XC™ Series Power Management and SEDC Administration Guide

(CLE 6.0.UP07)

S-0043

Contents

1 About the XC™ Series Power Management and SEDC Administration Guide....4
2 Display Power Consumption Information....6
3 Define Frequency of Data Collection....7
4 Manage Power Consumption....8
4.1 Create a Power Profile....8
4.2 View the Contents of a Power Profile....9
4.3 Validate a Power Profile....10
4.4 View System Power Usage Estimates....11
4.5 Use the xtpmaction power Action Interactively....11
4.6 Activate/Deactivate Power Profiles....12
4.7 Modify a Power Profile....12
4.8 Update a Profile....13
4.9 Rename a Profile....13
4.10 Duplicate a Profile....13
4.11 Remove a Profile....13
4.12 Take Action when Power Budget is Exceeded....13
4.13 Dynamic Fan Speed Control....14
4.13.1 Enable Dynamic Fan Speed Control....15
4.13.2 Configure and Validate Dynamic Cooling Control Variables....15
5 View Power Management Settings....18
6 Power Management During System Bringup and Shutdown....19
6.1 Boot CLE with Power Staging....19
6.2 Manage Power Ramp Rates....20
7 The Power Management Database (PMDB)....21
7.1 The PMDB Tables....23
7.2 Query Usage at the Job Level....24
7.2.1 Sample Job-level Query Scripts....25
7.3 Query Usage at the Cabinet Level....26
7.4 SEDC Data Storage on the PMDB....26
7.4.1 Query the PMDB using the xtgetsedcvalues Script....28
7.4.2 Query PMDB for SEDC scanid Information....29
7.4.3 Query PMDB for SEDC CPU Temperature Data....30
7.5 Export Queries to a CSV File....30
7.6 Tune the PMDB....31
7.7 PMDB Tuning Options....32

7.8 Check or Configure PMDB with the pmdb_util Command....33
7.9 Set Disk Storage Parameters....35
7.10 Manual Backup and Recovery of the PMDB....35
8 Export Power Data to a Network Management Station via SNMP....37
9 User Access to P-state Management....39
9.1 Set a P-state in aprun Command....39
9.2 Set a Performance Governor in an `aprun` Command....39
9.3 Use Workload Managers (WLMs) with BASIL....40
10 User Access to Power Management Data....41
11 Measure Per-Job Energy Usage....42
12 Create a Remote Power Management Database....43
12.1 Configure the SMW to Support a Remote PMDB....43
12.2 Configure and Create a Remote PMDB Image....46
12.3 Install and Deploy a Remote Power Management Database....50
12.4 About Upgrades and Patches to the PMDB Software....51
12.4.1 Apply Security Patches to the PMDB....52
12.5 Update the Remote Database Node Software....53
13 Cray Advanced Platform Monitoring and Control Utility for Workload Managers....56
14 Automatic Power Capping....57
15 Troubleshooting....58
15.1 Power Descriptors Missing After a Hardware Change....58
15.2 Invalid Profiles After a Software Change....58
15.3 Invalid Power Caps After Repurposing a Compute Module....58
15.4 Local Properties Settings Missing After Software Update....59

1 About the XC™ Series Power Management and SEDC Administration Guide

The *XC™ Series Power Management and SEDC Administration Guide (CLE 6.0 UP07) S-0043* enables system administrators to manage system heat dissipation, optimize power usage, and use the System Environment Data Collections (SEDC) tool to collect and report system environmental data.

Table 1. Record of Revision

Publication Title	Date	Updates
XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP07) S-0043	July 2018	Supports the SMW 8.0 UP07
XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP06) S-0043	March 2018	Supports the SMW 8.0 UP06
XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP05) S-0043	Oct 2017	Supports the SMW 8.0 UP05
XC™ Series Power Management and SEDC Administration Guide (CLE 6.0.UP04) S-0043	June 2017	Supports the SMW 8.0 UP04
XC™ Series Power Management Administration Guide (CLE 6.0.UP03) S-0043	Feb 2017	Supports the SMW 8.0 UP03
XC™ Series Power Management Administration Guide (CLE 6.0.UP02) S-0043	June 2016	Supports the SMW 8.0 UP02
XC™ Series Power Management Administration Guide (CLE 6.0.UP01) S-0043	March 2016	Supports the SMW 8.0 UP01
Monitoring and Managing Power Consumption on the Cray XC System (SMW 7.2.UP04) S-0043	Sept 2015	Supports SMW 7.2.UP04

Release Information

CLE 6.0 UP07 supports power management for Cavium™ ARM processors in XC series systems.

Typographic Conventions

`Monospace`	Indicates program code, reserved words, library functions, command-line prompts, screen output, file/path names, key strokes (e.g., `Enter` and `Alt-Ctrl-F`), and other software constructs.
`Monospaced Bold`	Indicates commands that must be entered on a command line or in response to an interactive prompt.
`Oblique` or *`Italics`*	Indicates user-supplied values in commands or syntax definitions.
Proportional Bold	Indicates a graphical user interface window or element.

\ (backslash)	At the end of a command line, indicates the Linux® shell line continuation character (lines joined by a backslash are parsed as a single line). Do not type anything after the backslash or the continuation feature will not work correctly.
`smw#`	As a prompt in an example, indicates that the user must be logged in as `root` to perform this task.
`crayadm@smw>`	As a prompt in an example, indicates that the user is logged into the SMW as `crayadm`.
`pmdb=>`	As a prompt in an example, indicates that the user is logged into the Power Management Database (PMDB)

Scope and Audience

This publication is written for Cray® XC™ Series system administrators.

Trademarks

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, Urika-GX, and YARCDATA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, ClusterStor, CRAYDOC, CRAYPAT, CRAYPORT, DATAWARP, ECOPHLEX, LIBSCI, NODEKARE. 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.

2 Display Power Consumption Information

The SMW `xtpget` command displays the current system power usage, average, and peak power over a defined sample period:

```
smw:~> xtpget --config input_file
```

The *`input_file`* specifies the size of a sampling period, in seconds, for peak and average power calculations, the delay time between readings, and the number of readings to display. These arguments can also be specified on the command line.

```
smw:~> xtpget -w 30 -d 60 -c 4
```

The command takes 4 readings, 60 seconds apart, with a calculation window of 30 seconds each, and returns output similar to the following:

```
MESSAGE:xtpget - 2013-04-28 18:38:40 - Current Power 25170.00 (W) Average Power 25206.73 (W)
Peak Power 25425.00 (W)
MESSAGE:xtpget - 2013-04-28 18:39:40 - Current Power 25179.00 (W) Average Power 25175.57 (W)
Peak Power 25378.00 (W)
MESSAGE:xtpget - 2013-04-28 18:40:40 - Current Power 25420.00 (W) Average Power 25152.10 (W)
Peak Power 25420.00 (W)
MESSAGE:xtpget - 2013-04-28 18:41:40 - Current Power 25259.00 (W) Average Power 25143.00 (W)
Peak Power 25259.00 (W)
```

3 Define Frequency of Data Collection

The `xtpmaction pscan` action supports setting both a system scan period and a high frequency period for a subset of modules. The valid range for *system_scan_period* is `1000 - 10000` milliseconds. The default, when on is specified, is `1000` milliseconds.

The high frequency scan provides a greater degree of granularity than the system scan. The valid range for *hf_scan_period* is 200 - 10000 milliseconds. The default is 200 milliseconds. For high frequency scanning use either the *module_list* option or the *module_list_file* option to specify a subset of modules by their valid `cnames`. Modules can be cabinets, chassis, or blades, but there can be no more than 96 blades total. When no scan period values are specified, the action uses the system defaults or, if they exist, cached scan values.

This example initiates both a system scan period of 2000 milliseconds and a high-frequency scan period of 200 milliseconds on the `c0-0c0s3` blade:

```
crayadm@smw> xtpmaction -a pscan --partition p1 --system-scan 2000 hf-scan 200 -n c0-0c0s3
```

This example initiates a high-frequency scan period of 333 milliseconds on all of the modules listed in a file:

```
crayadm@smw> xtpmaction -a pscan -q --partition p1 hf-scan 333 -N /tmp/module_list_file
```

The `-q` option reduces verbosity in the output displayed.

It should be noted that scanning rates are theoretical and the actual collection rates can vary between compute and service blades. Thus, for a given scanning rate over a fixed time interval, there may be missing entries in the PMDB.

> **TIP:** With the default scan settings, data may not be captured for an application that runs to completion in less than one second. This is a limitation of the minimum system scan period of 1 second. In this case Cray recommends identifying the blades that house the nodes that are running the application (up to 96 blades) and adding those blades to the high frequency module list. Then, turn on high-frequency scanning, specifying a scan period of 200 milliseconds, before running the an application. This will increase the amount of data stored in the PMDB for that job.

The optional `show` keyword displays the currently cached scan settings as shown in this example:

```
crayadm@smw> xtpmaction -a pscan --partition p1 show
Partition:                p1
Accel Sensors:            0x030303030303030300030000
Non-Accel Sensors:        0x303030300030000
Queue Time:               5
System Scan Period:       1000ms
High Freq Scan Period:    off
High Frequency Module List:     ['c0-0c0s6']
```

4 Manage Power Consumption

Power management on XC series systems running CLE 6.0 UP07 is initiated by creating one or more power profiles that establish limits on power consumption for a given set of components (or for the entire system), also known as power capping.

Power profiles are created on a per-partition basis, and each partition can have multiple profiles, although only one profile can be active in a partition at one time.

Creating multiple power profiles allows administrators to adapt system power consumption to specific conditions, such as the types of jobs that are running, and utility rates that vary according to time of day or demand.

4.1 Create a Power Profile

Power profiles are defined by selecting a percentage of the available power range for all node types within a partition. The power range is the difference between the maximum amount of power that each type of node can possibly consume and the minimum amount of power that is required to operate the node. For example, if the power range for a node is between a high of 250 Watts (W), and a low of 150W, the available range for power capping is 100W. A power profile created at 50% of the available range defines a limit of 50W, plus the minimum required wattage for this type of node (150W), which in this case equals a limit of 200W. Because a system or a partition contains both compute and service nodes and possibly multiple types of each of these, when a power profile is created at a chosen percentage this percentage is applied based on node types.

To create a power profile use the following command:

```
crayadm@smw> xtpmaction -a create --partition partition -P percentage
```

The following example creates a power profile that defines a 70% power cap on the compute nodes in a single-partition system.

```
crayadm@smw> xtpmaction -a create --profile nightlimit.p0 --percent 70

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

Descriptor                                    Limits           #Nodes  %node
%host  %accel
#(ComputeANC_SNB_115W_8c_32GB_14900_KeplerK20XAccel)
compute|01:000d:206d:0073:0008:0020:3a34:8100 node=380,accel=0  8       70
39     0 (accel uncapped)
#(ComputeANC_IVB_115W_10c_32GB_12800_KeplerK20XAccel)
compute|01:000d:306e:0073:000a:0020:3200:8100 node=380,accel=0  4       70
39     0 (accel uncapped)
#(Service_SNB_115W_8c_32GB_12800_NoAccel)
service|01:000a:206d:0073:0008:0020:3200:0000 node=0            6        0
0     0 (no power cap)
```

In this example the power profile sets a limit on the node control of 380W (70% of the available range) and does not set a limit on the accelerators. Because the accelerator is not capped the host (CPU plus memory) is limited to 39% of the available power range for the host portion of the node.

> **IMPORTANT:** The `create` action sets power limits on compute nodes only. Use the `update` action to set power limits on service nodes that are part of the same partition as the compute nodes specified in a profile.

Default Profile The default profile name is `__THRESH`*`percentage.partition`*, where *`percentage`* is an integer in the range of 0-100, and *`partition`* specifies the partition. If no partition is specified the system assumes `p0`, which is valid only for an single-partition system. To specify a name for the profile, use the `--profile` option.

