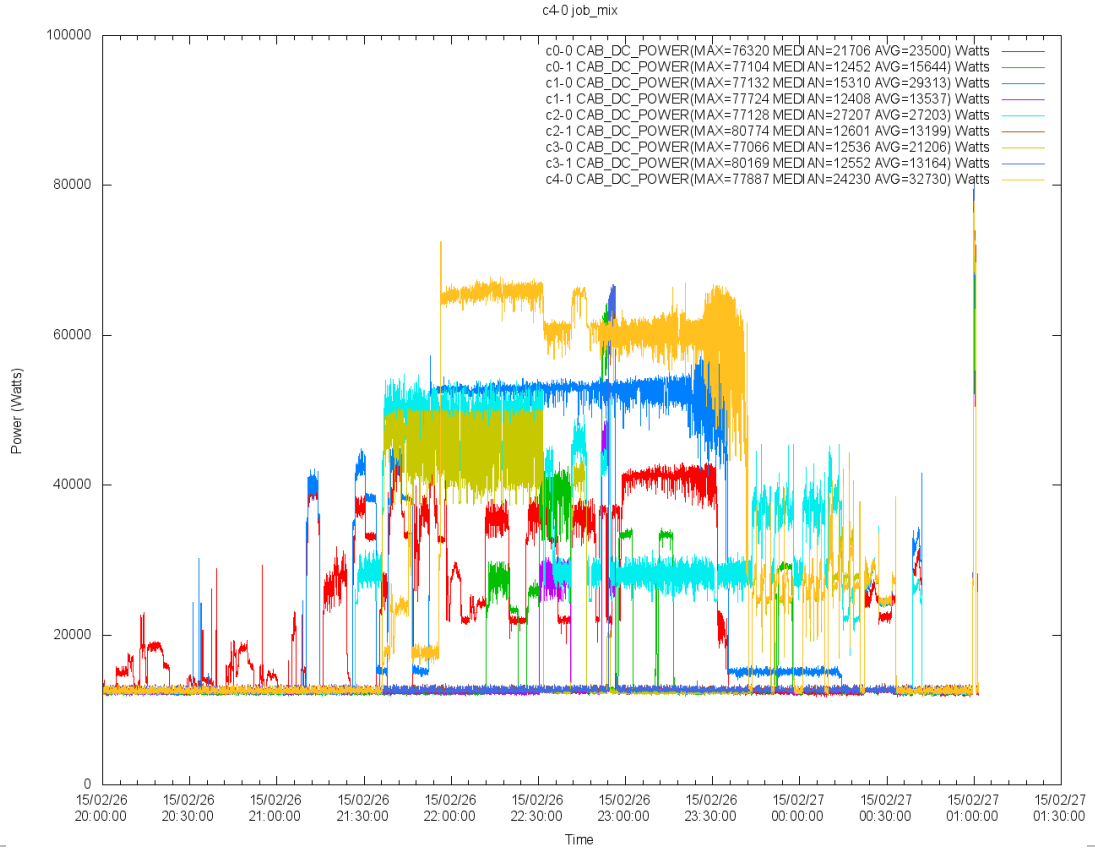
Job Mix: Cabinet Power



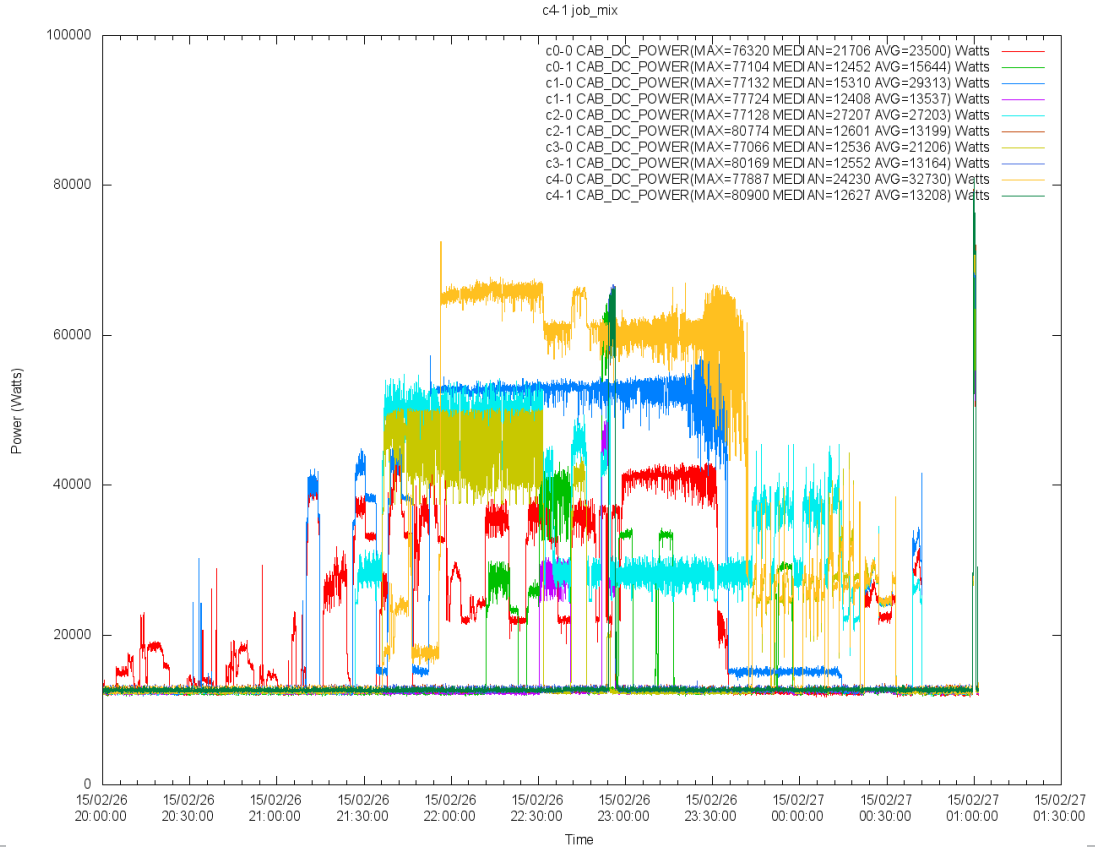
Job Mix: Cabinet Power



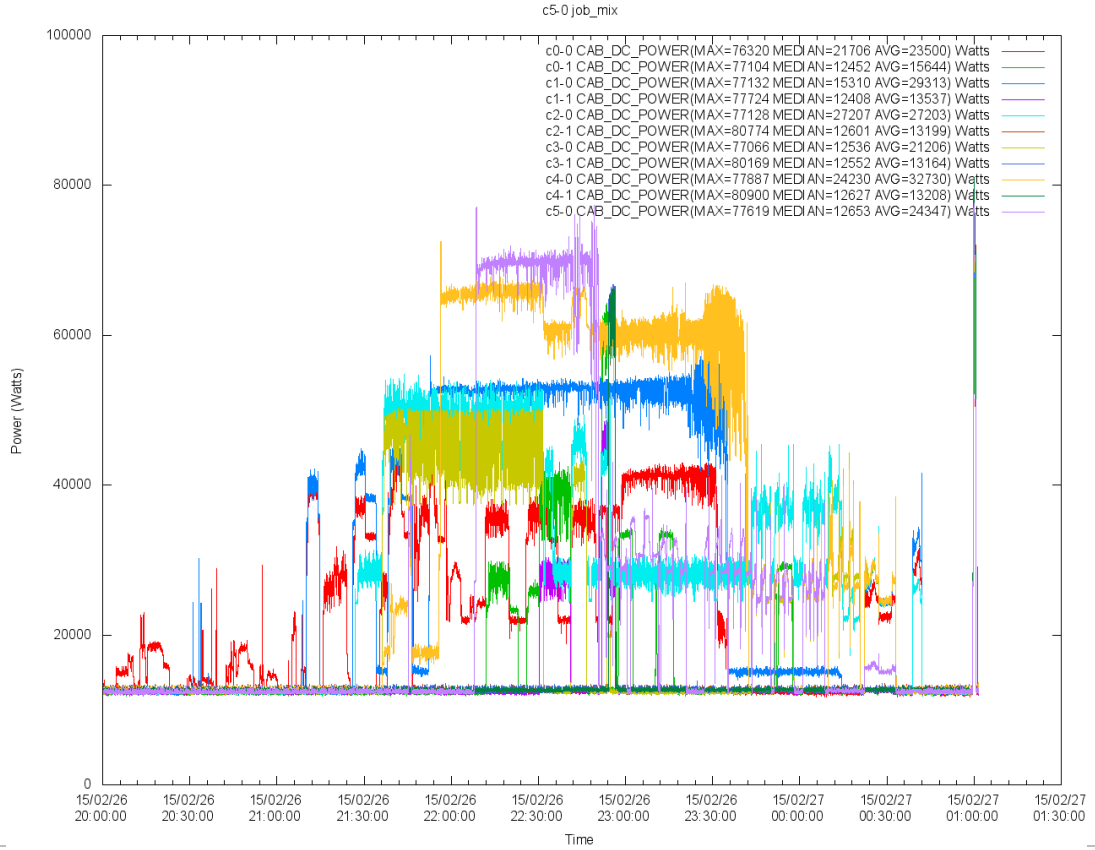
Job Mix: Cabinet Power



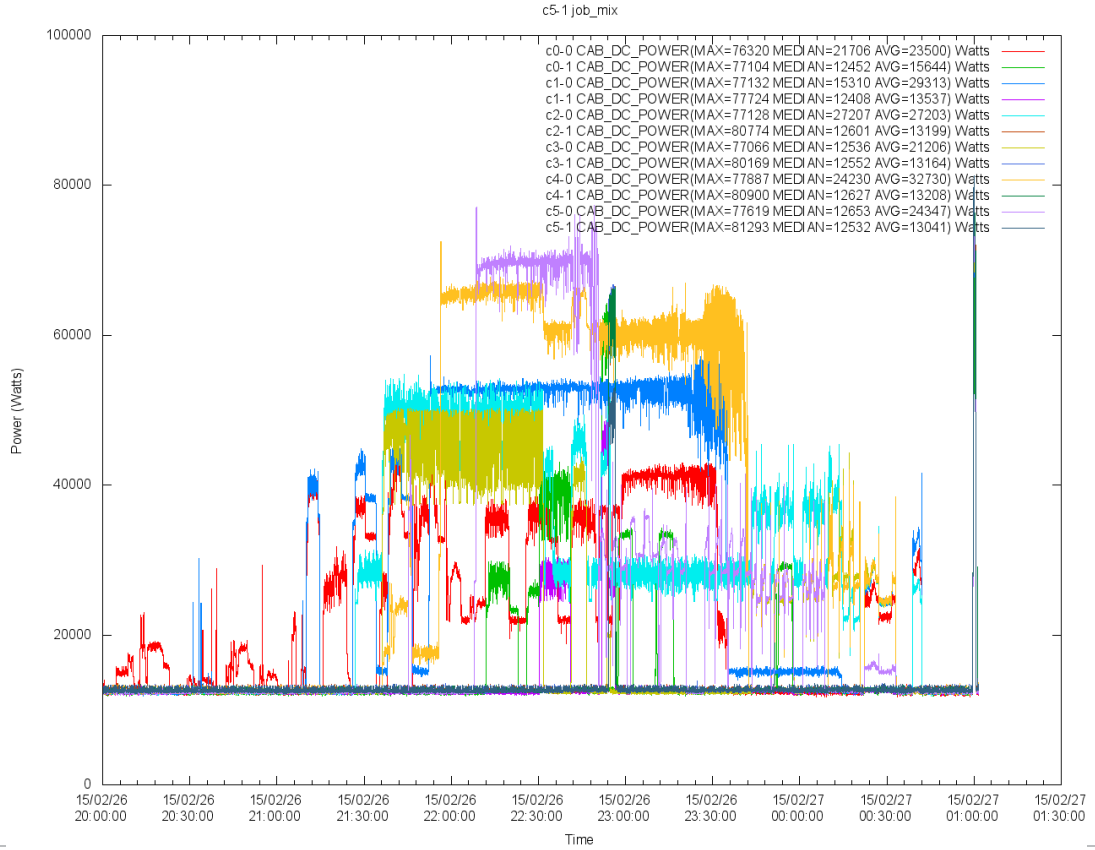
Job Mix: Cabinet Power



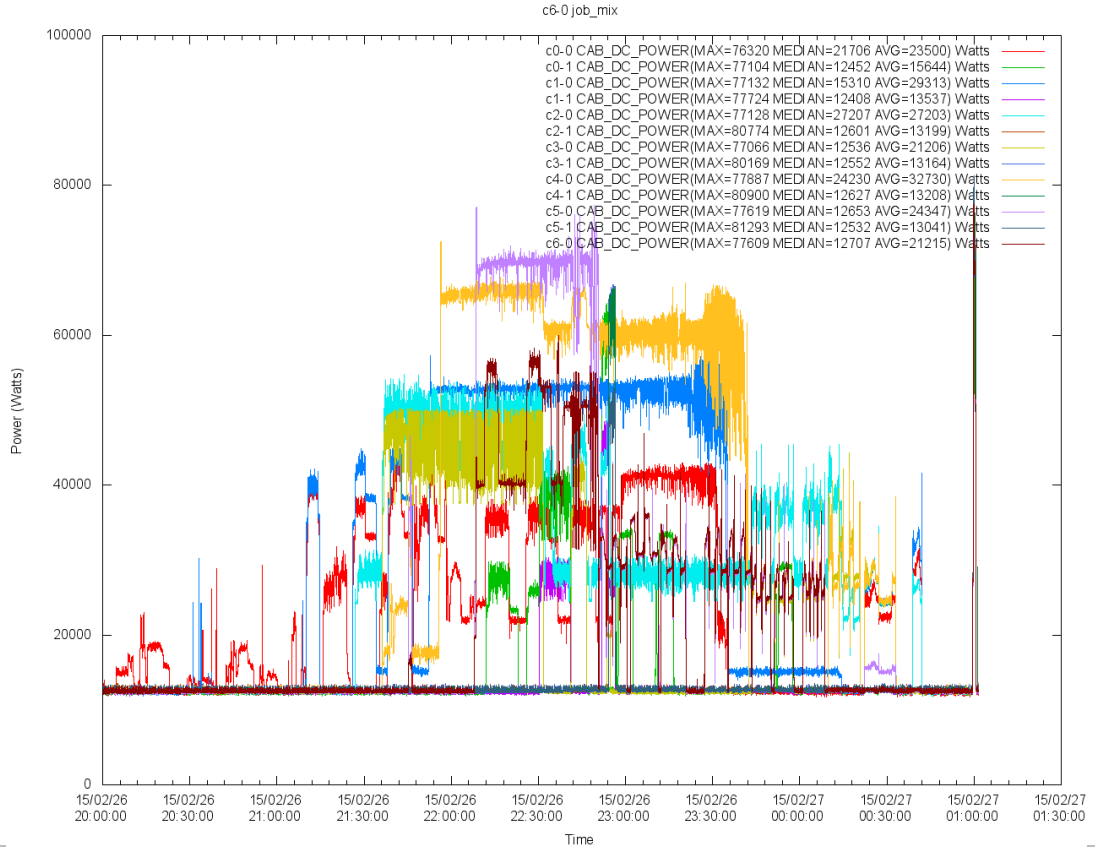
Job Mix: Cabinet Power



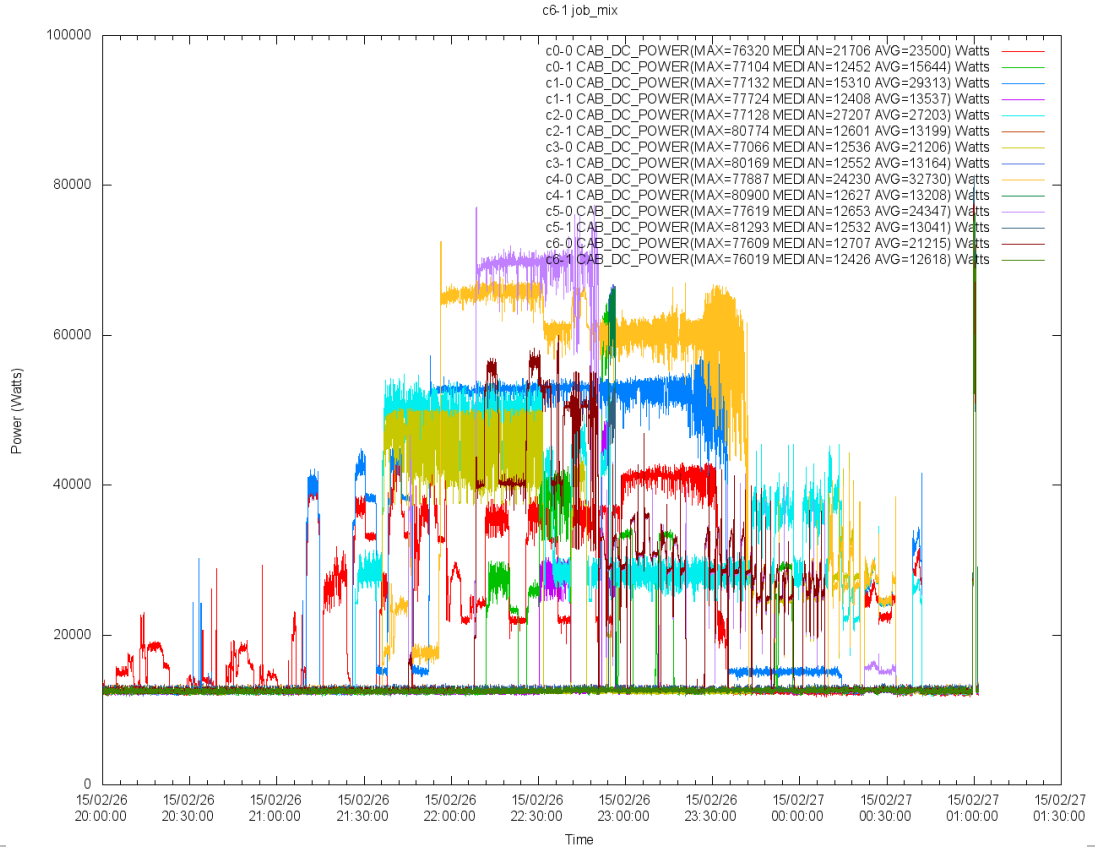
Job Mix: Cabinet Power



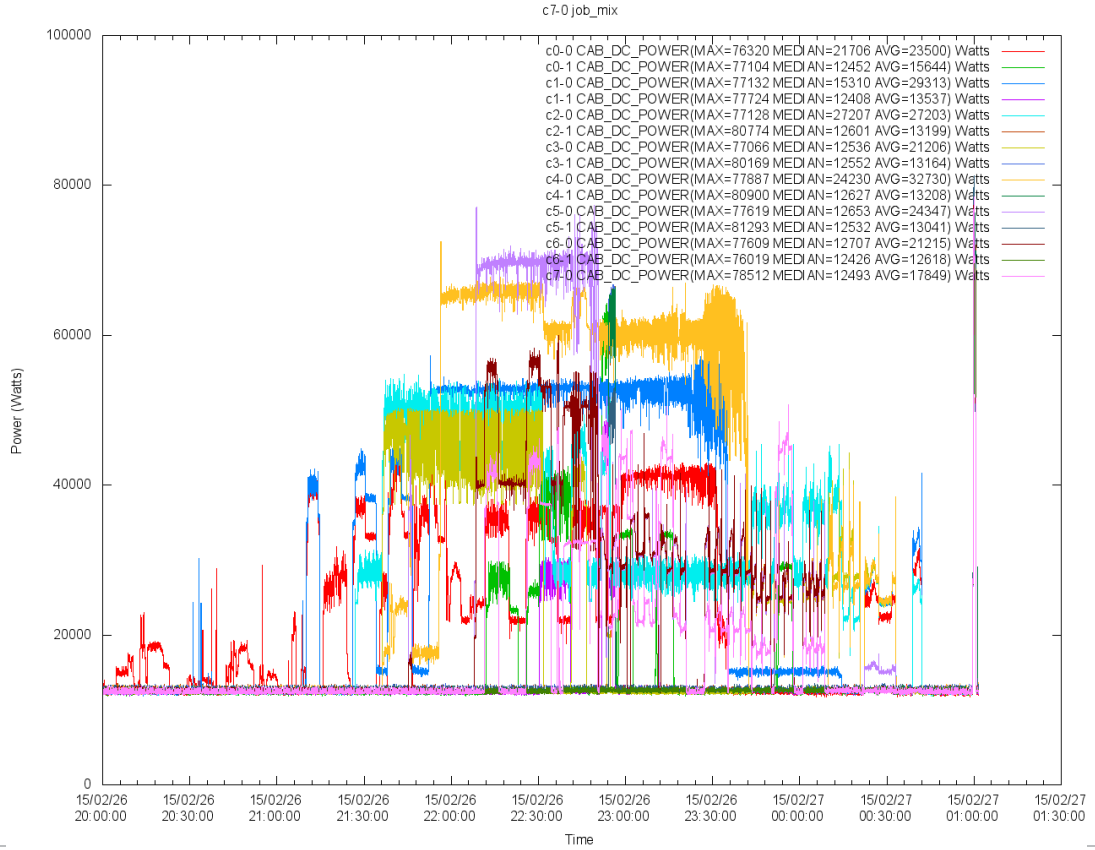
Job Mix: Cabinet Power



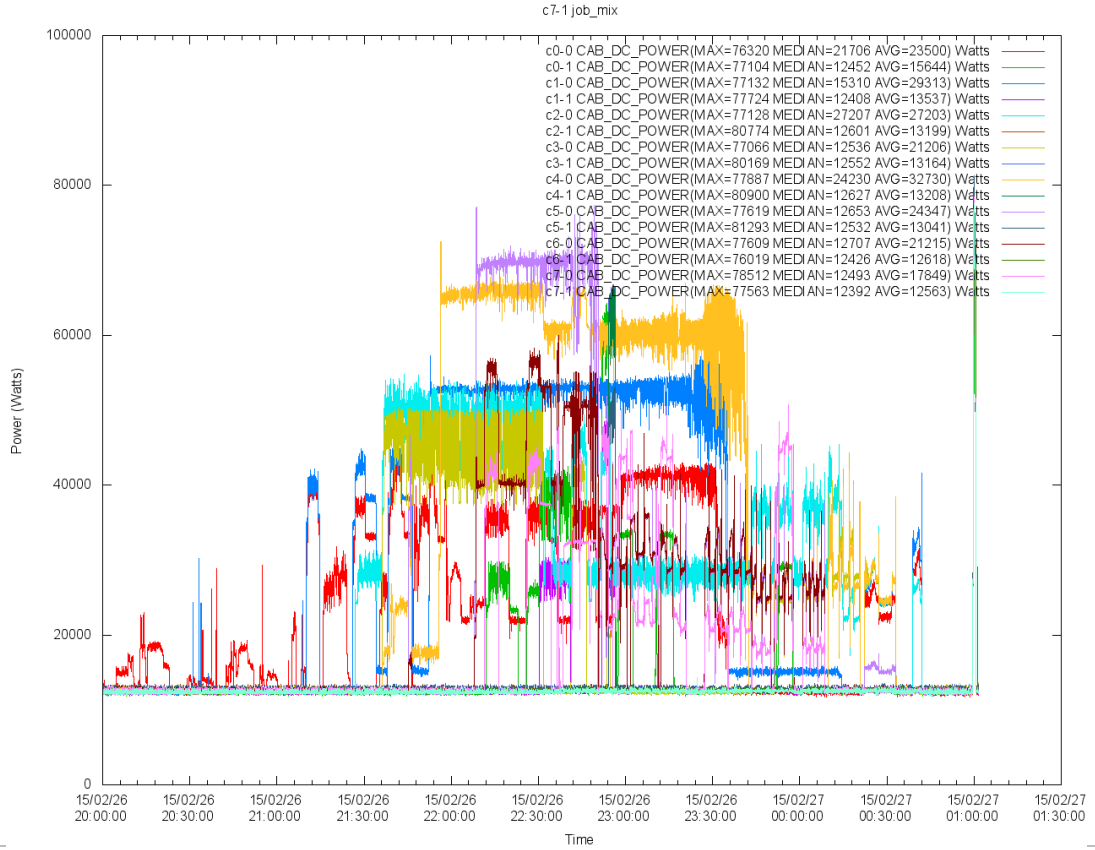
Job Mix: Cabinet Power



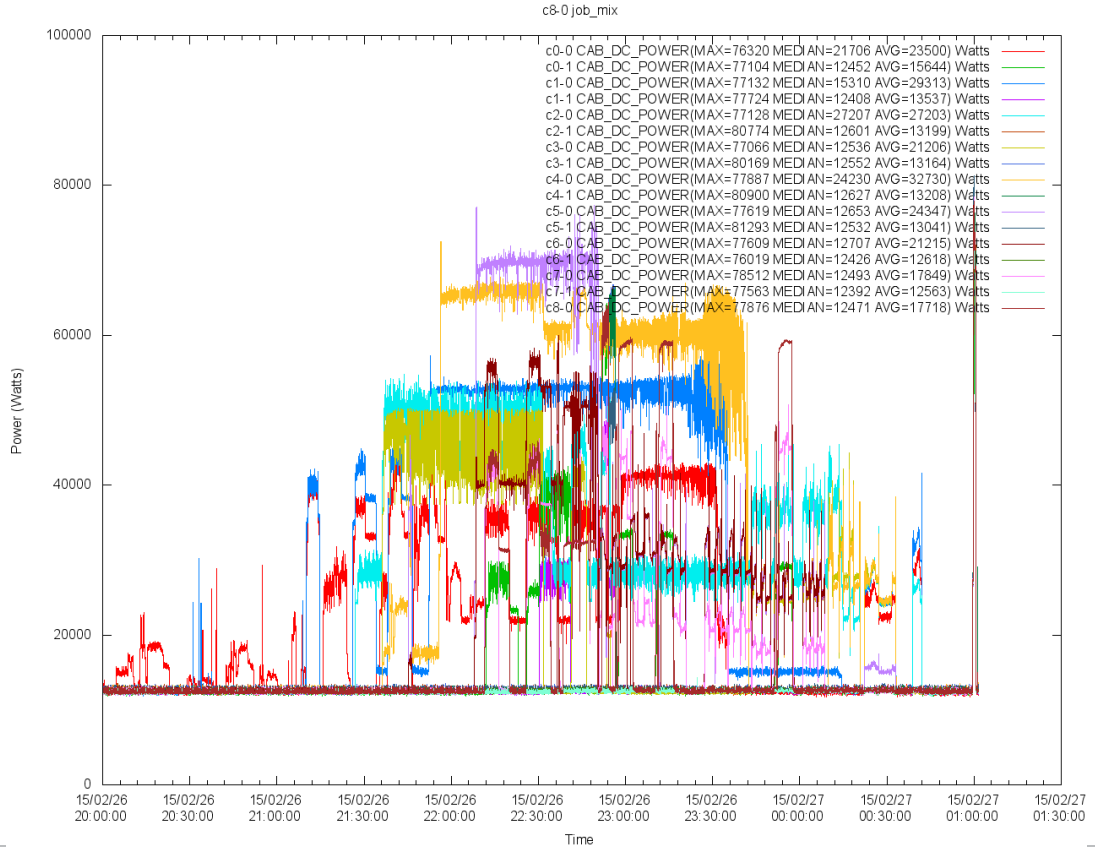
Job Mix: Cabinet Power



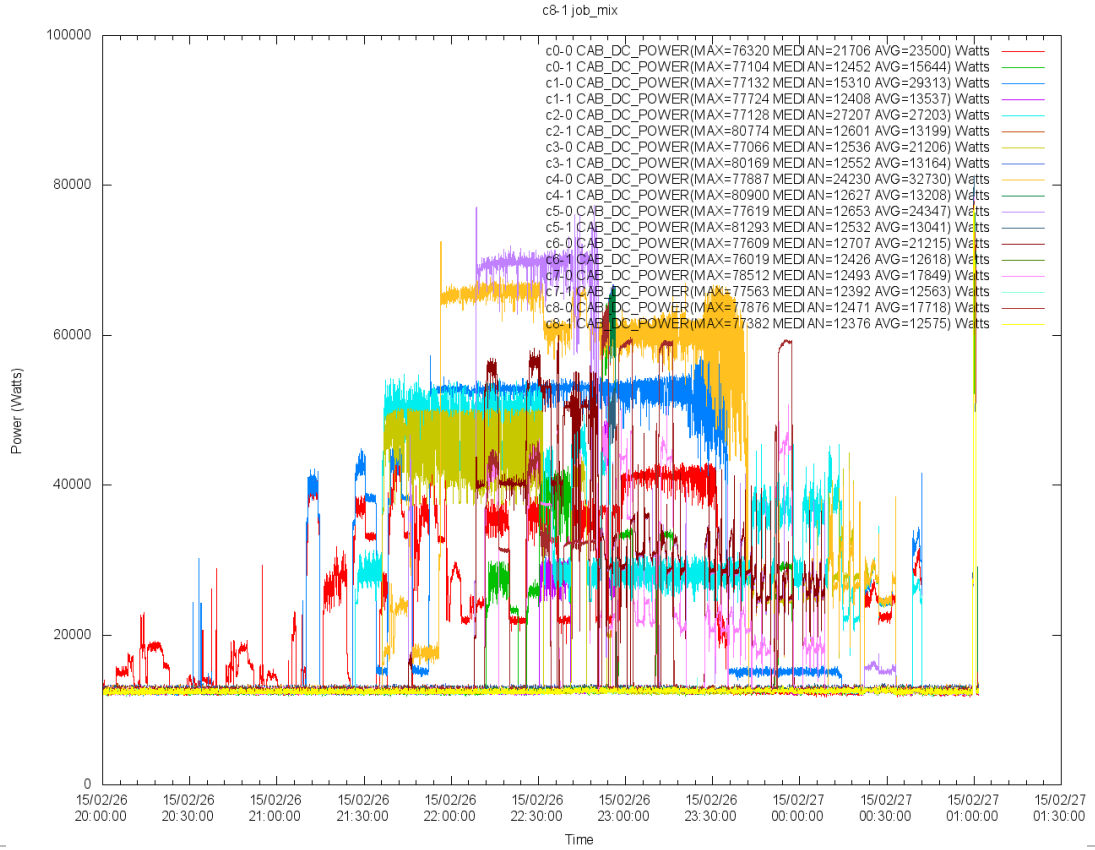
Job Mix: Cabinet Power



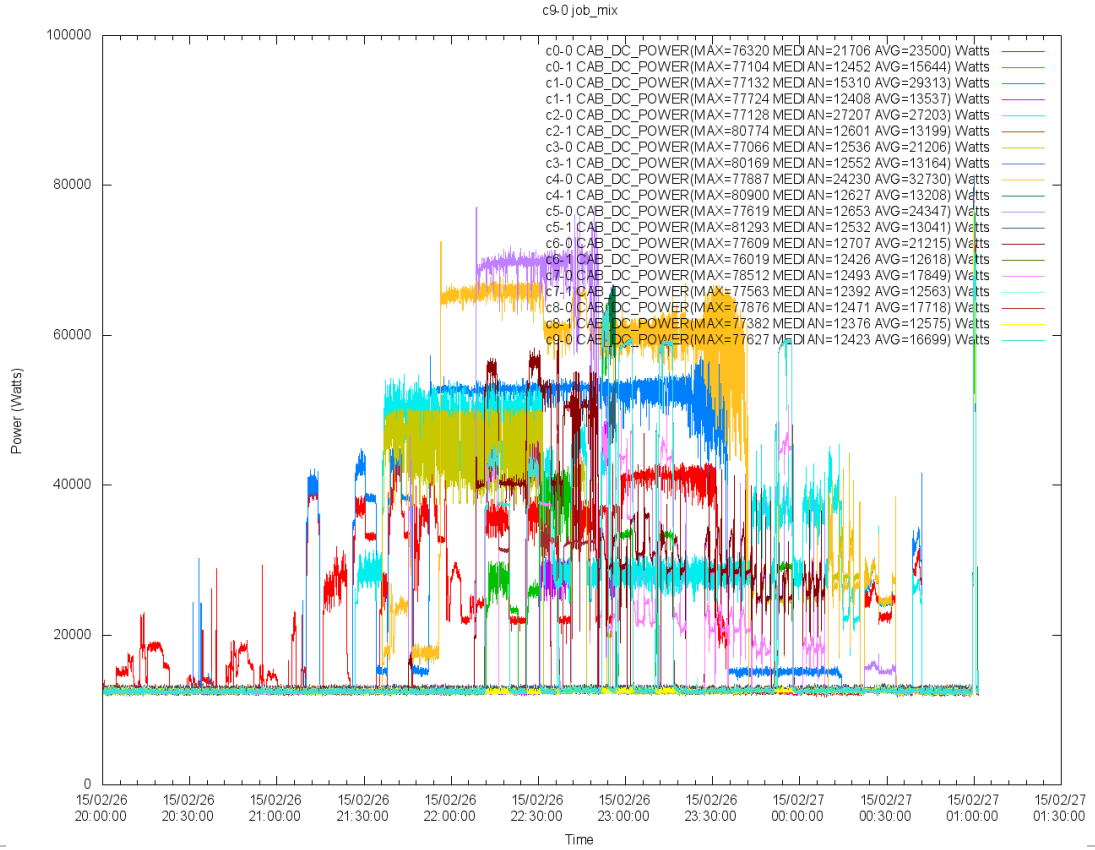
Job Mix: Cabinet Power



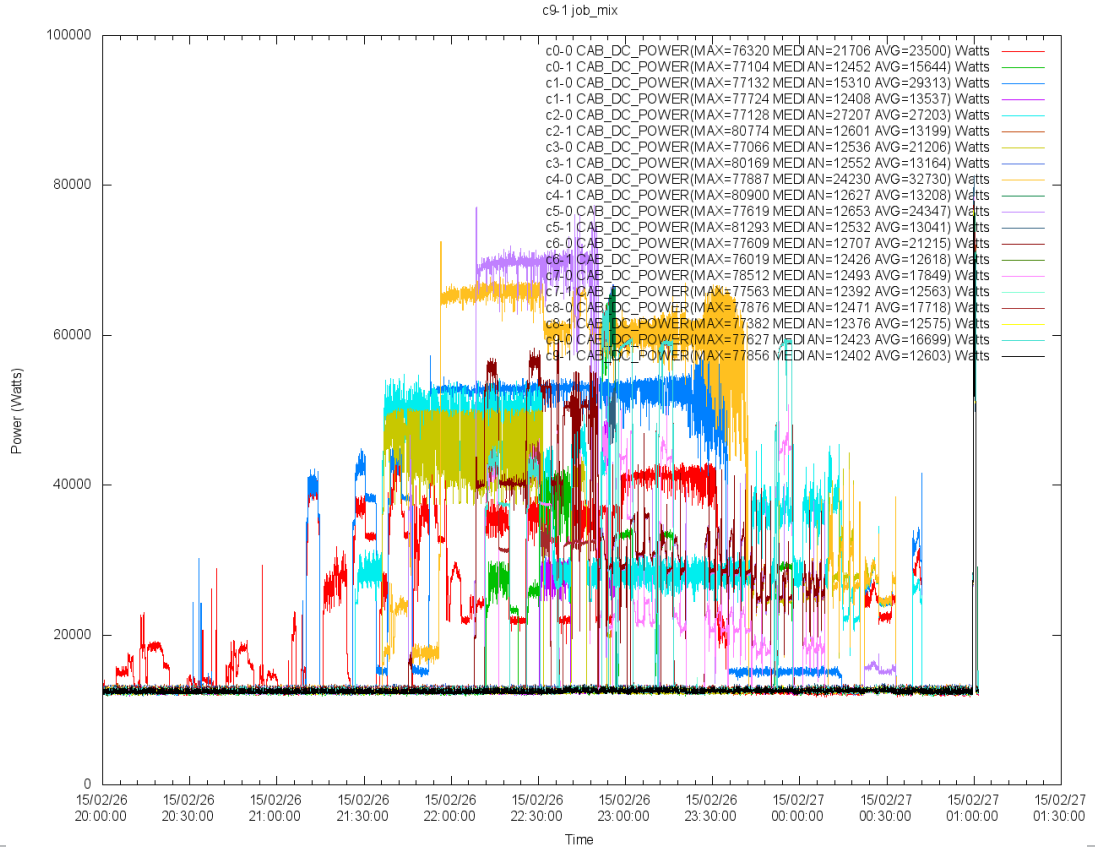
Job Mix: Cabinet Power



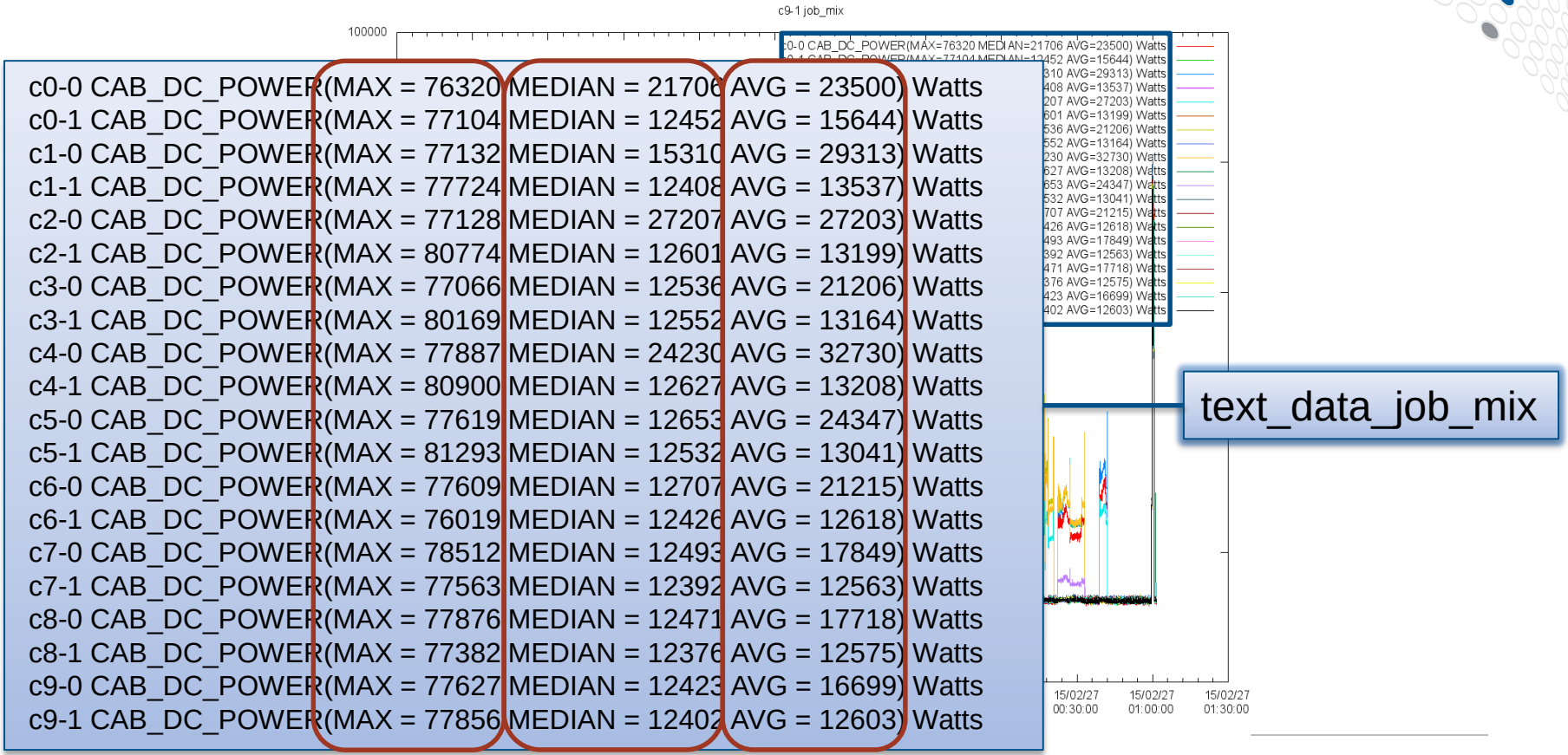
Job Mix: Cabinet Power



Job Mix: Cabinet Power



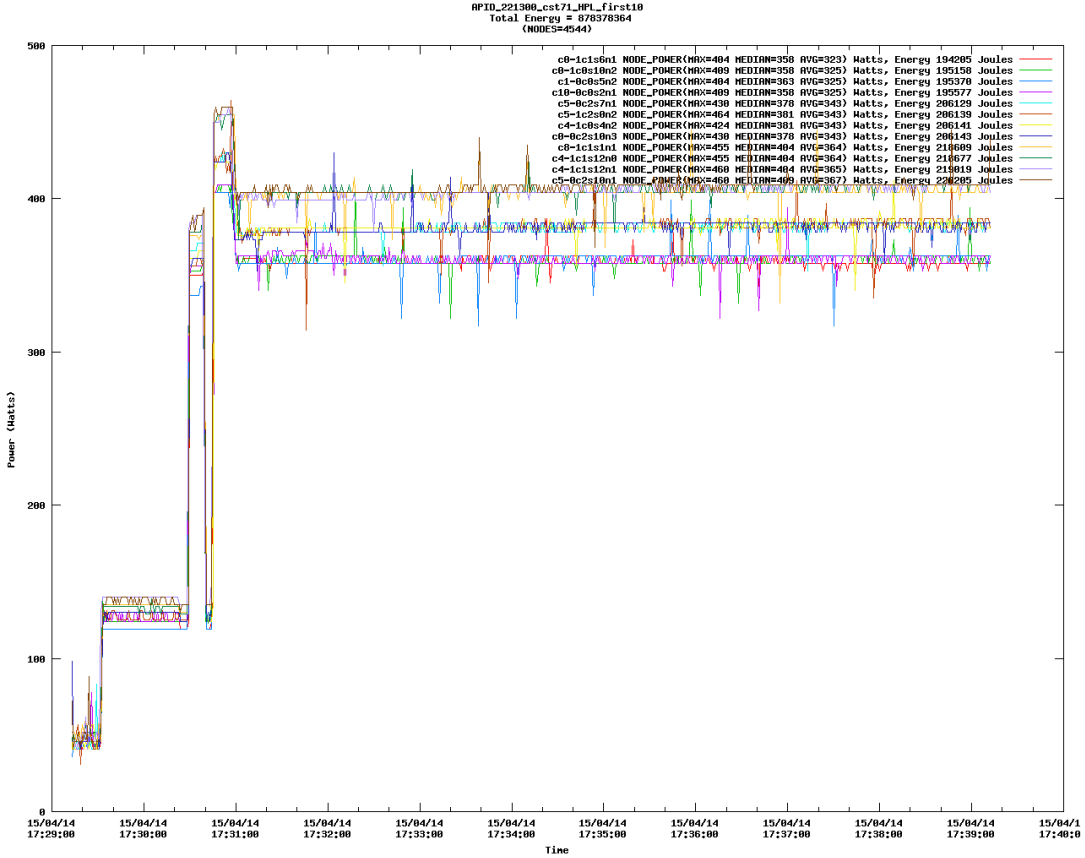
Job Mix: Cabinet Power



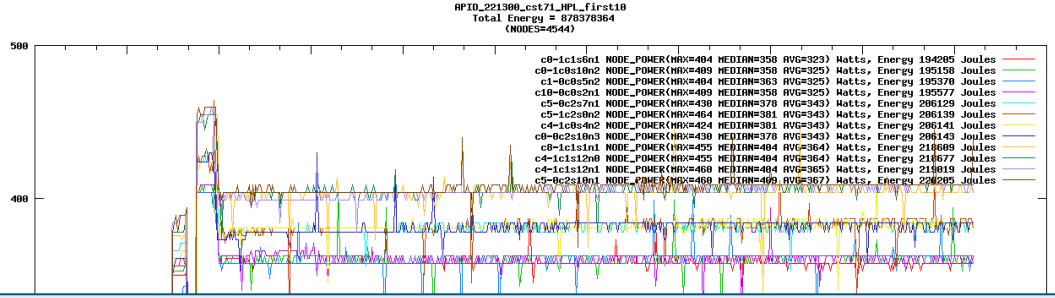
c0-0 CAB_DC_POWER(MAX = 76320 MEDIAN = 21706 AVG = 23500) Watts
c0-1 CAB_DC_POWER(MAX = 77104 MEDIAN = 12452 AVG = 15644) Watts
c1-0 CAB_DC_POWER(MAX = 77132 MEDIAN = 15310 AVG = 29313) Watts
c1-1 CAB_DC_POWER(MAX = 77724 MEDIAN = 12408 AVG = 13537) Watts
c2-0 CAB_DC_POWER(MAX = 77128 MEDIAN = 27207 AVG = 27203) Watts
c2-1 CAB_DC_POWER(MAX = 80774 MEDIAN = 12601 AVG = 13199) Watts
c3-0 CAB_DC_POWER(MAX = 77066 MEDIAN = 12536 AVG = 21206) Watts