Percentage Option The `-P percentage` option specifies a percentage of the difference between the minimum and maximum thresholds of the compute nodes. A percentage value of 0 specifies the most aggressive power cap possible, limiting the power consumption to the minimum wattage necessary to operate the node. A percentage value of 50 limits the power consumption to the middle of the range between the minimum and maximum thresholds. If the `-P percentage` is not used, the percentage value is set to 100. Be aware that applying a percentage value of 100% can affect power consumption; it is not the same as not applying power capping.

Overwrite Existing Profile Use the `-F` option to overwrite an existing profile with the same name. This option has no effect when a profile of the same name is the currently active profile.

4.2 View the Contents of a Power Profile

Use the `show` action to display a power profile and the current percentage of the available power range for each node type within a partition, as in this example:

```
smw:~> xtpmaction -a show --profile jtest.p0
```

```
Profile: /opt/cray/hss/default/pm/profiles/p0/jtest.p0

Descriptor                                    Limits             #Nodes  %node  %host  %accel
#(ComputeANC_IVB_260W_24c_64GB_14900_NoAccel)
compute|01:000d:306e:0104:0018:0040:3a34:0000 node=350           4       100    100    0
#(ComputeANC_IVB_260W_20c_32GB_12800_NoAccel)
compute|01:000d:306e:0104:0014:0020:3200:0000 node=350           8       100    100    0
#(Service_SNB_115W_8c_32GB_12800_NoAccel)
service|01:000a:206d:0073:0008:0020:3200:0000 node=0             8       0      0      0 (no power cap)
#(ComputeANC_SNB_115W_8c_32GB_12800_KeplerK20XAccel)
compute|01:000d:206d:0073:0008:0020:3200:8100 node=425,accel=0   4       100    89     0 (accel uncapped)
#(ComputeANC_IVB_260W_20c_64GB_12800_NoAccel)
compute|01:000d:306e:0104:0014:0040:3200:0000 node=350           4       100    100    0
#(ComputeANC_SNB_260W_16c_64GB_12800_NoAccel)
compute|01:000d:206d:0104:0010:0040:3200:0000 node=350           4       100    100    0
```

Alternatively, view a power profile directly by using the `cat` command as in this example.

```
smw:~> cat /opt/cray/hss/default/pm/profiles/p0/nightlimit.p0
```

```
# NOTE: This file should not be edited.
# Any changes to the file must be immediately applied
# to the relevant partition (see xtpmaction -a activate)
#
#(ComputeANC_SNB_115W_8c_32GB_14900_KeplerK20XAccel) supply: 425, host: 95:185, accel: 180:250, node: 275:435
compute|01:000d:206d:0073:0008:0020:3a34:8100,node=380,accel=0
#(ComputeANC_IVB_115W_10c_32GB_12800_KeplerK20XAccel) supply: 425, host: 95:185, accel: 180:250, node: 275:435
compute|01:000d:306e:0073:000a:0020:3200:8100,node=380,accel=0
#(Service_SNB_115W_8c_32GB_12800_NoAccel) supply: 425, host: 95:185, node: 95:185
service|01:000a:206d:0073:0008:0020:3200:0000,node=0
```

Note that a comment precedes each node type, listing the value for supply (the maximum amount of power available for the type of node), and the `min:max` limits for the host (CPU plus memory), the `accel` control and the `node` control. The minimum limit for the `node` control is equal to the host minimum plus the `accel` minimum. The maximum limit is equal to the host maximum plus the accel maximum.

The comment also shows the human-readable form of a node type descriptor. A node type descriptor consists of 8 hexadecimal fields, each of which provide information regarding the characteristics of the type of node. The human-readable form is a direct translation of these hexadecimal values:

`01:000d:206d:0073:0008:0020:3a34:8100`

This is rendered into human-readable form as:

`ComputeANC_SNB_115W_8c_32GB_14900_KeplerK20XAccel`.

4.3 Validate a Power Profile

After creating a power profile, validate it, especially if it will not be activated it immediately. Note that automatic internal validation is part of the `activate` action. Validation provides verification that:

- Each power descriptor has both a service and a compute copy.
- Each power descriptor is present in the default properties file.
- Each node in a partition has a power descriptor that represents that node in the profile.
- The limits defined for each power descriptor do not fall below or above the limits for each control defined for the power descriptor in the properties file.
- No controls defined for the power descriptor are mutually exclusive.
- No controls are defined for the power descriptor that are not defined for that same power descriptor in the properties file.

To validate a specific profile:

```
smw:> xtpmaction -a validate -f profile_name
```

To validate all of the profiles on the system:

```
smw:> xtpmaction -a validate all
```

Typically, if validation fails it is because the hardware on the system has changed or a node was repurposed, e.g., a service node was repurposed as a compute node. This can happen even if a blade is removed and replaced without changing anything. You may see an error message similar to this:

```
ERROR: descriptor service|01:000a:206d:0073:0008:0020:3200:0000 does not exist in
properties file
```

If this is the case, run the `xtdiscover` command to capture any changes that were made to the HSS database.

```
crayadm@smw:> su - root
smw:# xtdiscover
```

After running `xtdiscover`, revalidate to verify that the profiles are still appropriate, and recreate or update any profiles that fail validation.

In the absence of an error message it is not necessary to run `xtdiscover` but it will still be necessary to recreate the profile(s) that failed validation.

4.4 View System Power Usage Estimates

After creating a power profile, it can be useful to see an estimate of what the power usage will be when the profile is active. The `xtpmaction power` action provides an estimate of the total system power under the specified profile. If a profile is not specified, the command uses the currently active profile. If there is no active profile, a profile is generated automatically, with a node limit of 100 percent.

On a system with multiple partitions, specify the partition explicitly with the `--partition` option, or as the extension to the power profile name, for example `.p3`.

The default behavior of the `power` action is to base the estimate on all nodes, including those that are powered off. Use the `--powered` option to specify that the estimate be based only on the nodes that are powered on. Use the `--num_off` option to specify the number of nodes that should be assumed to be powered off. Be aware that these two options are mutually exclusive.

Use the `--percent_increase` and `--percent_decrease` options to specify a percentage by which to increase or decrease the current node limits. For example, if the profile sets the node limit to 60% of the available range, using `--percent_increase 20` will show a power usage estimate based on 80% of the available range.

The following example displays the projected power usage on a non-partitioned system (`p0`) for the profile `jtest.p0`:

```
smw:~> xtpmaction -a power --profile jtest.p0

Estimated power use for profile: jtest.p0

Sub total:   1640 Num:  4 Pwr:  410 100% Max: 410 (compute|ComputeANC_IVB_115W_10c_32GB_14900_KeplerK40SAccel)
Sub total:   1700 Num:  4 Pwr:  425 100% Max: 425 (compute|ComputeANC_SNB_115W_8c_16GB_10600_IntelKNCAccel)
Sub total:   1480 Num:  8 Pwr:  185 100% Max: 185 (service|Service_SNB_115W_8c_32GB_14900_NoAccel)
Sub total:   1700 Num:  4 Pwr:  425 100% Max: 425 (compute|ComputeANC_IVB_115W_12c_32GB_14900_IntelKNCAccel)
Sub total:   1400 Num:  4 Pwr:  350 100% Max: 350 (compute|ComputeANC_IVB_260W_24c_64GB_14900_NoAccel)
Sub total:   1440 Num:  4 Pwr:  360 100% Max: 360 (compute|ComputeANC_HSW_240W_28c_128GB_2133_NoAccel)
Sub total:   2800 Num:  8 Pwr:  350 100% Max: 350 (compute|ComputeANC_IVB_260W_20c_32GB_12800_NoAccel)
Sub total:   1640 Num:  4 Pwr:  410 100% Max: 410 (compute|ComputeANC_SNB_115W_8c_32GB_14900_KeplerK20XAccel)
Sub total:   1400 Num:  4 Pwr:  350 100% Max: 350 (compute|ComputeANC_IVB_260W_20c_64GB_12800_NoAccel)
Sub total:   1400 Num:  4 Pwr:  350 100% Max: 350 (compute|ComputeANC_SNB_260W_16c_64GB_12800_NoAccel)
Profile total:  16600
Sub total:     1400 Num: 14 Pwr:  100 Static blade power
Sub total:     3200 Num:  1 Pwr: 3200 Static cabinet power
Sub total:        0 Num:  1 Pwr:    0 Static system power
Static total:    4600
Combined total: 21200      Current system peak power use:     5509
```

If the results show that the power profile will not be effective in limiting power consumption to the desired level, recreate the profile with new values or use the `update` action to fine-tune the profile for individual node types.

Alternatively, use the `interactive` option to test a number of changes to a profile, and then create a new profile based on those changes.

4.5 Use the `xtpmaction power` Action Interactively

The `--interactive` (or `-i`) option for the `power` action brings up a menu of choices, which include all of the options available to the power action from the command line. In addition, there is an option to specify absolute power levels in Watts, rather than as a percentage of the current threshold, and an option to create a new profile based on the changes made while in interactive mode.

In the following example, a profile was not specified, so the command generates a profile with a default threshold of 100% for an unpartitioned system, and displays an estimated power usage for the entire system. In addition, the output presents a menu of choices:

```
smw:~> xtpmaction -a power -i
Estimated power use for profile: __THRESH100.p0
  Sub total:      850 Num:  2 Pwr:  425 100% Max: 425 (compute|01:000d:206d:0073:0008:0020:3a34:8100)
  Sub total:     1700 Num:  4 Pwr:  425 100% Max: 425 (compute|01:000d:306e:0082:000a:0020:3a34:8300)
  Sub total:     2550 Num:  6 Pwr:  425 100% Max: 425 (service|01:000a:206d:0073:0008:0020:3200:0000)
  Sub total:     1700 Num:  4 Pwr:  425 100% Max: 425 (compute|01:000d:306e:0073:000a:0020:3a34:8300)
Profile total:   6800
  Sub total:      600 Num:  6 Pwr:  100 Static blade power
  Sub total:     3200 Num:  1 Pwr: 3200 Static cabinet power
  Sub total:        0 Num:  1 Pwr:    0 Static system power
Static total:    3800
Combined total: 10600      Current system peak power use:     4928
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]:
```

Select an option to be prompted for an appropriate response. For example, choose option 4 to receive the following prompt:

```
[percent,descriptor]:
```

Each subsequent choice is additive, unless the choice is incompatible with a previous choice. For example, choosing option 1, then option 7 results in the display of a power estimate at a specified limit percentage with a specified number of compute nodes assumed to be off. If the next choice is option 4, this will replace the percentage limit (set previously with option 1) with a new percentage limit for the specified node type.

When satisfied with the new estimated limits, choose option 12 to create and save a power profile file.

If you replace a currently active power profile, the modified profile is sent immediately to the associated components.

4.6 Activate/Deactivate Power Profiles

Use the following command to validate and activate a power profile:

```
crayadm@smw> xtpmaction -a activate -f profile_name
```

Use the following command to deactivate the currently active profile:

```
crayadm@smw> xtpmaction -a deactivate
```

> **TIP:** When replacing an active profile with a different one, use only the `activate` action to enable the new profile. It is not necessary to first use the `deactivate` action.

4.7 Modify a Power Profile

The `xtpmaction` command allows you to update, rename, duplicate, and delete power profiles.

4.8 Update a Profile

The `update` action enables you to fine-tune a power profile, by modifying the power limit for each type of node individually. You can also use this action to apply a power cap to service nodes. To change the power limits for an individual descriptor in a power profile use the following command:

```
smw:~> xtpmaction -a update -f profile_name --desc power_descriptor --role \
node_type  -P percentage | --watts wattage --control control_name
```

Specify the profile name, descriptor, role, and the new power limit for that descriptor. The descriptor can be supplied as a hexadecimal value or in human-readable form. If the descriptor has more than one control, then specify the control name. Use the `show` action described in *View Power Management Settings* on page 18 to see the current descriptor information in the profile.