c3-1 CAB_DC_POWER(MAX = 80169 MEDIAN = 12552 AVG = 13164) Watts
c4-0 CAB_DC_POWER(MAX = 77887 MEDIAN = 24230 AVG = 32730) Watts
c4-1 CAB_DC_POWER(MAX = 80900 MEDIAN = 12627 AVG = 13208) Watts
c5-0 CAB_DC_POWER(MAX = 77619 MEDIAN = 12653 AVG = 24347) Watts
c5-1 CAB_DC_POWER(MAX = 81293 MEDIAN = 12532 AVG = 13041) Watts
c6-0 CAB_DC_POWER(MAX = 77609 MEDIAN = 12707 AVG = 21215) Watts
c6-1 CAB_DC_POWER(MAX = 76019 MEDIAN = 12426 AVG = 12618) Watts
c7-0 CAB_DC_POWER(MAX = 78512 MEDIAN = 12493 AVG = 17849) Watts
c7-1 CAB_DC_POWER(MAX = 77563 MEDIAN = 12392 AVG = 12563) Watts
c8-0 CAB_DC_POWER(MAX = 77876 MEDIAN = 12471 AVG = 17718) Watts
c8-1 CAB_DC_POWER(MAX = 77382 MEDIAN = 12376 AVG = 12575) Watts
c9-0 CAB_DC_POWER(MAX = 77627 MEDIAN = 12423 AVG = 16699) Watts
c9-1 CAB_DC_POWER(MAX = 77856 MEDIAN = 12402 AVG = 12603) Watts

text_data_job_mix

HPL on 4544 Nodes: First 10 Minutes

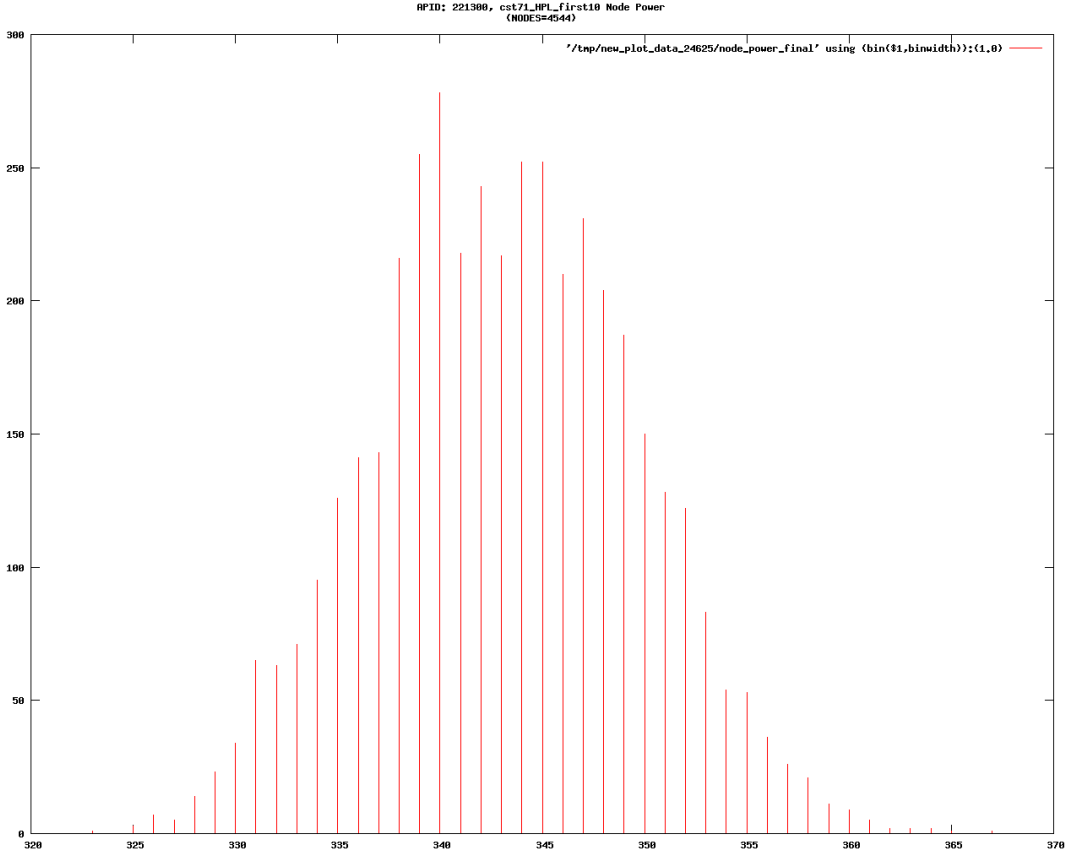


HPL on 4544 Nodes: First 10 Minutes




```
c0-1c1s6n1  NODE_POWER(MAX=404 MEDIAN=358 AVG=323) Watts, Energy 194205 Joules
c0-1c0s10n2 NODE_POWER(MAX=409 MEDIAN=358 AVG=325) Watts, Energy 195158 Joules
c1-0c0s5n2  NODE_POWER(MAX=404 MEDIAN=363 AVG=325) Watts, Energy 195370 Joules
c10-0c0s2n1 NODE_POWER(MAX=409 MEDIAN=358 AVG=325) Watts, Energy 195577 Joules
c5-0c2s7n1  NODE_POWER(MAX=430 MEDIAN=378 AVG=343) Watts, Energy 206129 Joules
c5-1c2s0n2  NODE_POWER(MAX=464 MEDIAN=381 AVG=343) Watts, Energy 206139 Joules
c4-1c0s4n2  NODE_POWER(MAX=424 MEDIAN=381 AVG=343) Watts, Energy 206141 Joules
c0-0c2s10n3 NODE_POWER(MAX=430 MEDIAN=378 AVG=343) Watts, Energy 206143 Joules
c8-1c1s1n1  NODE_POWER(MAX=455 MEDIAN=404 AVG=364) Watts, Energy 218609 Joules
c4-1c1s12n0 NODE_POWER(MAX=455 MEDIAN=404 AVG=364) Watts, Energy 218677 Joules
c4-1c1s12n1 NODE_POWER(MAX=460 MEDIAN=404 AVG=365) Watts, Energy 219019 Joules
c5-0c2s10n1 NODE_POWER(MAX=460 MEDIAN=409 AVG=367) Watts, Energy 220205 Joules
```

HPL on 4544 Nodes: First 10 Minutes



SQL & PMDB

Content from the 2015 CUG tutorial
(with minor updates)

PMDB Database Overview

- **PostgreSQL**
  - PostgreSQL 9.1.12
- **Round-robin data storage…**
  - Configurable: number of partitions, and rows/partition
  - Oldest data (partition) dropped to make room for new…
  - Hook script called when new partitions are created
    - Enables site-level customization

PMDB Database

- **PMDB power & energy**
  - Blade, Node, and Cabinet level power and energy data
  - Job/App ID, start time, end time, and node list data
  - Includes power/energy monitoring for accelerated blades

- **SEDC system & blade environmental data**
  - Cabinet inlet & outlet water temp and pressure, outlet air temp(s)
  - Blade level component data (temp, voltage, current)

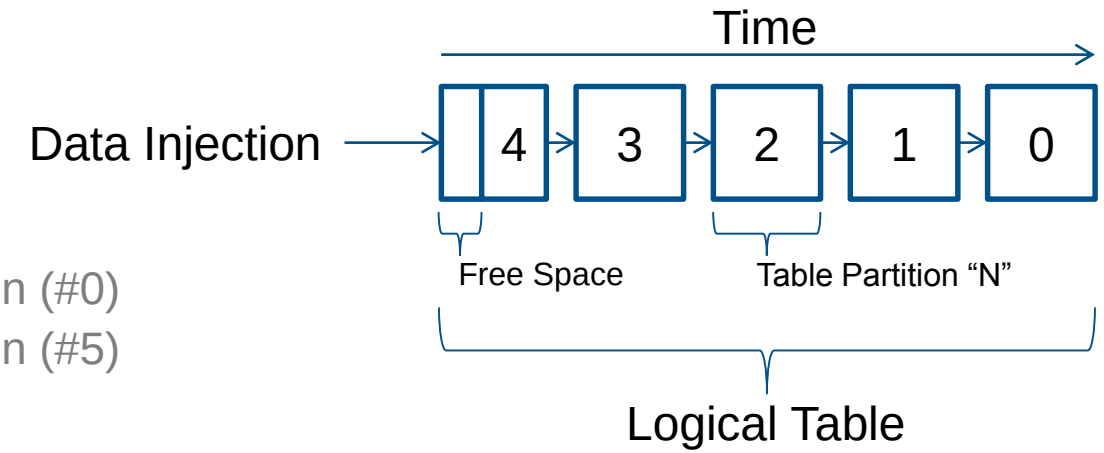
SEDC → PMDB: Now the default R/R