Specify the new power limit as a percentage of the available range, using the `-P` option, or as a specific wattage value within the range, using the `--watts` option.

> **NOTE:** Depending on node power constraints, it may not be possible to comply with the requested power limit on nodes with accelerators without adjusting the current `accel` or `node` limits.
>
> The role node type is either `compute` or `service`, and can be abbreviated as `c` or `s`.

4.9 Rename a Profile

To rename an existing profile:

```
smw:~> xtpmaction -a rename --profile profile_name new_profile_name
```

4.10 Duplicate a Profile

To create a duplicate of an existing profile with a new name:

```
smw:~> xtpmaction -a duplicate -f profile_name new_profile_name
```

4.11 Remove a Profile

To remove a power profile from the system:

```
smw:~> xtpmaction -a delete -f profile_name
```

4.12 Take Action when Power Budget is Exceeded

Use the following command to specify the action to take when node power consumption exceeds the threshold specified by the power profile:

```
smw:~> xtpmaction -a power_overbudget_action set action
```

The *action* can be one of the following values:

Table 2. Power Over Budget Actions

log	Logs the event. This is the default.
nmi	Halts the node and drops the kernel into debug mode.
power_off	Powers off the node.

CAUTION: Be aware that applying either of the non-default actions above will bring down nodes and cause applications to fail. Cray strongly recommends that system administrators review the log messages carefully and consult with Cray Service Personnel for alternative solutions before changing the default action.

4.13 Dynamic Fan Speed Control

Effective with SMW version 8.0.UP04, the HSS cooling system for liquid-cooled XC Series cabinets supports dynamic fan speed control by row or for the entire system.

When dynamic fan speed control is not enabled the HSS cooling software operates the cabinet fans at one of 3 fan speeds, defined as `fan_speed_idle` when the blades in the cabinet are not powered on, `fan_speed_high` when a CPU or GPU is within 8 degrees of the highest temperature that it can operate at without being throttled (TJMAX), and `fan_speed_normal` at all other times.

The speed setting of `fan_speed_normal` ensures that, under normal operation, the temperature of the CPU/GPU dies are maintained below the hot spot detection threshold. If the cooling water is at the required temperature and the temperature setpoint is set appropriately, no hot spot should be detected, as this setting is expected to cover the worst case. Typically, die temperatures on a production system fluctuate but are below the throttle threshold most of the time. Setting fan speed to a constant `fan_speed_normal` is unnecessary and can consume more energy than is needed to properly cool the system.

When the dynamic fan speed feature is enabled, the cabinets self-regulate their fan speed based upon observed CPU and/or GPU temperatures. Each cabinet in a row runs its fans at the same speed, based on the highest CPU or GPU temperature sensor reading from all of the blades in all cabinets within the row. The frequency with which fan speeds change in response to temperature sensor readings varies depending on the type of jobs running on the system, and is bounded by two pre-existing `ini` file variables:

- `fan_speed_step_up_delay` This variable controls how fast the system will switch to a higher speed in a fan speed table if die temperatures are increasing. The default is 20 seconds.
- `fan_speed_step_down_delay` This variable controls how fast the system will switch to a lower fan speed if die temperatures are decreasing.The default is 300 seconds.

> **IMPORTANT:** Cray recommends that these and other cooling variables related to dynamic fan speeds in the initialization files be kept at their default values. The exception is `fan_auto_speed_enable`, which enables dynamic fan speed control.

Enabling dynamic fan speed control does not supercede CPU hot spot detection and control. When a hot spot is detected, the cabinet fans in a row will still switch to the `fan_speed_high` setting and remain at that setting until the hot spot is cleared. Similarly, if the blades are powered down, the fans will run at the `fan_speed_idle` setting.

4.13.1 Enable Dynamic Fan Speed Control

Prerequisites

Dynamic fan speed has not enabled at the system level or on a specified row within the system.

Procedure

1. Edit the system-level (`hss.ini`) file or a row-level (`hss_rN.ini`) file in the `/opt/tftpboot/ccrd` directory to set the `fan_auto_speed_enable` variable to `1`.

   Setting fan speeds dynamically on systems with mixed blower types within the same row is not supported. On systems with both STD and HP blowers in separate rows, fan speed settings must be done via row-specific `ini` files.

   ```
   fan_auto_speed_enable=1
   ```

2. If the system is running, reload the `ini` file or files.

   ```
   crayadm@smw> xtccr load_ini
   ```

3. The cooling software on each blade will automatically generate fan speed tables based on the CPUs and/or GPUs that are on the blade. To view the current fan speed table run the following command on the SMW:

   ```
   crayadm@smw> xtdaemonconfig --daemon ccrd|grep _table
   c0-0c0s7: fan_auto_speed_table_cpu=-:92:2750|91:87:2600|86:82:2450|81:77:2300|76:72:2150|71:-:*2000
   c0-0c0s7: fan_auto_speed_table_gpu=-:80:2750|79:75:2600|74:70:2450|69:65:2300|64:60:2150|59:-:*2000
   ```

   The above fan speed tables were generated for both the CPUs and the GPUs on blade `c0-0c0s7`. Each set of values between the **|** symbol gives the temperature range in degrees C and the corresponding fan speed in RPM. For example, For CPUs, a fan speed of 2750 RPM is specified for component temperatures of 92°C and above, and a fan speed of 2600 RPM is specified for component temperatures between 91°C and 87°C.

4.13.2 Configure and Validate Dynamic Cooling Control Variables

Under normal circumstances, administrators need only set the `fan_auto_speed_enable` to 1 to enable dynamic fan speed control. All other dynamic fan speed related variables should be left at their default settings.

In particular, adjusting the `fan_auto_speeds` variable is not recommended as the automatically generated fan speed tables will always be correct for the type of hardware on each blade.

The following settings are described here for use in special situations where the default values are not adequate.

CAUTION: Cray recommends that these settings (other than `fan_auto_speed_enable`) be changed only in close consultation with Cray service. Refer to the `xtccr(8)` man page for complete list of all `xtccr` configuration attributes.

fan_auto_high_temp_offset

Specifies the offset from the highest temperature that a CPU or GPU can operate at without being throttled (TJMAX), that corresponds to the highest fan speed in a fan speed table. The default value of `fan_auto_high_temp_offset` is 10. The potential range of values for this variable are >= `0` and <= `20`. For example, if `fan_auto_speed_high` is not set and `fan_auto_high_temp_offset` is set if a component has a TJMAX of 100, then the highest fan speed in the fan speed table will be equal to `fan_speed_normal`, and the corresponding temperature for that fan speed will be at >= 90.

```
fan_auto_high_temp_offset=10
```

fan_auto_speed_enable

Enables automatic fan speed selection. Set this variable to 1 to enable dynamic fan speed support. The default value is 0 (disabled).

fan_auto_speed_high

Specifies the highest fan speed that can be used within a fan speed table, whether the table is user-specified or auto-generated. The default value of `fan_auto_speed_high` in auto-generated fan speed tables is the value of `fan_speed_normal`. The potential range of values for this variable are >= `fan_speed_normal` and <= `fan_speed_high`.

```
fan_auto_speed_high=3100
```

fan_auto_speed_min

This is the lowest fan speed that will be used in a fan speed table. This value can not be less than 1550 for standard blowers and 1900 for HP blowers. The default value is 1900 for standard blowers and 2400 for high-pressure (HP) blowers.

fan_auto_speed_temp_step

Defines the component temperature in degrees C that will cause a different fan speed to be selected from the fan speed table. Value must be >= 2 or <= 12. The default value is 5. This is used only if `fan_auto_speed_enable=1`.

```
fan_auto_speed_temp_step=5
```

fan_auto_speed_rpm_step

In an auto generated fan speed table, each speed is separated by `fan_auto_speed_rpm_step` from it's neighbor in the table. The highest fan speed in an auto generated table will be the result of `fan_speed_high` - `fan_auto_speed_rpm_step`. For example, if `fan_speed_high` is 3100 and `fan_auto_speed_rpm_step` is 150 then the speeds in the auto generated fan speed table will will be 2950, 2800, 2650, etc.

The value must be >= 100 or <= 300. The default value is 150. This has no effect on fan speeds defined via `fan_auto_speeds`. This is used only if `fan_auto_speed_enable=1`.

```
fan_auto_speed_rpm_step=150
```

fan_auto_speed_high

This is the highest fan speed in an auto-generated fan speed table. The default value is equal to fan_speed_normal (as defined in the .ini file, which is usually 2700 for standard blowers and 3400 for HP blowers.

fan_auto_high_temp_offset

This is the temperature offset (in degrees C), from tjmax to use when creating the entry for the highest fan speed in an auto-generated fan speed table. The default value is 10. Allowed values must be between 0 and 20.

fan_auto_speeds

Defines the contents of the fan speed table. The highest speed allowed is `fan_speed_high` and the lowest speed allowed is `fan_auto_speed_min`. A minimum of 2 and a maximum of 15 fan speeds may be defined. No duplicates are allowed. Auto fan speeds are switched whenever component temperatures vary by `fan_auto_speed_temp_step` degrees C.

This is used only if `fan_auto_speed_enable=1`. If `fan_auto_speed_enable=1` and `fan_auto_speeds` are not defined, then fan speed tables will be auto generated.

Cray does not recommend this configuration because the fan speed table is used for different types of CPUs/GPUs, whereas auto-generated fan speed tables are built using the threshold temperature (TJMAX) for each specific type of CPU/GPU.

fan_speed_step_up_delay

Specifies the amount of time before the system switches to a higher speed in a fan speed table when die temperatures are increasing. The default is 20 seconds.

fan_speed_step_down_delay

Specifies the amount of time before the system switches to a lower fan speed when die temperatures are decreasing.The default is 300 seconds.

INI File Validation

If the dynamic fan speed variables have been changed from their default values, it's important to validate the `.ini` files, prior to loading them onto the controllers. Use the `xtccr --validate` command to do this.

```
crayadm@smw> xtccr --validate=filename
```

Some of the variables defined in the cooling `.ini` files may be fully validated in this fashion, whereas other variables may only be provisionally validated, as information specific to each cabinet is required to fully validate the value of a variable.

Setting fan speeds dynamically via `xtccr` on systems with mixed blower types within the same row is not supported. On systems with both STD and HP blowers in separate rows, fan speed settings must be done by means of row-specific `.ini` files.

For example, the value of `fan_speed_high` can only be validated provisionally because knowledge of the type of fans installed within a cabinet (STD or HP) is required to fully validate the value.

5 View Power Management Settings

The following command displays the active profile on a single-partition system:

```
smw:~> xtpmaction -a active
late-night-profile.p0
```

Use the --partition option to view the active profile for a specific partition.

```

smw:~> xtpmaction -a active --partition p1
__THRESH100.p1
```

The following command displays all of the power profiles available on the system:

```
smw:~> xtpmaction -a list
__THRESH100.p1
__THRESH80.p2
late-night-profile.p1
mid-day-throttle.p1
simple.p2
```

Use the --partition option to view only the available profiles on a specific partition.

The following command displays a list of the properties of the power descriptors for the system:

```
smw:~> xtpmaction -a properties

DESCRIPTOR PROPERTIES:
 compute|01:000a:206d:005f:0006:0020:3200:0000,node=150:300
 service|01:000a:206d:005f:0006:0020:3200:0000,node=150:300
 compute|01:000d:206d:00be:000c:0040:3200:0000,node=150:300
 service|01:000d:206d:00be:000c:0040:3200:0000,node=150:300
 compute|01:000a:206d:0082:0008:0020:3200:0000,node=150:300
```

6 Power Management During System Bringup and Shutdown

Boot time power surges can be problematic when the power available to the site is insufficient to meet, even temporarily, a higher demand.

Rapid bringup of a very large system can cause power draws that affect, not just the site, but other users on that power grid. Similarly, a rapid drop in power consumption can create instability in the grid. Placing such stresses on the grid can eventually result in sites having to pay higher rates for power.

Additionally, sites that transition between multiple large systems may need to provide a period of overlap, where one system ramps down gradually while the other system ramps up.

- For very large systems, administrators can configure the `xtpowerd` daemon to control both powerup and power down situations, as described in *Manage Power Ramp Rates* on page 20
- On smaller systems it can be enough to specify a power threshold when booting the system or by forcing a simple staged bringup, as described in *Boot CLE with Power Staging*

6.1 Boot CLE with Power Staging

A staged system bring-up can prevent boot-time power consumption from exceeding a desired threshold. Prior to booting the system, use one of the following options for the `xtbounce` HSS tool to specify how the nodes will be powered up.

The `-S` option for the `xtbounce` command performs a staged power-up that ensures the system power draw never exceeds a threshold that was defined by the `xtpmaction -a system_power_threshold set` command.

```
crayadm@smw> xtpmaction -a system_power_threshold set threshold_wattage
crayadm@smw> xtbounce -S id-list
```

Alternatively, the `-F` option forces a simple staged power-up. First, 1/2 of the nodes are powered up, then 1/3 of the nodes, and finally the remaining 1/6 of the nodes. Be aware that if both the `-S` and `-F` options are specified, the `-F` overrides the `-S` option.

```
crayadm@smw> xtbounce -F id-list -p partition
```

The *`id-list`* specifies a list of identifiers to be initialized (bounced). Identifiers can be separated by either a comma or a space and can be identifiers for partitions, sections, cabinets (L1s), cages, or blades (L0s). When attempting a warm reset, nodes are allowed. If no identifiers are specified, the default is the specified partition. If no partition is specified, the default is the value of the *`CRMS_PARTITION`* environment variable. If the *`CRMS_PARTITION`* environment variable is not set and there is only one partition active, the default is the active partition. Otherwise, an error message is displayed along with the list of active partitions, and the command will abort. Valid *`partition`* values are of the form `p`*n*, where *n* is an integer in the range of 0-31.

6.2 Manage Power Ramp Rates

About this task

The `xtpowerd` daemon allows sites to control the rate at which nodes are powered on/off or booted over time in order to smooth out large fluctuations in power use. Site managers should be aware that there is a tradeoff between more gradual system power transitions and power efficiency. By design this feature uses power inefficiently. The tradeoff comes in easing stress on site and grid power infrastructure. The `xtnmi` command is not subject to ramp rate limiting.

By default, the power ramp rate limiting feature is turned off because the functionality is of use only for the largest XC Series systems. To enable the feature, edit the `/opt/cray/hss/default/etc/xtpowerd.ini` file:

Procedure

1. Edit the `ramp_limited=` line to change the value to true. This sets the ramp limit to approximately 2 MW/ minute.

2. In the rare instance where it is necessary to change the ramp limit, edit the `#ramp_limit=2000000` to remove the # character and set the ramp rate to a new value. Ensure that there are no spaces at the end of this line. Typically, this would be done only in consultation with the power utility.

3. After saving the `xtpowerd.ini` file, send a `SIGHUP` to the `xtpowerd` daemon to change the configuration and, on high-availability systems, sychronize the configuration on both SMWs.

   IMPORTANT: The `xtpowerd` ramp rate limiter assumes cabinet power operations are performed with the nodes powered off. Powering a cabinet off while the nodes are on will result in a power ramp rate limit violation. Use the `xtcli power down_node command` to power off nodes.

   TIP: For users of the Cray Advanced Platform Monitoring and Control (`capmc`) utility, enabling power ramp rate limiting may cause `capmc node_rules` latency times to be too short. If that is the case, manually increase the following rates:

   ```
   latency_node_off
   latency_node_on
   latency_node_reinit
   ```

7 The Power Management Database (PMDB)

Power consumption data that is collected by scanning the system is stored in the Power Management Database (PMDB). The PMDB can be located on the SMW or on a dedicated node.

System Environment data (SEDC) that is collected during system operation is also stored in the PMDB. SEDC provides environmental data such as voltage, current, power, temperature, humidity, hardware status, and fan speeds from all available sensors on hardware components. Although it is not a component of power management, administrators may find the SEDC data useful in analyzing system power usage when configuring power management settings.

The PMDB schema is defined on the SMW in `/opt/cray/hss/default/etc/xtpmdb.sql`. The following graphics represent the schema tables for power management and SEDC.

Figure 1. Power Management Tables

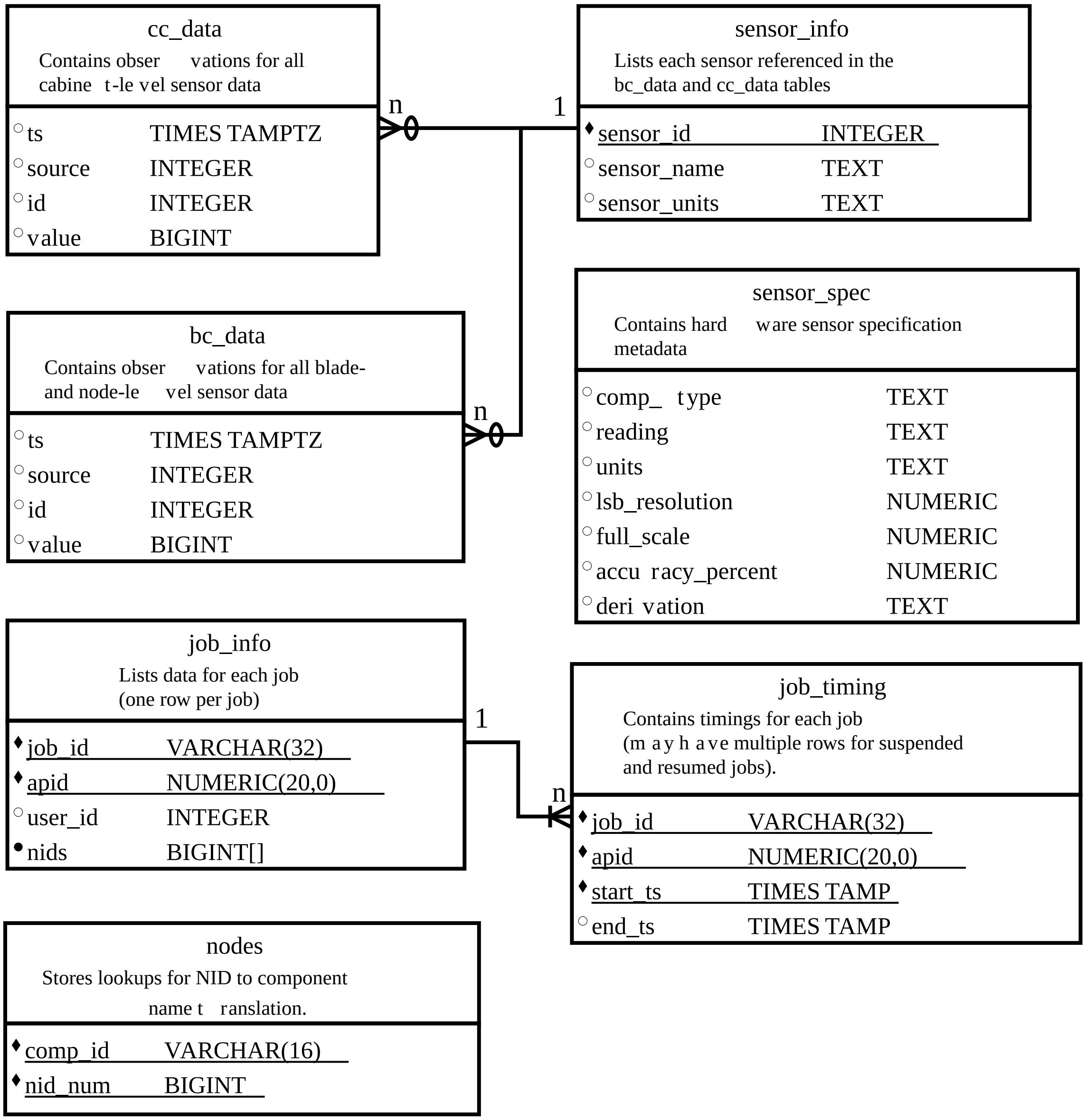
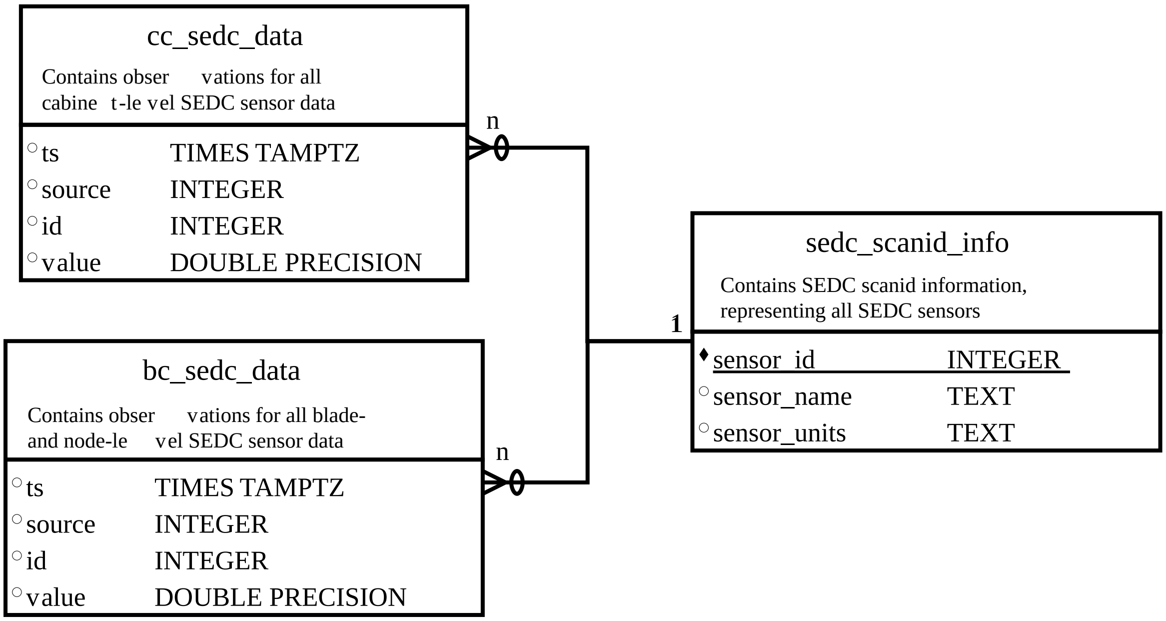


Figure 2. SEDC Tables



7.1 The PMDB Tables

The PMDB stores cabinet controller- and blade controller sensor data in two master tables, `pmdb.cc_data` and `pmdb.bc_data`. SEDC data is stored in the master `pmdb.cc_sedc_data` and `pmdb.bc_sedc_data` tables. Information about the SEDC sensors is stored in `pmdb.sedc_scanid_info`.

The sensor_info table for power monitoring and the `sedc_scanid_info` for the SEDC tables have the same table definition, as shown in *Power Management Tables* on page 22 and *SEDC Tables* on page 23.