- SEDC → flat files are deprecated, dropped support in R/R UP02…

Time Series Data Management

- **Why use SQL database?**
  - Databases store data… (lots of data)
  - Flexible query interface
- **Answer questions**
  - What was max power draw of any node?
  - When did socket temperature reach 50°C?
  - What was average system power last Monday?
- **Difficulties**
  - Data aging
  - Continuous input stream

Data Management

- **Time series data**
  - Continuous stream of input
  - Can't store data indefinitely
  - Age data out using table partitioning
- **Data lifecycle**
  - If no free space
    - Drop oldest partition (#0)
    - Create new partition (#5)
  - Write to free space



Data Management (continued) Table Partitioning

- **Define table prototype**
  - CREATE TABLE pmdb.bc_data (
  - ts TIMESTAMPTZ, source INT,
  - id INT, value BIGINT);
- **Create partitions via inheritance**
  - CREATE TABLE pmdb.bc_data_1 ( ) INHERITS (pmdb.bc_data);
- **Issue queries against “prototype” table**
  - SELECT * FROM pmdb.bc_data WHERE …;
- **“Age out” data by dropping partition & index**
  - DROP TABLE pmdb.bc_data_1 CASCADE;

Database Hooks (1 of 5)

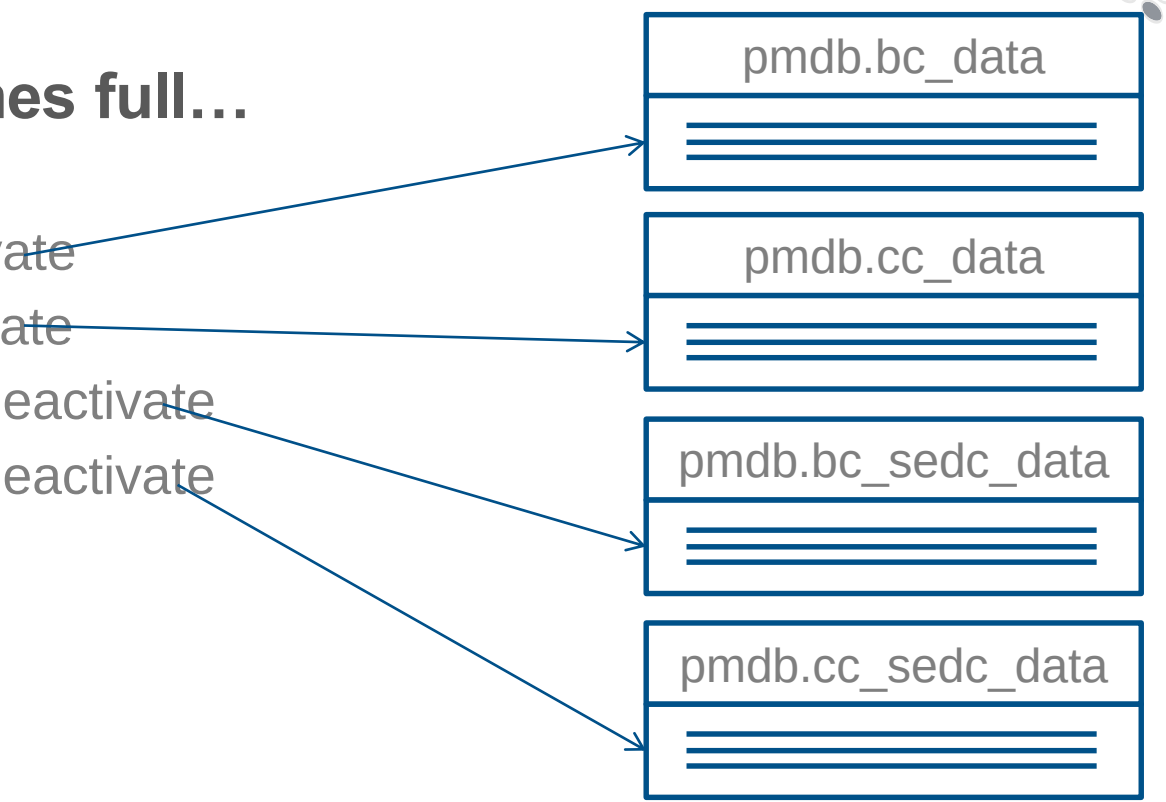
- **Entry point into PMDB**
  - Provide site customization
  - Can do “anything”
    - Default script (xtpmdbhook.sh) creates indexes & cleans up jobs table
- **“Event” driven**
  - Triggered by partition creation
  - Events for CC & BC data tables

Database Hooks (2 of 5)

- **Calling convention**
  - Command line utility which accepts two arguments
    - Event type
    - Partition name
    - '/path/to/hook-script' <event_name> <partition_name>
  - Output saved to power management log
    - /var/opt/cray/log/power_management-YYYYMMDD
- **Execution limits**
  - Bounded runtime
    - Default max time → 600 seconds
    - Configurable with 'xtpmdbconfig'
  - No more than 10 concurrent invocations
    - Prevents "fork bombs" if scripts get stuck

Database Hooks (3 of 5) Event types

- **Partition becomes full…**
  - bc_data_deactivate
  - cc_data_deactivate
  - cc_sedc_data_deactivate
  - cc_sedc_data_deactivate



Database Hooks (4 of 5)

- **Trigger frequency**
  - Depends on data collection configuration
- **Example system**
  - 4 cabinets
    - 48 blades per cabinet
  - pmdb.bc_data ~= 10 data samples per second per blade
    - System wide ~=1920 samples/sec
    - Partition depth → 1,000,000 rows
  - Calculate time to log 1M samples
    - 1,000,000 / 1920 → 521 sec → 8.6 minutes

Database Hooks (5 of 5) xtpmdbhook.sh

- **Default script on installation**
- **It's more than an example**
  - Cleans up "jobs" tables when blade data ages out
    - Prevents unbounded growth
  - Creates time indexes when partitions fill
    - "pmdb.bc_data"
    - "pmdb.bc_sedc_data"
  - Custom scripts must include this functionality
- **Contains example archiver function**
  - Saves gzipped binary table dump for 1 week

If you replace our default hook you need to deal with the growth of the job_* tables

PMDB: psql pmdb pmdbuser …

```
crayadm@smw:~> psql pmdb pmdbuser
psql (9.1.12)
Type "help" for help.

pmdb=> SELECT * FROM pmdb.cc_data limit 4;
              ts                 |  source   | id |   value
---------------------------------+-----------+----+------------
 2015-04-12 07:46:02.18304-05    | 201326592 |  3 |         252
 2015-04-12 07:46:02.18304-05    | 201326592 |  8 |        5460
 2015-04-12 07:46:02.342361-05   | 205520896 |  0 |       12944
 2015-04-12 07:46:02.342361-05   | 205520896 |  1 | 8121423755
```

PMDB: pmdb.cc_data

```
pmdb=> SELECT ts,source2cname(source) as source,id,value
pmdb-> FROM pmdb.cc_data limit 10;
             ts                    | source | id |   value
-----------------------------------+--------+----+------------
 2015-04-12 07:46:02.18304-05      | c0-0   |  3 |         252
 2015-04-12 07:46:02.18304-05      | c0-0   |  8 |        5460
 2015-0
 2015-0
 2015-04-12 07:46:02.342361-05     | c4-0   |  2 |       51921
 2015-04-12 07:46:02.342361-05     | c4-0   |  3 |         249
 2015-04-12 07:46:02.342361-05     | c4-0   |  8 |        5460
 2015-04-12 07:46:02.388346-05     | c3-0   |  0 |       12582
 2015-04-12 07:46:02.388346-05     | c3-0   |  1 | 6655286020
 2015-04-12 07:46:02.388346-05     | c3-0   |  2 |       51907
```

Cray extension converts the 32bit binary 'source' into a cname string

PMDB: pmdb.bc_data

```
pmdb=> SELECT ts,source2cname(source) as source,id,value
pmdb-> FROM pmdb.bc_data limit 10;
                ts                 |   source   | id |  value
-----------------------------------+------------+----+----------
 2015-04-14 06:53:03.488398-05 | c0-0c1s3   | 16 |      102
 2015-04-14 06:53:03.488398-05 | c0-0c1s3   | 17 | 43048852
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n0 | 32 |       41
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n0 | 33 | 50018319
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n1 | 40 |       41
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n1 | 41 | 51028737
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n2 | 48 |       41
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n2 | 49 | 51628372
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n3 | 56 |       41
 2015-04-14 06:53:03.488398-05 | c0-0c1s3n3 | 57 | 51447327
```

PMDB: pmdb.job_info & pmdb.job_timing

```
pmdb=> SELECT * FROM pmdb.job_info;
 job_id |  apid   | user_id |     nids
--------+---------+---------+--------------
 339    | 8362460 |   31137 | {764}
 339    | 8362461 |   31137 | {764}
 340    | 8362463 |   31137 | {764,765,766}
 341    | 8362465 |   31137 | {764,765,766}
```

```
pmdb=> SELECT * FROM pmdb.job_timing;
 job_id |  apid   |          start_ts          |          end_ts
--------+---------+----------------------------+----------------------------
 339    | 8362460 | 2015-04-14 09:17:42.850693 | 2015-04-14 09:17:44.919059
 339    | 8362461 | 2015-04-14 09:18:17.636577 | 2015-04-14 09:18:19.524404
 340    | 8362463 | 2015-04-14 09:19:26.458356 | 2015-04-14 09:19:28.689729
 341    | 8362465 | 2015-04-14 09:19:56.234028 | 2015-04-14 09:19:57.266162
```

PMDB: Using pmdb.job_info & pmdb.nodes

```
pmdb=> SELECT * FROM pmdb.job_info;
 job_id |  apid   | user_id |     nids
--------+---------+---------+--------------
 339    | 8362460 |   31137 | {764}
 339    | 8362461 |   31137 | {764}
 340    | 8362463 |   31137 | {764,765,766}
 341    | 8362465 |   31137 | {764,765,766}

pmdb=> SELECT * FROM pmdb.nodes WHERE nid_num in (764,765,766);
   comp_id    | nid_num
--------------+---------
 c3-0c2s15n0 |     764
 c3-0c2s15n1 |     765
 c3-0c2s15n2 |     766
```

PMDB: pmdb.cc_sedc_data

```
pmdb=> SELECT ts,source2cname(source) as source,id,value
pmdb-> FROM pmdb.cc_sedc_data limit 10;
              ts                 | source |  id  | value
---------------------------------+--------+------+-------
 2015-04-11 12:16:48.366747-05 | c2-0   | 1127 |     6
 2015-04-11 12:16:48.366747-05 | c2-0   | 1128 |     0
 2015-04-11 12:16:48.366747-05 | c2-0   | 1129 |   7.5
 2015-04-11 12:16:48.366747-05 | c2-0   | 1130 |   7.5
 2015-04-11 12:16:48.366747-05 | c2-0   | 1131 |   6.5
 2015-04-11 12:16:48.366747-05 | c2-0   | 1132 |   7.7
 2015-04-11 12:16:48.366747-05 | c2-0   | 1133 |   6.5
 2015-04-11 12:16:48.366747-05 | c2-0   | 1134 |   7.3
 2015-04-11 12:16:48.366747-05 | c2-0   | 1135 |   7.7
 2015-04-11 12:16:48.366747-05 | c2-0   | 1136 |   6.2
```

pmdb.cc_sedc_data with pmdb.sedc_scanid_info

```
pmdb=> SELECT ts,source2cname(source) as source,
pmdb-> sensor_name as sensor, value, sensor_units as units
pmdb-> FROM pmdb.cc_sedc_data INNER JOIN pmdb.sedc_scanid_info
pmdb-> ON cc_sedc_data.id=sedc_scanid_info.sensor_id limit 5;
             ts                | source |         sensor          | value | units
-------------------------------+--------+-------------------------+-------+-------
 2015-04-23 10:03:39.766238-05 | c3-0   | CC_T_COMP_CH0_AIR_TEMP0 | 21.49 | degC
 2015-04-23 10:03:39.766238-05 | c3-0   | CC_T_COMP_CH0_AIR_TEMP1 | 21.65 | degC
 2015-04-23 10:03:39.766238-05 | c3-0   | CC_T_COMP_CH0_AIR_TEMP2 | 21.98 | degC
 2015-04-23 10:03:39.766238-05 | c3-0   | CC_T_COMP_CH0_AIR_TEMP3 | 21.32 | degC
 2015-04-23 10:03:39.766238-05 | c3-0   | CC_T_COMP_CH1_AIR_TEMP0 |  23.1 | degC
(5 rows)
```

PMDB: Find Node Power Sensors Reporting < 10 W

```
crayadm@smw:~> psql pmdb pmdbuser
psql (9.1.12)
Type "help" for help.

pmdb=> SELECT source2cname(source) as cname FROM pmdb.bc_data
pmdb-> WHERE id in (16,32,40,48,56)
pmdb-> and value < 10 and ts > now() - interval '1 minute'
pmdb-> and source2cname(source) in
pmdb-> (select name from sm.expand('rt_node', 's0') where state = 7)
pmdb-> group by cname;
   cname
------------
 c5-0c0s2n0
 c3-0c2s0n1
```

Blade-Level Sensors → pmdb.bc_data

Collected by default on 2-socket Xeon Nodes

ID	Name	ID	Name	ID	Name	ID	Name
16	HSS Power	17	HSS Energy	18	HSS Voltage	19	HSS Current
32	Node 0 Power	33	Node 0 Energy	34	Node 0 Voltage	35	Node 0 Current
40	Node 1 Power	41	Node 1 Energy	42	Node 1 Voltage	43	Node 1 Current
48	Node 2 Power	49	Node 2 Energy	50	Node 2 Voltage	51	Node 2 Current
56	Node 3 Power	57	Node 3 Energy	58	Node 3 Voltage	59	Node 3 Current
64	Node 0 Accelerator Power	65	Node 0 Accelerator Energy	66	Node 0 Accelerator Voltage	67	Node 0 Accelerator Current
72	Node 1 Accelerator Power	73	Node 1 Accelerator Energy	74	Node 1 Accelerator Voltage	75	Node 1 Accelerator Current
80	Node 2 Accelerator Power	81	Node 2 Accelerator Energy	82	Node 2 Accelerator Voltage	83	Node 2 Accelerator Current
88	Node 3 Accelerator Power	89	Node 3 Accelerator Energy	90	Node 3 Accelerator Voltage	91	Node 3 Accelerator Current

Blade-Level Sensors → pmdb.bc_data

Optionally collected

ID	Name	ID	Name	ID	Name	ID	Name
16	HSS Power	17	HSS Energy	18	HSS Voltage	19	HSS Current
32	Node 0 Power	33	Node 0 Energy	34	Node 0 Voltage	35	Node 0 Current
40	Node 1 Power	41	Node 1 Energy	42	Node 1 Voltage	43	Node 1 Current
48	Node 2 Power	49	Node 2 Energy	50	Node 2 Voltage	51	Node 2 Current
56	Node 3 Power	57	Node 3 Energy	58	Node 3 Voltage	59	Node 3 Current
64	Node 0 Accelerator Power	65	Node 0 Accelerator Energy	66	Node 0 Accelerator Voltage	67	Node 0 Accelerator Current
72	Node 1 Accelerator Power	73	Node 1 Accelerator Energy	74	Node 1 Accelerator Voltage	75	Node 1 Accelerator Current
80	Node 2 Accelerator Power	81	Node 2 Accelerator Energy	82	Node 2 Accelerator Voltage	83	Node 2 Accelerator Current
88	Node 3 Accelerator Power	89	Node 3 Accelerator Energy	90	Node 3 Accelerator Voltage	91	Node 3 Accelerator Current

Collected by default on GPU/MIC (accelerated) Nodes

PMDB: pmdb.sensor_info

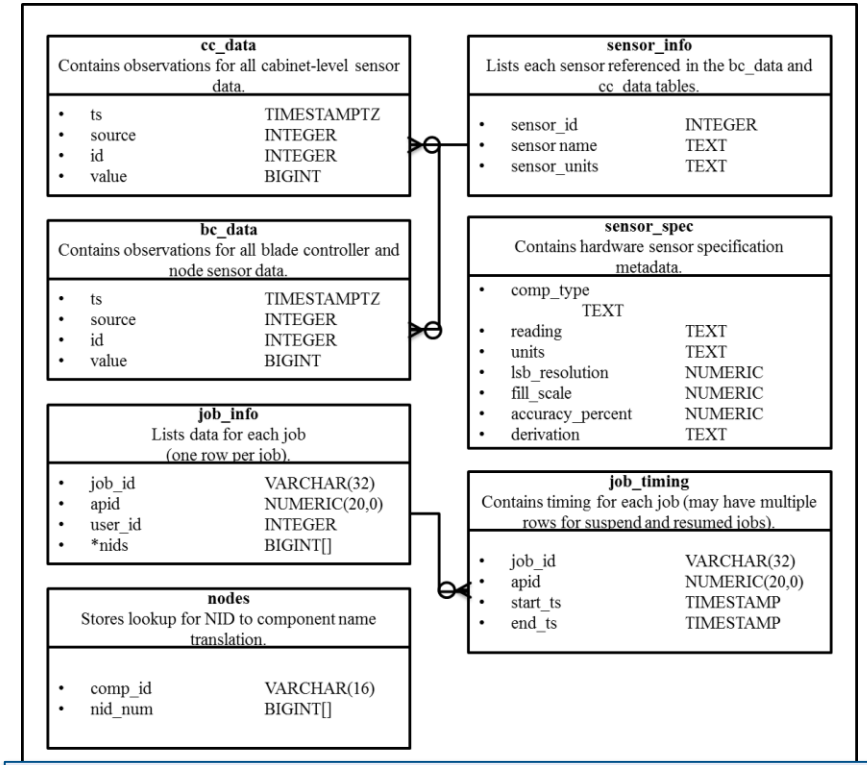
```
pmdb=> SELECT * FROM pmdb.sensor_info ;
 sensor_id |      sensor_name      | sensor_units
-----------+-----------------------+--------------
         0 | Cabinet Power         | W
         1 | Cabinet Energy        | J
         2 | Cabinet Voltage       | mV
         3 | Cabinet Current       | A
         8 | Cabinet Blower Power  | W
        16 | HSS Power             | W
        17 | HSS Energy            | J
        18 | HSS Voltage           | mV
        19 | HSS Current           | mA
        32 | Node 0 Power          | W
        33 | Node 0 Energy         | J
        34 | Node 0 Voltage        | mV
        35 | Node 0 Current        | mA
        40 | Node 1 Power          | W
        41 | Node 1 Energy         | J
        42 | Node 1 Voltage        | mV
        43 | Node 1 Current        | mA
        48 | Node 2 Power          | W
        49 | Node 2 Energy         | J
        50 | Node 2 Voltage        | mV
        51 | Node 2 Current        | mA
        56 | Node 3 Power          | W
        57 | Node 3 Energy         | J
        58 | Node 3 Voltage        | mV
        59 | Node 3 Current        | mA
        64 | Node 0 Accelerator Power   | W
        65 | Node 0 Accelerator Energy  | J
        66 | Node 0 Accelerator Voltage | mV
        67 | Node 0 Accelerator Current | mA
        72 | Node 1 Accelerator Power   | W
        73 | Node 1 Accelerator Energy  | J
        74 | Node 1 Accelerator Voltage | mV
        75 | Node 1 Accelerator Current | mA
        80 | Node 2 Accelerator Power   | W
        81 | Node 2 Accelerator Energy  | J
        82 | Node 2 Accelerator Voltage | mV
        83 | Node 2 Accelerator Current | mA
        88 | Node 3 Accelerator Power   | W
        89 | Node 3 Accelerator Energy  | J
        90 | Node 3 Accelerator Voltage | mV
        91 | Node 3 Accelerator Current | mA
```

OOPS!