Table 3. pmdb.sensor_info

Table Entry	Description
`sensor_id`	Integer specifying the sensor ID, which correlates with the ID field in the `pmdb.cc_data` and `pmdb.bc_data` tables.
`sensor_name`	Text field containing the name of the sensor, for example `Node 0 Power`.
`sensor_units`	Text field specify the unit of measure, for example, `W` for watts

The `pmdb.sensor_spec` table contains the following data:

Table 4. pmdb.sensor_spec

Table Entry	Description
`comp_type`	Text field specifying the component type.
`reading`	Text field containing what is being read.

Table Entry	Description
units	Numeric field specifying the units of measure.
lsb_resolution	Numeric field specifying the least significant bit resolution of the sensor.
full_scale	Numeric field containing the full scale of the sensor.
accuracy_percent	Numeric field containing the percent accuracy of a field. Typically this is about 2%.
derivation	Text field specifying how the value is derived, either computed or measured.

The pmdb.sedc_scanid_info table contains information about SEDC scanids, which represent sensors:

Table 5. pmdb.sedc_scanid_info

Table Entry	Description
sensor_id	Integer field specifying the SEDC scanid that represents a sensor. This field corresponds to the id field in the pmdb.cc_sedc_data and pmdb.bc_sedc_data tables. This field cannot be null.
sensor_name	Text field containing the name of the SEDC scanid.
sensor_units	Text field containing the units of measure for the sensor value.

The cc_sedc_data and bc_sedc_data tables contain data collected from cabinet-level and blade-level sensors, respectively:

Table 6. cc_sedc_data

Table Entry	Description
timestamp	Timestamp-with-time-zone field containing timestamp.
source	Integer field specifying the CC/BC controller that the data is from.
id	Integer field containing the SEDC scanid.
value	Double precision field containing the sensor value.

7.2 Query Usage at the Job Level

PMDB also stores job information for use in correlating power and energy to applications and jobs. Job-level querying is accomplished by querying the job attributes table, pmdb.job_info and the job timing table, pmdb.job_timing, then matching the results against the node-level data held in the blade controller data table, pmdb.bc_data. The pmdb.job_timing table supports multiple start-stop intervals for a single job_id-apid pair. To query at the job level, use the job_id, the ID assigned by the batch scheduler, along with an apid of 0. To query at the aprun level, use the job_id along with the apid of the aprun.

Because the batch scheduler and ALPS work in terms of NIDs and power management works with component names, it is necessary to translate between NIDs and component names to match the NIDs to sensor values. To translate a component name of a NID from the `pmdb.nodes` table to the PMDB-specific source value in `pmdb.bc_data` use the `cname2source` function provided with PMDB. Similarly, to translate a source value to a component name use the `source2cname` function, which is also provided with PMDB.

7.2.1 Sample Job-level Query Scripts

Cray provides a set of job-level query scripts that you can use as they are or as templates for creating your own reports based on the needs of your site.

The following example scripts are located on the SMW in the `/opt/cray/hss/default/pm/script_examples` directory.

`cray_pmdb_report_instant_power_all_jobs.sh`

This script reports the instantaneous power measurements by application ID (APID) using the `cray_pmdb_report_instant_power_all_jobs.sql` script. It does not take an argument. Sample output from a test system is below.

```
$ ./cray_pmdb_report_instant_power_all_jobs.sh
 APID | Watts
------+-------
 4413 |  1470
 3729 |   490
(2 rows)
```

`cray_pmdb_report_energy_single_job.sh`

This script reports the energy usage in Joules and KW/hour, and energy usage by component and NID. The script uses the `cray_pmdb_report_energy_single_job.sql` script, which takes an APID from an instance of `aprun` as an argument. Sample output from a short running application is below. Note that this script does not account for applications that might have multiple intervals because they were suspended and later resumed.

```
$ ./cray_pmdb_report_energy_single_job.sh 6517
 APID | Joules |          KW/h          |    Runtime
------+--------+------------------------+----------------
 6517 | 753523 | 0.20931194444444444444 | 00:15:05.00822
(1 row)

  Component   | NID | Joules
-------------+-----+--------
 c0-0c0s10n0 |  40 | 129841
 c0-0c0s10n1 |  41 | 128226
 c0-0c0s10n2 |  42 | 126521
 c0-0c0s10n3 |  43 | 127559
 c0-0c0s14n0 |  56 | 122688
 c0-0c0s14n1 |  57 | 118688
(6 rows)
```

`cray_pmdb_report_job_time_series.sh`

This script demonstrates how to handle jobs that were suspended and resumed. The script takes as an argument an APID from an instance of `aprun` to specify a particular job. It uses the `cray_pmdb_report_power_time_series_single_job_nid.sql` script iterating over all nodes used by the job to collect a time series for each overall interval for the job. The output is a CSV file, *`APID`*`.timeseries.csv`, which can be plotted and analyzed.

7.3 Query Usage at the Cabinet Level

A typical task for cabinet-level querying is to determine system-wide power and energy usage. Four sensors are scanned at 1 Hz from all cabinets and their data is collected in the pmdb.cc_data table:

```
qid |     sensor name       | units
---+---------------------+-------
 0 | Cabinet Power        | W
 2 | Cabinet Voltage      | mV
 3 | Cabinet Current      | A
 8 | Cabinet Blower Power | W
```

For example, the following SQL statement queries for the cabinet power for all cabinets in the system:

```
pmdb=> select ts, source2cname(source), \
value from pmdb.cc_data where id = 0 and ts in (select max(ts) from pmdb.cc_data group \
by source) order by source;

              ts               | source2cname | value
-------------------------------+--------------+-------
 2013-10-06 21:09:05.321138-05 | c0-0         |     0
 2013-10-06 21:09:05.646215-05 | c1-0         | 21008
 2013-10-06 21:09:05.147364-05 | c2-0         | 20936
 2013-10-06 21:09:05.152975-05 | c3-0         | 21106
 (4 rows)
```

To obtain the total power for the cabinet, add the cabinet power and cabinet blower power. Blower power collects cooling power for both XC liquid-cooled and XC-AC air-cooled systems. For example the following SQL query will return the total power for each cabinet:

```
pmdb=> select ts, source2cname(source), sum(value) \
from pmdb.cc_data where (id = 0 or id = 8) and ts in (select max(ts) \
from pmdb.cc_data group by source) group by source,ts;

              ts               | source2cname |  sum
-------------------------------+--------------+-------
 2013-10-06 21:12:52.614351-05 | c0-0         |  4440
 2013-10-06 21:12:52.435166-05 | c1-0         | 23328
 2013-10-06 21:12:52.364438-05 | c2-0         | 28061
 2013-10-06 21:12:52.632076-05 | c3-0         | 28827
(4 rows)
```

7.4 SEDC Data Storage on the PMDB

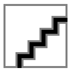
Effective with the 8.0UP02 release of the SMW software, the Power Management Database (PMDB) is the sole location for storing SEDC data. Legacy support for using group log files has been removed.

In the PMDB, the SEDC configuration is stored on the cabinet and blade controllers in the read-only `sedc.ini` file. By default the contents of the `sedc.ini` file is based on two Cray-provided JSON files, `blade_json.sedc` and `cab_json.sedc`, for blade-level and cabinet-level configuration, respectively. These files are located in the `/opt/cray/hss/default/etc` directory on the SMW.

Administrators can override the default SEDC configuration by creating custom JSON files for data collection at either the blade or cabinet level. Use the `sedc_enable_default` command to specify the path to the custom JSON files,and to specify a partition on which to enable the custom configuration. This begins the process that transfers the JSON files via `erfs` down to the controllers and into the `sedc.ini` file. If no options are specified, the `sedc_enable_default` command uses the default settings for storing sensor data to the PMDB. For more information, see the `sedc_enable_default(8)` man page.

Within the PMDB the SEDC schema is laid out like this:

Figure 3. PMDB SEDC Tables



The `pmdb.sedc_scanid_info` table contains information about SEDC `scanids`, which represent sensors:

sensor_id
Integer field specifying the SEDC `scanid` that represents a sensor. This field corresponds to the `id` field in the `pmdb.cc_sedc_data` and `pmdb.bc_sedc_data` tables. This field cannot be `null`.

sensor_name
Text field containing the name of the SEDC `scanid`.

sensor_units
Text field containing the units of measure for the sensor value.

Using the indexes `"sedc_scanid_info_pkey" PRIMARY KEY, btree (sensor_id)` will retrieve a list of sensor ID, sensor names, and sensor units:

```
  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
 1001,CC_V_VDD_2_5V,   V
 1002,CC_V_VDD_3_3V,   V
 1003,CC_V_VDD_3_3V_MICROA,   V
 1004,CC_V_VDD_3_3V_MICROB,   V
 1005,CC_V_VDD_5_0V,   V
```

The `cc_sedc_data` and `bc_sedc_data` tables contain data collected from cabinet-level and blade/node-level sensors, respectively:

timestamp
Timestamp-with-time-zone field containing timestamp.

source
Integer field specifying the CC/BC controller that the data is from.

id
Integer field containing the SEDC `scanid`.

value
Double precision field containing the sensor value.

The rules for the `cc_sedc_data` table are:

```
pmdb_cc_sedc_data_insert AS
ON INSERT TO pmdb.cc_sedc_data DO INSTEAD
INSERT INTO pmdb.cc_sedc_data_1 (ts, source, id, value)
VALUES (new.ts, new.source, new.id, new.value)
Number of child tables: 1 (Use \d+ to list them.)
```

For example:

```
pmdb=>SELECT ts,source2cname(source),value,id FROM pmdb.cc_sedc_data WHERE id=1011;

  ts                   | source2cname | value  |  id
-----------------------+--------------+--------+---------
2015-12-05 21:57:58.591468-05 | c0-0 | 16.75  |  1011
2015-12-05 21:58:28.610826-05 | c0-0 | 16.75  |  1011
2015-12-05 21:58:58.664612-05 | c0-0 | 16.75  |  1011
2015-12-05 21:59:28.686032-05 | c0-0 | 16.75  |  1011
2015-12-05 21:59:58.806264-05 | c0-0 | 16.75  |  1011
2015-12-05 22:03:25.476539-05 | c0-0 | 16.75  |  1011
2015-12-05 22:03:40.455622-05 | c0-0 | 16.75  |  1011
2015-12-05 22:03:55.484566-05 | c0-0 | 16.75  |  1011
2015-12-05 22:04:10.506321-05 | c0-0 | 16.75  |  1011
2015-12-05 22:04:25.526522-05 | c0-0 | 16.75  |  1011
2015-12-05 22:04:40.545297-05 | c0-0 | 16.75  |  1011
```

The rules for the `bc_sedc_data` table are:

```
pmdb_bc_sedc_data_insert AS
ON INSERT TO pmdb.bc_sedc_data DO INSTEAD
INSERT INTO pmdb.bc_sedc_data_1 (ts, source, id, value)
VALUES (new.ts, new.source, new.id, new.value)
Number of child tables: 1 (Use \d+ to list them.)
```

For example:

```
pmdb=>SELECT ts,source2cname(source),value,id FROM pmdb.bc_sedc_data WHERE id=1213;

  ts                          | source2cname | value  |  id
------------------------------+--------------+--------+---------
2015-12-05 22:38:52.071771-05 | c0-0c0s13    | 5.002  | 1213
2015-12-05 22:39:12.073579-05 | c0-0c0s13    | 5.002  | 1213
2015-12-05 22:39:32.078652-05 | c0-0c0s13    | 5.002  | 1213
2015-12-05 22:39:52.082684-05 | c0-0c0s13    | 5.002  | 1213
2015-12-05 22:40:12.087797-05 | c0-0c0s13    | 5.002  | 1213
```

Note that the nodes in a scanID are logical nodes, not physical nodes.