PMDB: pmdb.sensor_info

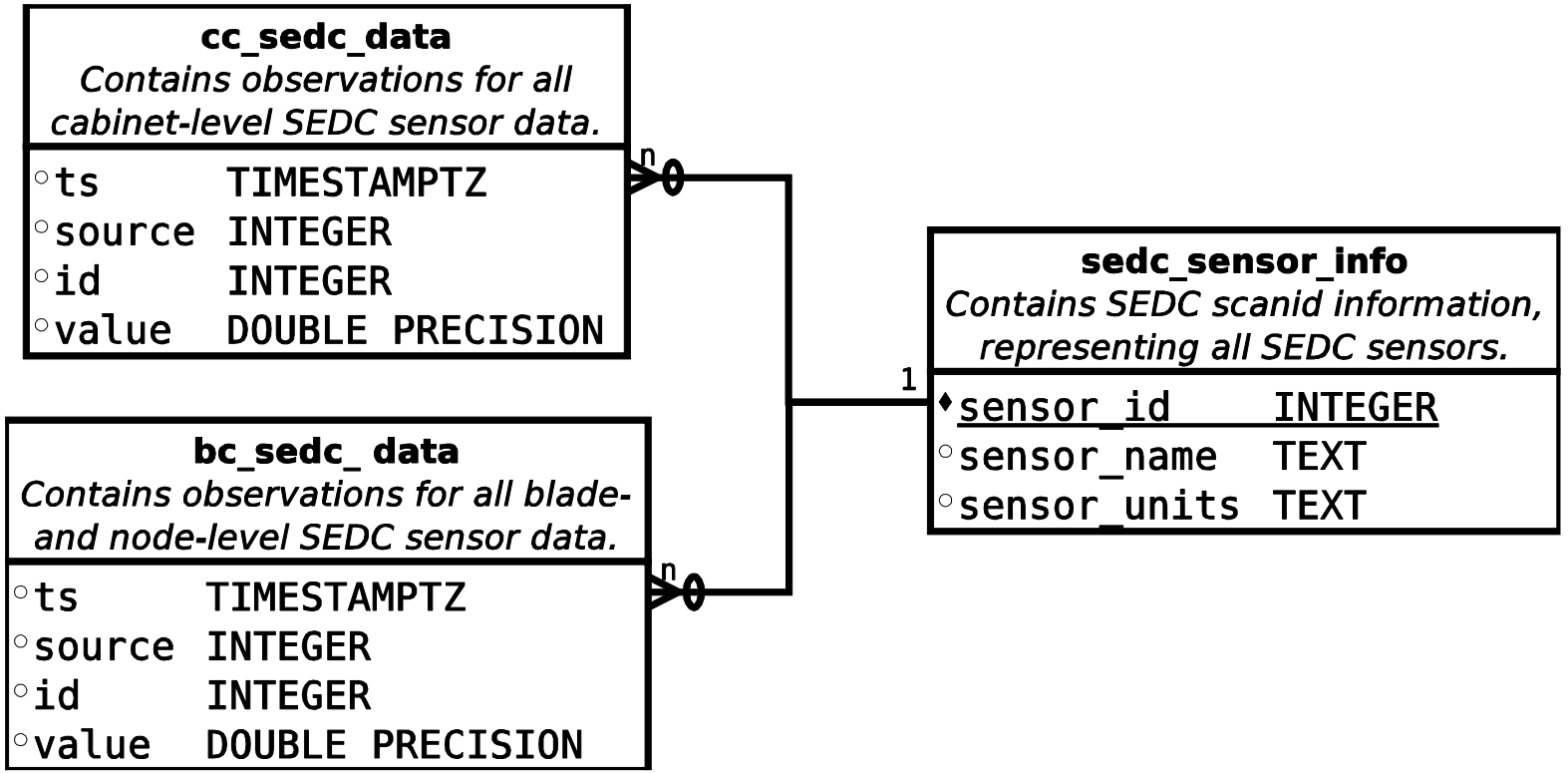
```
pmdb=> SELECT * FROM pmdb.sensor_info ;
 sensor_id |          sensor_name          | sensor_units
-----------+-------------------------------+--------------
         0 | Cabinet Power                 | W
         1 | Cabinet Energy                | J
         2 | Cabinet Voltage               | mV
         3 | Cabinet Current               | A
         8 | Cabinet Blower Power          | W
        16 | HSS Power                     | W
        17 | HSS Energy                    | J
        18 | HSS Voltage                   | mV
        19 | HSS Current                   | mA
        32 | Node 0 Power                  | W
        33 | Node 0 Energy                 | J
```

PMDB Database Schema




```
smw~:/opt/cray/hss/default/etc/xtpmdb.sql
```

PMDB Database Schema, SEDC

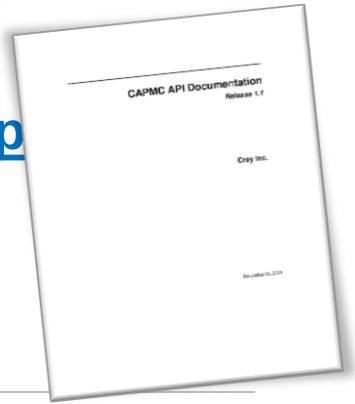


CAPMC

Cray Advanced Platform Monitoring and Control

(Content from 2015 CUG tutorial)

http://docs.cray.com/books/S-2553-11/S-2553-11.p



CAPMC

- **Available on Cray XC Supercomputer Systems**
  - Available with CLE 5.2.UP02 / SMW 7.2.UP02
- **Primary goal is to enable WLM partners**
  - Access to system power/energy data
  - Access to node-, job-, and app-level power/energy data
  - Access to node-, job-, and app-level power capping controls
    - Accelerator (GPU/MIC) power capping on enabled node types
  - Ability to power off (and on) compute nodes

CAPMC Applets: System-Level Monitoring

- **get_system_power [-s start_time] [-w window]**
  - Returns system-level power data
- **get_system_power_details [-s start_time] [-wwindow_length]**
  - Returns cabinet-level data for all cabinets in the system

Time Format: 'yyyy-mm-dd hh:mm:ss'

CAPMC Applets: Node-Level Monitoring

- **get_node_energy_stats [-s start_time] [-e end_time] **
  [--nids nid_list] [--apid apid] [--jobid job_id]
  - Returns statistics for node-level energy (fixed size response)
- **get_node_energy [-s start_time] [-e end_time] **
  [--nids nid_list] [--apid apid] [--jobid job_id]
  - Returns node-level energy data (one record for each node)
- **get_node_energy_counter -t time [--apid apid] [--jobid job_id] **
  [--nids nid_list]
  - Returns raw accumulated energy counter data (one record for each node)
  - Multiple calls needed, raw counters used for delta calculations

CAPMC Applets: Node Power ON | OFF

- **node_on --nids nid_list**
  - Turn-on nodes and boot Linux making them ready to run jobs
- **node_off --nids nid_list**
  - Shutdown Linux and power off the nodes
- **node_rules**
  - Returns information to the WLM w/respect to node on/off operations
  - Allows System admin to establish constraints

nid_list: ‘1,3,9-11, 100-300'

CAPMC Applets: Node Power ON | OFF

- **node_status [--nids nid_list] [--filter 'opt|opt|opt|...']**
  - Returns current status for requested nodes
  - Allows WLM to poll for status of nodes it is power on/off
  - Filters:
    - show_all
    - show_off
    - show_on
    - show_halt
    - show_standby
    - show_ready
    - show_diag
    - show_disabled

CAPMC Applets: Power Capping

- **get_power_cap_capabilities [--nids nid_list]**
  - Returns power capabilities per node-type, for requested nodes
- **get_power_cap [--nids nid_list]**
  - Returns current power cap settings, one record per node
- **set_power_cap --nids nid_list [--node watts] [--accel watts]**
  - Set power cap settings

CAPMC setup on the SMW (1 of 3)

- As root, copy files:

```
cp /var/opt/cray/certificate_authority/certificate_authority.crt \
              /etc/opt/cray/capmc/capmc-ca.crt
cp /var/opt/cray/certificate_authority/client/client.key \
              /etc/opt/cray/capmc/capmc-client.key
cp /var/opt/cray/certificate_authority/client/client.crt \
              /etc/opt/cray/capmc/capmc-client.crt
```

CAPMC setup on the SMW (2 of 3)

- **As root, find the full "hostname" needed for the os_service_url:**

```
cd /var/opt/cray/certificate_authority/hosts
snake-smw:~ # openssl x509 -in host.crt -text | grep "CN=" | grep Subject
        Subject: C=XX, ST=XX, O=XX, OU=XX, CN=snake-smw.us.cray.com
```

- **Create capmc.json file (/etc/opt/cray/capmc/capmc.json)**

```
>snake-smw:/etc/opt/cray/capmc # cat capmc.json
        {
           "os_key":         "/etc/opt/cray/capmc/capmc-client.key",
           "os_cert":        "/etc/opt/cray/capmc/capmc-client.crt",
           "os_cacert":      "/etc/opt/cray/capmc/capmc-ca.crt",
           "os_service_url": "https://snake-smw.us.cray.com:8443"
        }
```

CAPMC setup on the SMW (3 of 3)

- # As root, fix file permissions to look like this:

```
>snake-smw:/etc/opt/cray/capmc # ls -altr
          total 24
          -r--------  1 crayadm crayadm  245 Jul 28  2014 capmc.json
          -r--------  1 crayadm crayadm 1123 Jul 28  2014 capmc-ca.crt
          -r--------  1 crayadm crayadm  887 Jul 28  2014 capmc-client.key
          -r--------  1 crayadm crayadm 3010 Jul 28  2014 capmc-client.crt
          drwxr-xr-x  2 root    root    4096 Apr  3 14:00 .
          dr-xr-xr-x 15 root    root    4096 Apr  8 22:41 ..
```

RUR

Content from 2015 CUG tutorial

Resource Utilization Reporting (RUR)

- **RUR supports a plugin architecture**
  - Many types of data collected using the same infrastructure
- **Several output plugins can be configured**
  - RUR is configured off by default
- **Documented in S-2393:**
  - "Managing System Software for the Cray Linux Environment"

RUR Energy Plugin

- **Collects compute node energy usage data**
- **First introduced in CLE 5.0.UP00**
  - One piece of output data: total energy used across all nodes
  - Output data formatted in JSON list format
- **Updated in follow-on CLE releases**
  - Significant numbers of new energy related data points
  - Output data can be formatted in optional JSON dictionary format
  - Maintains backward support for prior JSON list data format

RUR Energy Plugin: Configuration

- **Configure by editing the file:**
  - /etc/opt/cray/rur/rur.conf
  - It's in the shared root, so use xtopview on the boot node

```
...
[plugins]
   gpustat: true
   taskstats: true
   timestamp: true
   energy: true
   memory: false
...
```

```
...
[energy]
stage: /opt/cray/rur/default/bin/energy_stage.py
post:  /opt/cray/rur/default/bin/energy_post.py
arg:   json-dict
...
```

RUR Energy Plugin: Output, json-list

- **Default output format (until CLE 6.0.UP00).**
- **energy_used:**
  - The total energy (joules) used across all nodes
  - On accelerated nodes, this includes energy used by the accelerators

```
2013-08-30T11:19:06.545114-05:00 c0-0c0s2n2 RUR 18657 p2-
20130829t090349 [RUR@34] uid: 12345, apid: 10963, jobid: 0,
cmdname: /scratch/myuser/myapp/bin64/myapp.ex
plugin: energy ['energy_used', 318]
```

RUR Energy Plugin: Output, json-dict (1 of 3)

- **Broader set of metrics, more easily parsed by Python**

- **energy_used:**
  - Total energy (same as 'energy_used‘ in json-list format)
- **nodes:**
  - Number of nodes in job
- **nodes_power_capped:**
  - Number of nodes with nonzero power cap
- **nodes_throttled:**
  - Number of nodes that experienced throttling
    - (e.g., CPU or memory thermal/power throttling)

RUR Energy Plugin: Output, json-dict (2 of 3)

- **Broader set of metrics, more easily parsed by Python**

- **max_power_cap:**
  - Maximum nonzero power cap
- **max_power_cap_count:**
  - Number of nodes with the maximum nonzero power cap
- **min_power_cap:**
  - Minimum nonzero power cap
- **min_power_cap_count:**
  - Number of nodes with the minimum nonzero power cap

RUR Energy Plugin: Output, json-dict (3 of 3)

- Broader set of metrics, more easily parsed by Python

```
2015-01-16T10:23:10.624977-06:00 c0-0c0s1n1 RUR 13513 \
  p1-20150115t070816[RUR@34] uid: 12795, apid: 81870, jobid: 0, \
  cmdname: /bin/hostname, plugin: energy \
{
  "nodes_throttled": 1, "min_accel_power_cap_count": 0,
  "nodes_with_changed_power_cap": 0, "max_power_cap_count": 0,
  "energy_used": 101, "max_power_cap": 0, "nodes_memory_throttled": 1,
  "accel_energy_used": 0, "max_accel_power_cap_count": 0,
  "nodes_accel_power_capped": 0, "min_power_cap": 0,
  "max_accel_power_cap": 0, "min_power_cap_count": 0,
  "min_accel_power_cap": 0, "nodes_power_capped": 0,
  "nodes": 1, "nodes_cpu_throttled": 1
}
```

RUR Energy Plugin: Output Location

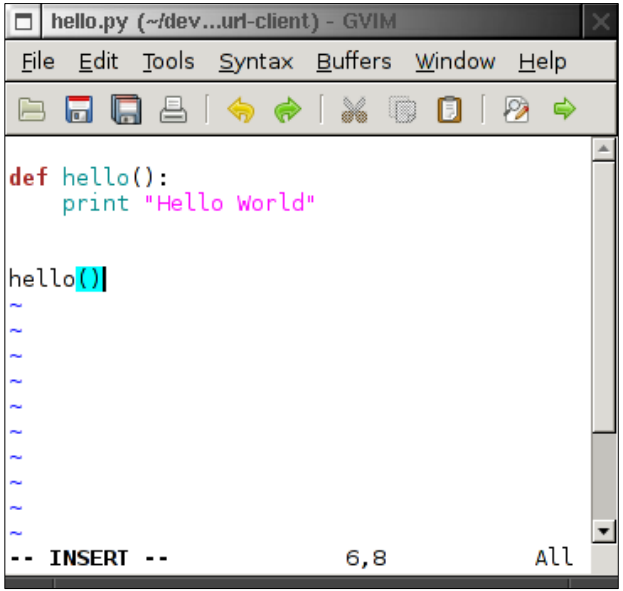
- **Lightweight Log Manager (LLM)**
  - Configured as default output plugin.
  - The ‘llm’ plugin writes to file on the SMW:
    - /var/opt/cray/log/partition-current/messages-date
- **Users can**
  - Opt-in for the ‘user’ plugin
  - Redirect plugin output to a specific file or directory
  - Override the default report type, …
  - See S-2393: section on “Cray-Supplied Output Plugins”

Prototype Application

Content from 2015 CUG tutorial

Objective

- **How can we…**
  - Visualize cabinet power
  - Use CAPMC API interface
  - Run on remote white-box
- **Constraints**
  - No direct HTTPS access to SMW
  - Write few lines of code

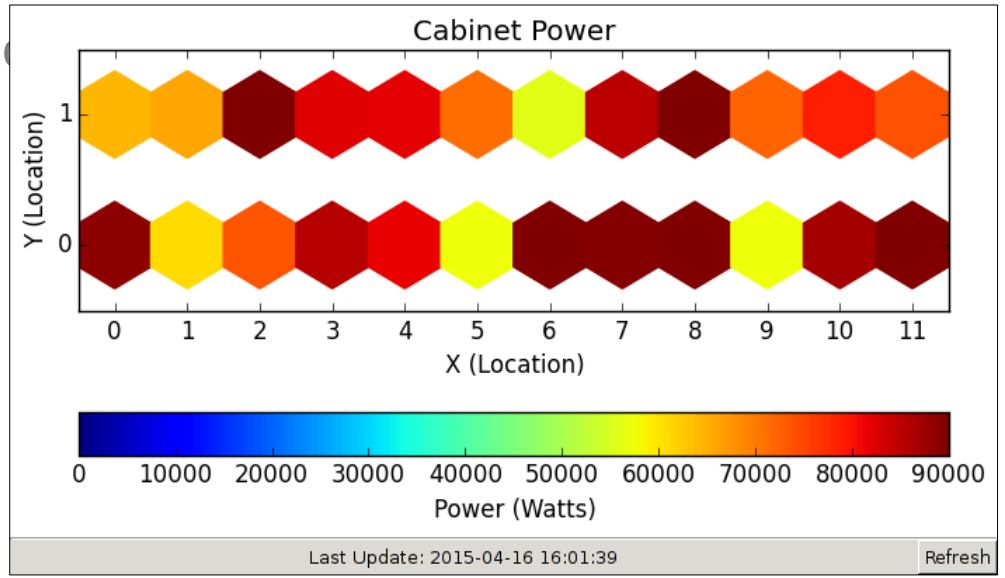


Benefits

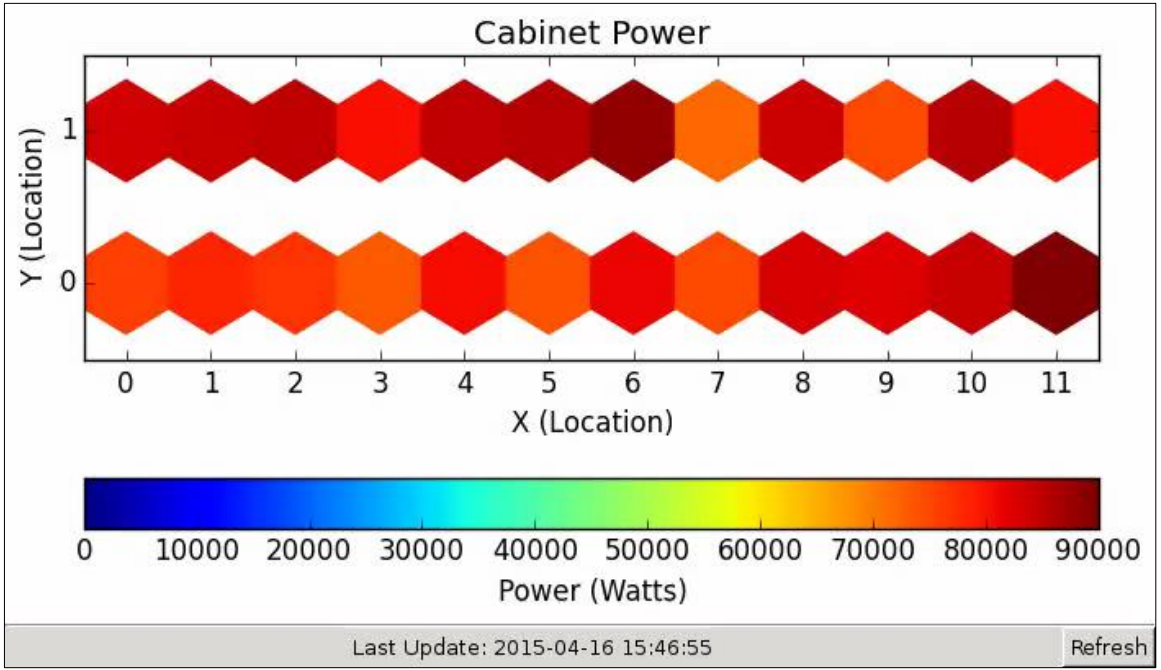
- **Easier data interpretation**
  - “instrument cluster” vs “error log”
- **Visually indicate abnormal behavior**
  - Cabinet emergency power off
    - Power level suddenly drops to zero
  - No applications running
    - Power level near idle state
  - etc…

UI Components

- **Cabinet power data**
  - Each hexagon represents
  - X = column, Y = row
  - (4,1) → cabinet c4-1
  - Approximates floor plan
    - Dependent upon cabling
- **Colorized power scale**
- **Indicate "freshness"**
- **Poll every 10 seconds**



Completed Example



(Video playback 40X real time, 24 cabinet system running HPL)

3rd Party Libraries

- **Software Components**
  - Python – http://www.python.org
    - Programming language
  - PyGTK – http://www.pygtk.org
    - Graphics toolkit
  - Matplotlib – http://www.matplotlib.org
    - Data visualization toolkit
  - PyCURL – http://pycurl.sourceforge.net
    - Multi-platform file transfer library

GTK Framework

- **Instantiate UI**
  - Define Python class
  - Implement ‘boilerplate’

```
class BinMapDemo:

    def replot(self):
        # Add acquisition & plotting code here!
        pass

    def timer_tick(self):
        self.replot()
        return True

    def plot_button_cb(self, widget, data=None):
        self.replot()

    def delete_event(self, widget, event, data=None):
        return False

    def destroy(self, widget, data=None):
        gtk.main_quit()
```

```
    def __init__(self):
        self.window = gtk.Window(gtk.WINDOW_TOPLEVEL)
        self.window.connect("delete_event", self.delete_event)
        self.window.connect("destroy", self.destroy)
        self.graph = gtk.Image()
        self.plot_button = gtk.Button("Refresh")
        self.plot_button.connect("clicked", self.plot_button_cb, None)
        self.ulabel = gtk.Label("Last Update: <none>")

        self.vbox = gtk.VBox()
        self.hbox = gtk.HBox()
        self.vbox.add(self.graph)
        self.vbox.add(gtk.HSeparator())
        self.vbox.add(self.hbox)
        self.hbox.add(self.ulabel)
        self.hbox.pack_end(self.plot_button, False, False, 0)
        self.window.add(self.vbox)
        self.window.show_all()
        self.timer_id = gobject.timeout_add(10000, self.timer_tick)
        self.replot()

    def main(self):
        gtk.main()

if __name__ == '__main__':
    demo = BinMapDemo()
    demo.main()
```

Plot Graph

- **Query CAPMC http API**
  - "wish" a query function that returns tuple of new data
  - "get_cab_power()"
- **Matplotlib "hexbin"**
  - Configure range, labels, axes
  - Save plot to temp file
  - Redraw screen with temp file

```
def replot(self):
    (x, y, m) = get_cab_power()
    plt.clf()
    # vmin and vmax set the colorbar range
    plt.hexbin(x, y, gridsize=(max(x), max(y)),
               C=m, vmin=0, vmax=90000)
    plt.axis([-0.5, max(x) + 0.5, -0.5, max(y) + 0.5])
    plt.title("Cabinet Power")
    cb = plt.colorbar(orientation='horizontal')
    cb.set_label("Power (Watts)")
    cb.set_ticks(range(0, 95000, 10000))
    ax = plt.axes()
    ax.set_aspect(1.8)
    ax.set_yticks(range(0, max(y) + 1))
    ax.set_xticks(range(0, max(x) + 1))
    ax.set_xlabel("X (Location)")
    ax.set_ylabel("Y (Location)")
    plt.savefig('cab-power.png',
                bbox_inches='tight', dpi=100)
    self.graph.set_from_file("cab-power.png")
    now = str(datetime.datetime.today()).split('.')[0]
    self.ulabel.set_text("Last Update: %s" % now)
```

Data Query (PyCURL)

- **CAPMC API Query**
  - Post JSON object to URL
  - Munge result
    - JSON array → tuple of lists
  - Still missing http request function

```
def get_cab_power():
    (x, y, m) = ([], [], [])

    data = capmc_post(
            'https://<example-machine>:8443',
            '/capmc/get_system_power_details',
            json.dumps({}),
            '127.0.0.1:8080')

    if isinstance(data, int):
        return (x, y, m)

    data_obj = json.loads(data)
    if "cabinets" in data_obj:
        for cab in data_obj["cabinets"]:
            x.append(int(cab["x"]))
            y.append(int(cab["y"]))
            m.append(int(cab["avg"]))

    return (x, y, m)
```

HTTP Request (PyCURL)

- **HTTP request function**
  - Post text to URL
  - Tell PyCURL about X.509 certificate files
  - Optional SOCKS5 proxy host
  - Return HTTP error code or result body text
- **Only have SSH access?**
  - Tunnel HTTP via SSH proxy
    - ssh –D 8081 –N user@remote-system
    - socks5h="127.0.0.1:8081"

```
def capmc_post(host, path, post_body, socks5h=None):
    c = pycurl.Curl()
    rx_buffer = StringIO()
    tx_buffer = StringIO(post_body)
    c.setopt(c.URL,      host + path)
    c.setopt(c.CAINFO,   '/path/to/capmc-cacert.pem')
    c.setopt(c.SSLKEY,   '/path/to/capmc-client.key')
    c.setopt(c.SSLCERT, '/path/to/capmc-client.pem')
    c.setopt(c.HTTPHEADER, ['Content-type: application/json'])
    if socks5h != None:
        c.setopt(c.PROXY, 'socks5h://' + socks5h)
    c.setopt(c.POST, 1)
    c.setopt(c.READDATA, tx_buffer)
    c.setopt(c.POSTFIELDSIZE, len(tx_buffer.getvalue()))
    c.setopt(c.WRITEDATA, rx_buffer)
    c.perform()
    sts = c.getinfo(c.RESPONSE_CODE)
    if sts != 200:
        c.close()
        return sts
    body = rx_buffer.getvalue()
    c.close()
    return body
```

Data Query Revisited

- **CAPMC API Query via "capmc" subprocess**
  - Call "capmc" CLI utility
    - No need to write networking code as in previous example
    - CLI is just another program
    - SSH tunneling not supported
  - Munge result
    - JSON array → tuple of lists

```
def get_cab_power():
    (x, y, m) = ([],[],[])

    p = subprocess.Popen(
        ['capmc', 'get_system_power_details'],
        stdout=subprocess.PIPE)
    data = p.stdout.read()
    rc = p.wait()

    if rc != 0:
        return (x, y, m)

    data_obj = json.loads(data)
    if "cabinets" in data_obj:
        for cab in data_obj["cabinets"]:
            x.append(int(cab["x"]))
            y.append(int(cab["y"]))
            m.append(int(cab["avg"]))

    return (x, y, m)
```

Summary

- **Access system telemetry off SMW**
  - Cabinet power query has negligible impact on system operation
- **Utilize software / libraries not available on SMW**
  - Many “useful” Python libraries not shipped on SMW
- **CAPMC built using standards based interfaces**
  - Can use Cray supplied ‘capmc’ CLI client
  - Can use or develop custom 3rd party client

xtpmaction

Content from 2015 CUG tutorial

SMW command line interface for power monitoring and control

xtpmaction Overview

- **Single point of control for PM operations on the SMW**
  - Enforces policy where needed
- **High level functionality**
  - Power capping setup and management
  - Configuration for power/energy monitoring frequency
  - Configure power threshold setting

xtpmaction -a help

```
crayadm@smw:~> xtpmaction -a help
HELP
    Get help on a specific action
Usage:
    xtpmaction -a help [ACTION]
    ACTIONS:
        activate
        active
        create
        deactivate
        delete
        duplicate
        help
        list
        power
        power_overbudget_action
        properties
        propinfo
        pscan
        rename
        reset
        show
        supported
        sysinfo
        system_power_threshold
        update
        validate
        xtinfo
```