If the pmdb_auto_migrate service is enabled and PMDB_AUTO_MIGRATE='true' is set in `pmdb_migration.conf`, `sedc_enable_default` will be executed automatically by `systemd`. If not, the script must be executed manually immediately after system software installation. To verify that `sedc_enable_default` has been executed, use the following command to check for successful start of this service, which is a reliable indicator that `sedc_enable_default` also worked..

```
smw# systemctl status pmdb_auto_migrate
```

Note that configurations will be reset to default when booting to new versions of SMW software where the PMDB schema has changed.

7.4.1 Query the PMDB using the xtgetsedcvalues Script

The `xtgetsedcvalues` Python script extracts System Environment Data Collections (SEDC) data from the Power Management Database (PMDB) and returns the data fields as comma-separated values (CSV). The script supports three types of queries:

- A **data query** retrieves real SEDC data values that were reported by cabinet controllers (cc) or blade controllers (bc). This query returns the time, source (cname), numeric sensor ID, string sensor name, value, and unit of measure for each controller.
- A **sensor listing query** retrieves a listing of sensors with data recorded in the PMDB. This query returns the source (cname), numeric sensor ID, string sensor name, value, and unit of measure for each sensor specified.
- A **sensor table dump** retrieves information about all of the sensors in the sensor_info tables of the PMDB. This query returns the numeric sensor ID, string sensor name, and unit of measure.

> **IMPORTANT:** The `--type` (`-t`) argument, specifying either blade controllers (bc) or cabinet controllers (cc) is required for all SEDC queries.

For a data query use the `--type` option with either the bc or cc argument. Add the `--sensor-name-regex` with or without the `--sensor-ids` option to query specific sensor names and/or sensor IDs. For more information on these and other options, see the `xtgetsedcvalues` man page.

To query for a list of sensors for which there is data recorded in the PMDB use `--type` option with either the bc or cc argument, plus the `--list-sensors` option. To refine the information returned,use the `-c` option to limit the query to a specific list of physical IDs (cnames), and the `-s` and `-e` options to specify a time range to query. For more information on these and other options, see the `xtgetsedcvalues` man page.

To dump the contents of the sensor_info tables, use `--type` option with either the bc or cc argument, plus the `--dump-sensor-table` option. Note that this option dumps all entries in the PMDB sensor table, regardless of whether there is any data recorded for them.

Results are output to `sdtout` as comma-separated values or, by using the `--output` option, to a specified file. Because the potential exists for this output file to be quite large, it is recommended to also use the `--gzip` option to compress the output file.

7.4.2 Query PMDB for SEDC `scanid` Information

About this task

SEDC monitors sensors at cabinet level (`CC_` in the scanID name), blade level (`BC_` in the scanID name) and node level (`BC_`*x*`_NODE`*n*`_` in the scanID name).

The following example query returns a list of `sensor_ids` and the associated `sensor_name` and `sensor_unit`.

```
pmdb=> select * from pmdb.sedc_scanid_info;
sensor_id | sensor_name | sensor_units
----------+------------------------+-------------
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
1001 | CC_V_VDD_2_5V            | V
1002 | CC_V_VDD_3_3V            | V
1003 | CC_V_VDD_3_3V_MICROA     | V
1004 | CC_V_VDD_3_3V_MICROB     | V
1005 | CC_V_VDD_5_0V            | V
1006 | CC_T_COMP_AMBIENT_TEMP0  | degC
1007 | CC_T_COMP_AMBIENT_TEMP1  | degC
```

```
1008 | CC_T_COMP_WATER_TEMP_IN  | degC
1009 | CC_T_COMP_WATER_TEMP_OUT | degC
1010 | CC_T_COMP_CH0_AIR_TEMP0  | degC
. . .
```

Alternatively, this query prints the `sensor_id` information to a CSV file.

```
crayadm@smw> psql pmdb pmdbuser -t -A -F"," -c "select * from \pmdb.sedc_scanid_info" \
> ~/tmp/outfile-SEDC-scanids.csv
```

For an explanation of the options used in this query, see *Export Queries to a CSV File* on page 30 and the `psql` man page on the SMW.

7.4.3 Query PMDB for SEDC CPU Temperature Data

About this task

This query returns the number of cabinets within a specific range of IDs where there were CPUs with a temperature of 50°C or greater:

```
pmdb=> SELECT COUNT(*), source2cname(source) AS cname, id FROM pmdb.bc_sedc_data \
WHERE id >= 1300 AND id <= 1307 AND value >= 50, group by source, id;
count | cname | id
-----------------------
2 | c0-0c0s8 | 1302
2 | c0-0c0s8 | 1300
```

To determine the specific temperatures and the time of the events:

```
pmdb=> SELECT ts, source2cname(source) AS cname, id, value \
FROM pmdb.bc_sedc_data WHERE id >= 1300 AND ID <= 1307 AND value >= 50;
ts | cname | id | value
-------------------------------+----------+------+-------
2014-09-25 09:42:58.822325-05 | c0-0c0s8 | 1300 | 51
2014-09-25 09:43:38.916163-05 | c0-0c0s8 | 1300 | 51
2014-09-25 09:44:19.01072-05 | c0-0c0s8 | 1302 | 50
2014-09-25 09:44:59.058131-05 | c0-0c0s8 | 1302 | 51
(4 rows)
```

7.5 Export Queries to a CSV File

To run a query and have the output go to a comma-separated value (CSV) formatted file, run the query as:

```
smw:~> psql pmdb pmdbuser -t -A -F"," -c "query" output_filename
```

For example:

```
smw:~> psql pmdb pmdbuser -t -A -F"," -c "select * from pmdb.cc_data limit 5" \
> /tmp/outfile.csv
smw:~> cat /tmp/outfile.csv
2013-09-26 08:37:56.778032-05,202375168,0,17237
2013-09-26 08:37:56.778032-05,202375168,2,51900
2013-09-26 08:37:56.778032-05,202375168,3,332
2013-09-26 08:37:57.777829-05,202375168,0,16910
2013-09-26 08:37:57.777829-05,202375168,2,51898
```

The options passed to `psql` have the following meanings:

`-t`

Turns off printing of column names and result row count footers

-A

Specifies unaligned output mode

-F

Specifies the field separator, in this case ","

-c

Specifies the query string to execute

For more information on using the psql command-line interface to PostgreSQL, see the psql man page on the SMW.

7.6 Tune the PMDB

About this task

The xtpgtune utility automatically tunes configuration of the PostgreSQL cluster for the Cray Platform Management Database (PMDB) for optimal performance. Alternatively, sites with larger systems can disable automatic tuning and manually tune the PMDB by modifying the default PostgreSQL settings. A well-tuned PMDB will speed up report queries and minimize disk I/O for transaction check-pointing.

By default, the systemd service calls xtpgtune prior to starting the postgreSQL database on the SMW or the database node. The script detects the server hardware configuration, creates a backup of the previous configuration, then edits the PostgreSQL configuration file, postgresql.conf, directly.

System administrators can also manually invoke the xtpgtune script from root. Generally, this is not necessary as the script runs automatically at every SMW boot.

To disable autotuning, edit the pmdb_migration.conf file and set PMDB_AUTO_TUNE to False.

Manual tuning begins by locating the database configuration file named postgresql.conf. On a default installation, that file is located in /var/lib/pgsql/data/postgresql.conf and is owned by user postgres. If the configuration file is not in that location, open a psql prompt as user postgres and execute a show config_file query:

Procedure

1. Log on as root:

```
smw > su -
```

2. Become user postgres to obtain file location:

```
smw # su - postgres
postgres=# psql -c "show config_file"
              config_file
 ------------------------------------
 /var/lib/pgsql/data/postgresql.conf
(1 row)
```

3. Return to root and edit the `postgresql.conf` file:

```
smw > su -
```

4. Modify the configuration file as described in *PMDB Tuning Options* on page 32:

```
smw # cd /var/lib/pgsql/data
smw # vi postgresql.conf
```

5. When editing is complete, verify that the permissions have not changed:

```
smw # ls -la postgresql.conf
-rw------- 1 postgres postgres 19178 Dec 13  2012 postgresql.conf
```

6. Restart the Cray management system (RSMS) and the PMDB:

```
smw # rsms stop
smw # systemctl restart postgresql
smw # rsms start
```

IMPORTANT: Be sure to stop RSMS before restarting PMDB, then restart RSMS.

7.7 PMDB Tuning Options

The following examples describe a subset of the tunable parameters in PostgreSQL. These tuning suggestions assume a typical CPU-only 10-cabinet XC series system and an SMW with 8GB of memory. Note that these options are tuned automatically by the `xtpgtune` script and need not be manually updated. The following is informational.

IMPORTANT: If the SMW shows significant signs of swap usage then the SMW does not have enough memory installed. The hardware will need to be upgraded before performing any database tuning.

shared_buffers

The `shared_buffers` setting should be configured between 15%-25% of installed memory. A good starting point is 20%. If, after tuning, you notice swapping, adjust this setting down to 15% of RAM. If swapping persists after lowering to 15% Cray recommends installing additional RAM in the SMW.

Because PostgreSQL uses shared memory verify the operating system is configured sufficiently. The `shared_buffers` setting cannot be less that 1 GB and cannot be larger than the `kern.shmmax` setting. To determine this setting:

```
smw # /sbin/sysctl kernel.shmmax
kernel.shmmax = 18446744073709551615
```

effective_cache_size

The cache size is an estimate of how much memory is available to the operating system for caching. Use the `free` command to determine memory availability.

```
smw:~> free
             total       used       free     shared    buffers     cached
Mem:      16347816   14609000    1738816     169772         60   13024860
-/+ buffers/cache:    1584080   14763736
Swap:     33559420    1278800   32280620
```

Add the values given for `free` and `cached` to obtain a reasonable estimate. Choose the smaller of this result and 50% of the system RAM. In this example, the sum of the free and cached memory is approximately 7.4GB, so the effective cache size should be set to 50% of the RAM, or 4GB. Be aware that this setting is an estimate, not a memory allocation.

checkpoint_completion_target

Postgres syncs dirty pages from the shared buffers to disk during each checkpoint. The completion target is a setting that effectively limits the amount of checkpoint-related disk I/O during this time. The usable values are between 0.5 (the default) and 0.9. To lower the average write overhead, increase this parameter to 0.9.

max_connections

Set to 500

max_locks_per_transaction

Set to 256. Setting `max_connections` and `max_locks_per_transaction` allows for enough memory for locking and connection data structures (about 32MB).

For additional guidance on tuning the PMDB see *http://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server*.

7.8 Check or Configure PMDB with the pmdb_util Command

The power monitoring database is checked, configured, and repaired using the `pmdb_util` command line tool. (The `pmdb_util` command supercedes the `xtpmdbconfig` command, which is no longer supported.) The `pmdb_util` tool is documented in greater detail in the `pmdb_util(8)` man page or access help by entering `pmdb_util -h`. The following discussion focuses on the more commonly used subcommands.

Checking the PMDB

To check the PMDB to make certain all of its features are working properly, use the `pmdb_util check --all` command, as shown in this example:

```
smw:~ # pmdb_util check --all
[check_data_directory()]:     INFO: Data directory exists and matches installed version of PostgreSQL.
[  tune_configuration()]:     INFO: xtpgtune successfully tuned configuration. Output:
[  tune_configuration()]:     INFO:     >>> PostgreSQL configuration is already tuned.
[ check_pmdbuser_auth()]:     INFO: User entry found.
[ check_pmdb_database()]:     INFO: pmdb database exists in PostgreSQL.
[     check_pmdb_user()]:     INFO: pmdbuser exists in PostgreSQL.
[     check_functions()]:     INFO: PMDB functions exist in PostgreSQL.
[        check_schema()]:     INFO: erfs schema exists in pmdb database in PostgreSQL.
[        check_schema()]:     INFO: sdbhwinv schema exists in pmdb database in PostgreSQL.
[        check_schema()]:     INFO: diags schema exists in pmdb database in PostgreSQL.
[        check_schema()]:     INFO: sm schema exists in pmdb database in PostgreSQL.
[        check_schema()]:     INFO: hsslocks schema exists in pmdb database in PostgreSQL.
[        check_schema()]:     INFO: pmdb schema exists in pmdb database in PostgreSQL.
[           check_all()]:     INFO: -----------------------------------------------
[           check_all()]:     INFO: RESULTS:
[           check_all()]:     INFO: ---   pmdbuser_auth: SUCCESS ---
[           check_all()]:     INFO: ---   pmdb_database: SUCCESS ---
[           check_all()]:     INFO: ---       pmdb_user: SUCCESS ---
[           check_all()]:     INFO: ---       functions: SUCCESS ---
[           check_all()]:     INFO: ---     erfs_schema: SUCCESS ---
[           check_all()]:     INFO: --- sdbhwinv_schema: SUCCESS ---
[           check_all()]:     INFO: ---    diags_schema: SUCCESS ---
[           check_all()]:     INFO: ---       sm_schema: SUCCESS ---
[           check_all()]:     INFO: --- hsslocks_schema: SUCCESS ---
[           check_all()]:     INFO: ---     pmdb_schema: SUCCESS ---
[           check_all()]:     INFO: PMDB passed all checks!
[           check_all()]:     INFO: -----------------------------------------------
```

The `pmdb_util check --all` command automatically attempts to fix any issues found with the PMDB in a non-destructive way. If desired, this auto-fix functionality can be disabled by passing the `--no_auto_fix` option, as in this example: `pmdb_util check --all --no_auto_fix`.

If an automatic fix would destroy data, `pmdb_util check` will produce an error message indicating that manual intervention is required. For example:

```
[  config_pmdb_schema()]: CRITICAL: The pmdb schema seems
to be intact but this function was not given explicit permission
to clobber the existing data! Run again with the --clobber_ok flag
to force replace this schema.
```

To force-fix the issue, re-run `pmdb_util check` with the `--clobber_ok` option, for example: `pmdb_util --clobber_ok check --all`.

By default, `pmdb_util` is set as a requirement for PostgreSQL to start, and if any issues are found it wil attempt to resolve them as described above. If an issue cannot be resolved, PostgreSQL will not be allowed to start:

```
smw:~ # systemctl start postgresql
A dependency job for postgresql.service failed. See 'journalctl -xe' for details.
```

To determine why PostgreSQL can't start, use the `systemctl status pmdb_util` command or examine the `/var/opt/cray/log/smwmessages*` log to see what went wrong.

Configuring the PMDB

To start fresh and reinitialize the PMDB, run the `pmdb_util config --init` command. For example:

```
smw:~ # pmdb_util config --init
[     config_data_dir()]:     INFO: Configuring PostgreSQL data directory...
[     config_data_dir()]:     INFO: Old data directory removed.
[     config_data_dir()]:     INFO: New data directory successfully created. Output from initdb:
[     config_data_dir()]:     INFO:     The files belonging to this database system will be owned by user "postgres".
[     config_data_dir()]:     INFO:     This user must also own the server process.
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:     The database cluster will be initialized with locale "en_US.UTF-8".
[     config_data_dir()]:     INFO:     The default database encoding has accordingly been set to "UTF8".
[     config_data_dir()]:     INFO:     The default text search configuration will be set to "english".
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:     Data page checksums are disabled.
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:     creating directory /var/lib/pgsql/data ... ok
[     config_data_dir()]:     INFO:     creating subdirectories ... ok
[     config_data_dir()]:     INFO:     selecting default max_connections ... 100
[     config_data_dir()]:     INFO:     selecting default shared_buffers ... 128MB
[     config_data_dir()]:     INFO:     selecting dynamic shared memory implementation ... posix
[     config_data_dir()]:     INFO:     creating configuration files ... ok
[     config_data_dir()]:     INFO:     running bootstrap script ... ok
[     config_data_dir()]:     INFO:     performing post-bootstrap initialization ... ok
[     config_data_dir()]:     INFO:     syncing data to disk ... ok
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:     WARNING: enabling "trust" authentication for local connections
[     config_data_dir()]:     INFO:     You can change this by editing pg_hba.conf or using the option -A, or
[     config_data_dir()]:     INFO:     --auth-local and --auth-host, the next time you run initdb.
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:     Success. You can now start the database server using:
[     config_data_dir()]:     INFO:
[     config_data_dir()]:     INFO:         pg_ctl -D /var/lib/pgsql/data -l logfile start
[  tune_configuration()]:     INFO: xtpgtune successfully tuned configuration. Output:
[  tune_configuration()]:     INFO:     >>> Wrote backup of /var/lib/pgsql/data/postgresql.conf to /var/lib/pgsql/data/
postgresql.conf-2017-10-19T11:01:00.817061
[  tune_configuration()]:     INFO:     >>> Wrote configuration to /var/lib/pgsql/data/postgresql.conf
[config_pmdbuser_auth()]:     INFO: Configuring pmdbuser auth entry...
[ check_pmdbuser_auth()]:    ERROR: User entry NOT found!
[config_pmdbuser_auth()]:     INFO: Adding pmdbuser entry to pg_hba.conf file...
[config_pmdbuser_auth()]:     INFO: User entry added.
[    config_pmdb_user()]:     INFO: Creating pmdbuser in PostgreSQL...
[    config_pmdb_user()]:     INFO: Created database user successfully.
[config_pmdb_database()]:     INFO: Configuring the pmdb database in PostgreSQL...
[config_pmdb_database()]:     INFO: Created the pmdb database successfully.
[        check_schema()]:    ERROR: pmdb schema does NOT exist in pmdb database PostgreSQL!
[  config_pmdb_schema()]:     INFO: Installing the XTPMD schema...
[  config_pmdb_schema()]:     INFO: XTPMD schema successfully installed.
[   config_extensions()]:     INFO: Installing extensions...
[   config_extensions()]:     INFO: xtpmd extensions installed.
```

```
[   config_extensions()]:     INFO: erfs extensions installed.
[   config_extensions()]:     INFO: Installation of dir_fdw extension and the diags schema succeeded.
[   config_extensions()]:     INFO: sbdhwinv extension installed.
[   config_extensions()]:     INFO: hsslocks extension installed.
[          initialize()]:     INFO: ---------------------------------------------
[          initialize()]:     INFO: PMDB Initialization SUCCEEDED
[          initialize()]:     INFO: ---------------------------------------------
```

Any facet of the PMDB can be configured by using `pmdb_util config` with one or more of the specific options. For more information, enter the `pmdb_util config -h` command.

7.9 Set Disk Storage Parameters

Edit the `/var/adm/cray/pmdb_migration/pmdb_migration.conf` file to set the disk storage parameters for a job. The options are:

PMDB_AUTO_PRUNE

If true, the PMDB database will be pruned periodically of old data (power and job information). Which data is removed and when is determined by the `PMDB_MAX_SIZE` and `PMBD_JOBS_NUM_DAYS` parameters.

Default: `PMDB_AUTO_PRUNE='true'`

PMDB_MAX_SIZE

The maximum size in megabytes that the PMDB database will be allowed to grow to. The data removed starts with the oldest and progressively works to newer data until the target is hit. Note that there is some delay between pruning runs, so it is possible that usage could grow beyond this number, especially on larger systems. If the disk at the mount point `/var/lib/pgsql` is also used for other data, this setting should be set such that the PMDB cannot grow into other data.

Default: `PMDB_MAX_SIZE=512000`

PMDB_JOBS_NUM_DAYS

The number of days' worth of job information to keep in the PMDB. All jobs older than this will be pruned from the database. To always keep all job information, set this to `0` (disabled).

Default: `PMBD_JOBS_NUM_DAYS=30`

The administrator must decide how much of the disk space to use for PMDB and set the `PMDB_MAX_SIZE` accordingly. The data in the `job_info` and `job_timing` tables stored for the amount of time specified in `PMBD_JOBS_NUM_DAYS`.

If pruning is enabled, the PMDB is kept to the size indicated.

If pruning is disabled (set to 0), the administrator must manually prune `pmdb.job_info` and `pmdb.job_timing` tables in the PMDB. When pruning is disabled, it is possible for job data ***to consume the entire disk causing the PMDB to crash***.

7.10 Manual Backup and Recovery of the PMDB

It may be useful to backup the PMDB if, for example, a particular time interval of data should be saved for historical purposes. Also an update of the SMW software will preserve the configuration information but not the collected data, so you may wish to back up the PMDB prior to updating the SMW software. For backup and restoration of the PMDB two utilities, `pg_dump` and `pg_restore` are included with PostgreSQL on the SMW software distribution.

To dump the contents of the PMDB use the following command:

```
$ pg_dump -c -U pmdbuser pmdb > pmdb.dump.yyyymmdd.sql
```

Dump files created with `pg_dump` in this way are created in plain text and can be created consistently even while PMDB is in use.

> **NOTE:** Depending on the size of the database, execution of `pg_dump` may take a long time. For example, a PMDB of 100 blade table partitions each with 2 million rows and 50 cabinet table partitions each with 100,000 rows will produce a plain text dump file of about 8 GB in size and will take approximately 10 minutes to generate.

Use the `psql` utility to restore the PMDB from the backup created with `pg_dump`. Note that prior to restoring PMDB from a backup it is necessary to stop the `xtpmd` daemon and that doing so will stop all HSS functionality.

```
# rsms stop
$ psql -U pmdbuser pmdb < pmdb.dump.yyyymmdd.sql
# rsms start
```

Alternatively, create a dump with the `pg_dump` custom dump format, which uses the `zlib` compression library to compress the output:

```
$ pg_dump -U pmdbuser -Fc pmdb > pmdb.dump.yyyymmdd
```

To restore from the `pg_dump` custom format, use `pg_restore`:

```
# rsms stop
$ pg_restore -U pmdbuser -Fc pmdb.dump.yyyymmdd
# rsms start
```

8 Export Power Data to a Network Management Station via SNMP

Effective with the 8.0 version of the SMW software, Cray provides support for exporting power data to network management stations that use SNMP. A new daemon, `xtsnmpd`, acts as an AgentX subagent under the Net-SNMP-provided snmpd master agent. While the master agent responds for default, SLES- and NET-SNMP-provided management information bases (MIBs), `xtsnmpd` responds for two Cray-specific MIBs, `CRAY-XC-MIB`. and `CRAY-SMI`. The `CRAY-SMI` MIB provides the structure of management information for the overall Cray enterprise. The `CRAY-XC-MIB` MIB contains system-level power data, including system-level instantaneous/current power, peak power, average power, and accumulated energy. These files are stored for reference on the SMW under `/opt/cray/hss/default/etc/snmp/mibs`. They are also placed in the Net-SNMP MIBs directory on the SMW so Net-SNMP client programs on the SMW can reference them automatically.

The `xtsnmpd` daemon is considered system-wide for SNMP clients with overall system access, so partition rights/boundaries do not apply.

Administrators have no direct interaction with `xtsnmpd`; all adminstrative tasks are handled via the `xtsnmpd_setup` command.

IMPORTANT: The `xtsnmpd_setup` command must be run as `root`.

By default the `xtsnmpd` daemon is disabled. To enable the daemon:

```
smw:~ # xtsnmpd_setup --on
```

Be aware that it can take up to 60 seconds for `xtsnmpd` to register with the snmpd master agent, during which time Cray SNMP objects may not be available.