Power Capping with xtpmaction (1 of 2)

- **Create a profile**
  - xtpmaction --action create --percent 100 \
    --profile cap_100
- **Customize your profile**
  - xtpmaction -a power --profile cap_100 -i
- **Activate the profile**
  - xtpmaction -a activate --profile cap_100

Power Capping with xtpmaction (2 of 2)

- **List all profiles**
  - xtpmaction -a list
- **Show the active profile**
  - xtpmaction -a show
- **Deactivate the profile**
  - xtpmaction -a deactivate [--profile cap_100]

xtpmaction -a help create

CREATE

Create a power profile based on a percentage of the available host range (max - min) of compute nodes. Default profile name will be '__THRESH%%.pX‘ where the percentage will be specified instead of '%%'. If no percentage is specified, 100 percent will be used. Power caps will be applied to the node control for compute nodes only. Accelerators (if present) will not be power capped by this function, unless power availability constraints are such that a limit on accelerator power use is required.
The profile name to create can be specified with a '--profile' option

ARGS:

-q flag may be used to reduce verbosity
--force flag may be used to overwrite target profile if it exists

Usage:

```
xtpmaction -a create [-q][--force][--partition PARTITION] \
      [--percent PERCENTAGE][--profile PROFILE]
```

Examples:

```
xtpmaction --action create --percent 80 --profile THRESH_PROFILE
xtpmaction --action create --percent 80 --partition p0
```

xtpmaction --action create --percent 100 \ --profile cap_100

```
crayadm@cst72:~/stevem> xtpmaction --action create --percent 100 --profile cap_100

Profile: /opt/cray/hss/default/pm/profiles/p0/cap_100.p0

Descriptor                                  Limits          #Nodes   %node   %host   %accel
#(ComputeANC_HSW_270W_24c_64GB_2133_NoAccel)
compute|01:000d:306f:010e:0018:0040:0855:0000 node=415        566      100     100     0
#(ComputeANC_HSW_270W_24c_64GB_2133_NoAccel)
service|01:000d:306f:010e:0018:0040:0855:0000 node=0          2        0       0       0   (no power cap)
#(Service_SNB_115W_8c_32GB_14900_NoAccel)
service|01:000a:206d:0073:0008:0020:3a34:0000 node=0          4        0       0       0   (no power cap)
crayadm@cst72:~/stevem>
```

xtpmaction -a help power

POWER

Show a total system power estimate. If no profile specified, the current active profile is used, or if no profile is active, a profile will be auto-generated with a node limit of 100 percent. The partition must be specified if multiple partitions exist, unless the partition is part of the specified power profile name. If the percentage argument is prefaced with a +/- sign, then this is understood as a percentage increase/decrease of the current percentage of node limit range. The '--percent_increase' and '--percent_decrease' arguments specify a percentage by which to increase or decrease the current node limit percentage. For example, a power profile with an 80 percent node limit would become a 90 percent node limit if the '--percent_increase 10' argument was provided. The '--powered' argument will restrict the power estimate to nodes that are currently powered on. By default, all nodes, powered or unpowered are included in the power estimate. The '-noff/--num_off' argument specifies number of compute nodes to assume are powered off. The '-i/--interactive' argument specifies that the power estimate should run in interactive mode. When run in interactive mode the user is shown a menu of choices for altering the power estimate, re-displaying the power estimate, or generating a power profile from the estimate.

Arguments:

```
    [-p/--partition PARTITION][--percent_increase PERCENTAGE] [-f/--profile PROFILE]   [--powered]
    [-P/--percent PERCENTAGE] [--percent_decrease PERCENTAGE] [-noff/--num_off NUMBER] [-i/--interactive]
```

Examples:

```
   xtpmaction -a power --profile   fullspeed.p2
   xtpmaction -a power --partition p2 --profile fullspeed
   xtpmaction -a power --partition p2
   xtpmaction -a power --profile   fullspeed -P 80
   xtpmaction -a power --profile   fullspeed -P -20 -noff 5
   xtpmaction -a power --profile   fullspeed -i --powered
   xtpmaction -a power --profile   fullspeed -i -P 80
   xtpmaction -a power --profile   fullspeed --percent_increase 10
   xtpmaction -a power --profile   fullspeed --percent_decrease 10
```

xtpmaction --action power --profile cap_100 -i

```
crayadm@cst72:~/stevem> xtpmaction --action power --profile cap_100 -i

Estimated power use for profile: cap_100.p0

  Sub total:        234890 Num:   566 Pwr:     415 100% Max: 415 (compute|ComputeANC_HSW_270W_24c_64GB_2133_NoAccel)
  Sub total:           830 Num:     2 Pwr:     415 100% Max: 415 (service|ComputeANC_HSW_270W_24c_64GB_2133_NoAccel)
  Sub total:           740 Num:     4 Pwr:     185 100% Max: 185 (service|Service_SNB_115W_8c_32GB_14900_NoAccel)
Profile total:      236460
  Sub total:         14600 Num:   146 Pwr:     100 Static blade power
  Sub total:         18000 Num:     3 Pwr:    6000 Static cabinet power
  Sub total:             0 Num:     1 Pwr:       0 Static system power
Static total:        32600
Combined total:     269060      Current system peak power use:   232352

Choose an option:
         1) percentage
         2) percentage increase
         3) percentage decrease
         4) percentage increase and descriptor to apply increase to
         5) percentage decrease and descriptor to apply decrease to
         6) watts and descriptor to apply setting to
         7) number of nodes assumed powered off
         8) number assumed off and descriptor to apply power off assumption to
         9) use powered nodes only
        10) use powered/unpowered nodes
        11) show power estimate
        12) create power profile
Choice: ('q' to quit) [1-12]: 6
('c' to cancel) [watts,descriptor]: 350,compute|ComputeANC_HSW_270W_24c_64GB_2133_NoAccel
watts,descriptor: 350,compute|ComputeANC_HSW_270W_24c_64GB_2133_NoAccel
```

Demo Live?

xtpmaction --action power --profile cap_100 -i

```
Choose an option:
         1) percentage
         2) percentage increase
         3) percentage decrease
         4) percentage increase and descriptor to apply increase to
         5) percentage decrease and descriptor to apply decrease to
         6) watts and descriptor to apply setting to
         7) number of nodes assumed powered off
         8) number assumed off and descriptor to apply power off assumption to
         9) use powered nodes only
        10) use powered/unpowered nodes
        11) show power estimate
        12) create power profile
Choice: ('q' to quit) [1-12]: 12
Choice: ('c' to cancel) [cap_100.p0]: cap_350w
Created profile: /opt/cray/hss/default/pm/profiles/p0/cap_350w.p0
```

```
Choose an option:
         1) percentage
         2) percentage increase
         3) percentage decrease
         4) percentage increase and descriptor to apply increase to
         5) percentage decrease and descriptor to apply decrease to
         6) watts and descriptor to apply setting to
         7) number of nodes assumed powered off
         8) number assumed off and descriptor to apply power off assumption to
         9) use powered nodes only
        10) use powered/unpowered nodes
        11) show power estimate
        12) create power profile
Choice: ('q' to quit) [1-12]: q
crayadm@cst72:~/stevem>
```

Demo Live?

xtpmaction -a power --profile cap_100

566 compute nodes, each with a cap set at 415 watts

146 Blades, w/static power of 100 watts/blade

3 cabinets, 4000 watts/cab

```
crayadm@cst72:/opt/cray/hss/default/pm> xtpmaction --action power --profile cap_100

Estimated power use for profile: cap_100.p0

  Sub total:        234890 Num:   566 Pwr:     415 100% Max: 415 (compute|ComputeANC_HSW_270W_24c_[illegible]133_NoAccel)
  Sub total:           830 Num:     2 Pwr:     415 100% Max: 415 (service|ComputeANC_HSW_270W_24c_64GB_2133_NoAccel)
  Sub total:           740 Num:     4 Pwr:     185 100% Max: 185 (service|Service_SNB_115W_8c_32GB_14900_NoAccel)
Profile total:      236460
  Sub total:         14600 Num:   146 Pwr:     100 Static blade power
  Sub total:         12000 Num:     3 Pwr:    4000 Static cabinet power
  Sub total:             0 Num:     1 Pwr:       0 Static system power
Static total:        26600
Combined total:     263060      Current system peak power use:   224668
```

Demo Live?

Worst-case power estimate

xtpmaction -a help validate

VALIDATE

Validate a profile, partition, properties file or all of the above.

If the 'all' argument is specified, the profile and partition options are ignored

Usage:

```
xtpmaction -a validate [--profile PROFILE] \
    [--partition PARTITION][all]
```

Examples:

```
xtpmaction -a validate --profile PROF.p0
xtpmaction -a validate --partition p0
xtpmaction -a validate properties
xtpmaction -a validate all
```

xtpmaction -a help activate

ACTIVATE

Activate a specified profile.

Usage:

```
xtpmaction -a activate [--partition PARTITION] --profile PROFILE
```

Examples:

```
xtpmaction -a activate --profile PROF.p0
xtpmaction -a activate --profile PROF    --partition p0
xtpmaction -a activate --profile PROF.p2 --partition p2
```

xtpmaction -a help deactivate

DEACTIVATE

Deactivate a specified profile for a specified optional partition.
Deactivate the current active profile if no profile is specified
If no partition suffix specified on profile, the command
attempts to add one (i.e. PROF -> PROF.p0)

Usage:

```
xtpmaction -a deactivate [--partition PARTITION][--profile PROFILE]
```

Examples:

```
xtpmaction -a deactivate
xtpmaction -a deactivate -p p0
xtpmaction -a deactivate --profile PROF.p0
xtpmaction -a deactivate --profile PROF --partition p0
```

xtpmaction -a help delete

DELETE

Delete specified profile. If no partition suffix on profile name, the command will attempt to add one. If the active profile is deleted, it will be deactivated

ARGS:

-q flag may be used to reduce verbosity

Usage:

```
xtpmaction -a delete [-q] [--partition PARTITION] --profile PROFILE
```

Examples:

```
xtpmaction -a delete --profile PROF.p0
xtpmaction -a delete --partition p2 --profile PROF
xtpmaction -a delete -q --profile PROF.p0
```

xtpmaction -a help list

LIST

List available profiles.

If no partition specified, list all profiles available in all partitions

Usage:

```
xtpmaction -a list [--partition PARTITION]
```

Examples:

```
xtpmaction -a list
xtpmaction -a list --partition p2
```

xtpmaction -a help pscan

PSCAN

Set power management scanning frequencies Allow setting both a system scan rate and a high frequency rate.
The scan frequency will be cached for subsequent invocations. Valid frequency values are:

#	Integer system scan period in milliseconds ([1000-10000])
#	Integer hf scan period in milliseconds ([200-10000])
on	Period gets system default value
off	Turn off scanning

If no scan frequency value is specified, the previously stored scan value will be used if it exists.
The optional 'show' keyword will display the currently cached scan settings

ARGS:

-n MODULE_LIST (a list of modules on which to apply scan settings)
-q flag may be used to reduce verbosity
-c flag may be used to reset any cached settings to their default values.

Usage:

```
xtpmaction -a pscan [-q] [-c] [--partition PARTITION] [--system-scan SysScanFreq]\
      [--hf-scan HighFreqScanFrequency][-n modulelist][-N modulelist_file] [show]
```

Examples:

```
xtpmaction -a pscan -q --system-scan on --hf-scan off
xtpmaction -a pscan -q --hf-scan 333 -n 'c0-0c0s0,c0-0c0s1'
xtpmaction -a pscan -q --hf-scan 333 -n 'c0-0c0s0,c0-0c0s1' --system-scan off
xtpmaction -a pscan -q --partition p0 --hf-scan on -n 'c0-0c0s0,c0-0c0s1'
xtpmaction -a pscan --partition p0 --hf-scan 333 -N /tmp/MODULELIST_FILE
xtpmaction -a pscan --partition p0 show
```

Using xtpmaction -a pscan

Show current settings

```
crayadm@cst72:~/stevem> xtpmaction -a pscan show
Partition:                  p0
Accel Sensors:              0x030303030303030300030000
Non-Accel Sensors:          0x303030300030000
Queue Time:                 5
System Scan Period:         1000ms
High Freq Scan Period:      200ms
High Frequency Module List:     []
crayadm@cst72:~/stevem>
```

Using xtpmaction -a pscan

Enable high frequency scanning in one blade

```
crayadm@cst72:~/stevem> xtpmaction -a pscan --hf-scan on -n c0-0c1s4
Checking cached system scan settings...
CMD: cat /tmp/tmpHgCasF|xtpscan --start \
        --sensor=0x303030300030000 --queue-time=5 --period=1000 -N -
CMD: cat /tmp/tmpy
        --sensor=0
```

```
crayadm@cst72:~/stevem> xtpmaction -a pscan show
Partition:                p0
Accel Sensors:            0x03030303030303030030000
Non-Accel Sensors:        0x303030300030000
Queue Time:               5
System Scan Period:       1000ms
High Freq Scan Period:    200ms
High Frequency Module List:     ['c0-0c1s4']
```

xtpmaction -a help power_overbudget_action

POWER OVERBUDGET ACTION:
Get or set the node power overbudget action on all blades.
'get' will display the current overbudget action. 'set' will update the
Supported overbudget actions are:
log (log the event) (this is the default
nmi (halts the node and drops
power_off (powers off the

Usage:

```
tion ['get'|'set' ACTION]
verbudget_action
power_overbudget_action get
-a power_overbudget_action set log
xtpmaction -a power_overbudget_action set nmi
xtpmaction -a power_overbudget_action set power_off
```

CAUTION: Be aware that applying either of the non-default actions above will bring down nodes and cause applications to fail (If an over-budget condition is detected). We strongly recommend that before changing the default action you review the log messages carefully and consult with Cray Service Personnel for alternative solutions.

Additional Resources

"Monitoring and managing power consumption on the Cray XC30 system"

- **Cray S-0043**
- http://docs.cray.com/books/S-0043-7204/S-0043-7204.pdf

"CLE XC™ System Administration Guide"

- **Cray S-2393**
- http://docs.cray.com/books/S-2393-5204xc/S-2393-5204xc.pdf

"CAPMC API Documentation"

- **Cray S-2553**
- http://docs.cray.com/books/S-2553-10/S-2553-10.pdf

Q&A

Steven J. Martin
stevem@cray.com

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, REVEAL,THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

Backup Slides