To disable the daemon:

```
smw:~ # xtsnmpd_setup --off
```

To create a new user named "username" with the authentication password "secret" and privacy password "supersecret":

```
smw:~ # xtsnmpd_setup --create_user username \
--new_auth_pw secret --new_priv_pw supersecret
```

To remove a user named "user_to_delete" the command must be invoked using the credentials of either the user being removed or some other user. In this example, the "master_user" has the authentication password "secret" and privacy password "supersecret":

```
smw:~ # xtsnmpd_setup --remove_user user_to_delete \
 --user master_user --auth_pw secret --priv_pw supersecret
```

The `create_user` and `remove_user` actions both require an SNMP restart. It can take up to 60 seconds for the `xtsnmpd` daemon to register with the `snmpd` master agent, during which time Cray SNMP objects may not be available.

To change a user's password, remove the user then add the user again with new password(s).

For more information, see the `xtsnmpd_setup(8)` man page.

9 User Access to P-state Management

The P-state is the CPU frequency used by the compute node kernel while running the application. A performance governor is the kernel algorithm used to dynamically maintain the CPU frequency of the node. To affect the power and/or performance of an ALPS-submitted job, users on the login node can specify either a P-state or a performance governor to be used on the nodes running their application. Note that users can specify one or the other, but not both.

9.1 Set a P-state in an aprun Command

Cavium Processors

CLE 6.0 UP07 supports Cavium™ ARM processors in XC series systems. Instead of supporting a list of valid frequencies, Cavium processors support minimum and maximum frequencies, and any frequency between the minimum and maximum can be set. To find the minimum and maximum frequencies on Cavium processor compute node, run the following commands:

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_min_freq
# cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_max_freq
```

To set a P-state (i.e. frequency) using aprun, use the --p-state option and specifiy the desired frequency in KHz. The requested frequencies will be silently bounded by the minimum and maximum frequencies; e.g. a requested frequency lower than the minimum will be silently set to the minimum.

> **IMPORTANT:** While the scaling_min_freq and scaling_max_freq files provide accurate frequency bounds available for control, these values can be modified when frequency limits are set via the capmc set_freq_limits applet. For this reason, always determine the minimum and maximum frequencies by reading the cpuinfo_min_freq and cpuinfo_max_freq files.

Intel Processors

To set a P-state using aprun, use the --p-state option and specify the desired frequency in KHz. To find a list of available frequencies, run the following command on an Intel-based compute node:

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
```

If the requested frequency does not match one of the available frequencies, the value of p-state is rounded down to the next supported frequency.

9.2 Set a Performance Governor in an `aprun` Command

To specify a performance governor in an `aprun` command, use the `--p-governor` option, specifying the performance governor to be used by the compute node kernel while running the application. To find a list of available performance governors, run the following command on a compute node:

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
```

To find the default performance governor:

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

For example, to find the available governors for the node with the NID of 40:

```
login:~> aprun -n 1 -L 40 cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
conservative ondemand userspace powersave performance
Application 13981 resources: utime ~0s, stime ~0s, Rss ~3616, inblocks ~54, outblocks ~93
```

Then, to run a job (*`hostname`*) on NID 40 with the `powersave` governor:

```
login:~> aprun -n 1 -L 40 --p-governor=powersave hostname
nid00040
Application 13982 resources: utime ~0s, stime ~0s, Rss ~3616, inblocks ~19, outblocks ~28
```

9.3 Use Workload Managers (WLMs) with BASIL

The Batch Application Scheduling Interface Library (BASIL) supports the ability of WLMs to select a fixed `P-state`, or alternate `P-state` governor at reservation time. For example:

```
<BasilRequest protocol=1.4 method=RESERVE>
<ReserveParamArray user_name=username batch_id=id>
<ReserveParam architecture=XT width=1 p-state=10>
<ReserveParam architecture=XT width=1 p-governor=p-gov-name>
</ReserveParamArray>
</BasilRequest>
```

Again, `p-state` and `p-governor` are mutually exclusive.

Be aware that any `aprun` commands within this reservation will inherit the `p-state` or `p-governor` settings from the reservation. If an invocation of `aprun` uses its own `p-state=khz` option, the value of *`khz`* must be equal to or lower than the value of `p-state` in the reservation. Similarly if an invocation of `aprun` uses its own `p-governor` option, it must match any `p-governor` specified in the reservation.

10 User Access to Power Management Data

Users on an XC Series system where power management is enabled have access to compute node power and energy data on a set of files located in `/sys/cray/pm_counters/`. These files are:

`power`	Point-in-time power, in Watts. When accelerators are present, includes `accel_power`. See limitation below on data collection from accelerators.
`energy`	Accumulated energy, in joules. When accelerators are present, includes `accel_energy`. See limitation below on data collection from accelerators.
`generation`	A counter that increments each time a power cap value is changed.
`startup`	Startup counter.
`freshness`	Free-running counter that increments at a rate of approximately 10Hz.
`version`	Version number for power management counter support.
`power_cap`	Current power cap limit, in Watts; 0 indicates no capping.
`accel_energy`	Accumulated accelerator energy, in joules, if accelerator is present. See limitation below on data collection from accelerators.
`accel_power`	Accelerator point-in-time power, in Watts, if accelerator is present. See limitation below on data collection from accelerators.
`accel_power_cap`	Current accelerator power cap limit, in Watts, if accelerator is present; 0 indicates no capping.

Note that each node has a power supply that can support a fixed number of Watts. The combined power consumption of the CPU and the accelerator can never exceed this limit, thus, power to either the CPU or the accelerator must be capped so as not to exceed the total amount of wattage available.

Limitation for NVIDIA® Pascal® GPUs

Due to certain hardware limitations data collected in the PMDB for Pascal GPU nodes does not include accelerator-level power data and is limited to the node-level.

11 Measure Per-Job Energy Usage

About this task

This procedure calculates the energy consumed by XC Series compute nodes only. It does not include energy consumption by the service nodes or network resources that were used in the job.

Procedure

1. Before starting a job, use the files described in *User Access to Power Management Data* on page 41 to record the startup and energy data values for each node that will be used to run the job.

2. Run the job.

3. After the job completes, record the startup and energy data values again. Verify that the startup value has not changed. If it has changed, a blade-controller was restarted and the measurements are not valid.

4. If the startup value has not changed, for each node subtract the energy value at job start from the energy value at job completion. This is the energy consumed by each node during the job.

5. Add the energy consumption values for each node to derive the total energy consumed by the compute nodes.

   Users can specify a location in their home directory where Resource Utilization Reporting (RUR) will write the computed energy for a job.

12 Create a Remote Power Management Database

There are several reasons why it may be desirable to move the PMDB from the SMW to dedicated hardware. A site may need to have logging and telemetry data streamed from the XC series system to site-specific data center infrastructure management (DCIM) systems, or to store more power data than the SMW can support. For security purposes it can be useful to separate the monitoring from the control facilities on the SMW. Especially on large system configurations, it can be useful to isolate the PMDB from the rest of the critical infrastructure on the SMW.

Keep in mind that, with this configuration, only the power monitoring and SEDC data moves to the remote database. Other applications on the PMDB that are not typically accessed by administrators, such as `erfsd` and `xtdiagd`, continue to use the PostgreSQL instance running on the SMW. Thus, other documentation may refer to the remote PMDB (or database node) or to the on-SMW PMDB.

12.1 Configure the SMW to Support a Remote PMDB

Prerequisites

Requires an available hardware server similar to the SMW, such as a Dell R630, with a minimum of 128GB of RAM and several TB of HDD storage with a 512-byte sector size, and H330 RAID controller, and a four port Ethernet interface. In addition, remote-media-based installation (DVDISO) requires iDRAC Enterprise.

About this task

This procedure details steps to configure the SMW to use a remote PMDB. If this is an update to an existing remote PMDB node, most of these steps will have already been completed, however it is recommended to verify that the necessary configurations are still in place.

Procedure

1. Edit the `/etc/dhcpd.d/dhcpd.static.conf` file on the SMW to configure the DHCP server to provide a host name and fixed IP address based on an Ethernet MAC address. The Ethernet MAC address specified must match that of the database node's Ethernet port that is connected to the HSS management network. The IP address assigned to the database node should fall within the IP range of 10.1.1.10 - 10.1.1.25 with hostnames `dh0` - `dh15`, respectively. This follows the convention used in the database node's preconfigured `/etc/hosts` file. The IP address range chosen does not conflict with other reserved IP ranges on the HSS network.

   Placing the static host configuration within a `group` directive allows overriding the global `default-lease-time` and `max-lease-time` settings to more appropriate values, which have been optimized for PXE booting the HSS controllers.

   Note that this configuration code should be appended to the end of the `dhcpd.static.conf` file. Changes made to this file will persist across system software upgrades.

```
group {
       default-lease-time 900;
       max-lease-time 7200;
       host dh0 {
                 option host-name "dh0";
                 hardware ethernet D4:AE:52:E7:6B:CC;
                 fixed-address 10.1.1.10;
                }

}
```

2. Restart dhcpd to load the new configuration file.

```
smw# systemctl restart dhcpd
```

3. Edit the /etc/hosts file on the SMW to map the fixed IP address of the additional database node to a hostname. The IP address to host name entry must match that assigned by the DHCP server.

```
10.1.1.10 dh0
```

4. Configure the Event Router Daemon (ERD) on the SMW.

   The event router daemon has two main configuration files, /opt/cray/hss/default/etc/erd.ini, which configures general runtime settings and /opt/cray/hss/default/etc/erd.broadcast.conf, which configures event disposition, or message routing. The important parameters that must be configured include which peers (database nodes) the SMW's event router should connect to, and the name of the SMW. Upon startup the Event Router will automatically remove itself from thepeer_hosts list in accordance with its own hostname. Therefore, each database node and the SMW can have an identical peer_hosts setting.

   Edit erd.ini to add these lines:

```
hostname=smw
#Specify multiple database nodes as a semicolon-delimited list
peer_hosts=dh0
```

   a. Edit the erd.ini file to instruct the cabinet controller's event router to connect to the upstream database node.

      The cabinet controller event routers download this customized erd.ini using TFTP upon boot. The SMW's IP address is implicitly included in this list. All cabinet controllers will connect to the SMW. Specify additional upstream hosts as a semi-colon-separated list of IPv4 addresses. The address has the form regex:ip, where regex is a Perl-compatible regular expression.

      On startup the event router daemon for each cabinet controller matches its own hostname against the regular expression for each item in the upstream hosts list. If a match is made, then that cabinet controller initiates a connection to the specified host. If there is no match, the cabinet controller ignores this entry.

      The regex: portion of the address is optional and if it is omitted the cabinet controller will always attempt to connect to the specified IP address.

      This example appends the l1_upstream_hosts parameter (from the upstream host defined earlier in this topic) to the SMW erd.ini file:

```
smw# echo "l1_upstream_hosts=10.1.1.10" >> /opt/cray/hss/default/etc/erd.ini
```

   b. Edit erd.ini to enable the aggregate flow stat collector. This allows all top level nodes to send event traffic statistics to a single logger. The database image is configured to automatically start the flow capture daemon, erd-flowcapd. This step is optional, but recommended.

```
flow_export=true
#The ‘flow_collector’ setting must be an IPv4 address.
#The ERD flow export code can not resolve hostnames.
flow_collector=10.1.1.10
```

c. Optional: Edit the `erd.broadcast.conf` file on the SMW to prevent power data from flowing into the SMW by commenting out the line containing `ec_power_data`. If `sedc_manager` is configured to use the PMDB for data collection (the default) instead of the legacy flat files, also comment out the line containing `ec_sedc_data`.

   The default configuration enables all traffic that would normally flow into the SMW to flow into the database node as well. This step optimizes traffic so that only environmental data is be allowed to flow up into the database node. Leave the remaining configuration parameters unchanged

```
# ec_power_data = up_smw
# ec_sedc_data = up_smw
```

5. Edit the `/opt/cray/hss/default/etc/pmdb_conn_settings.ini` file to allow `xtremoted`, the application server that handles CAPMC API requests, to connect to the remote database.

By default `xtremoted` uses the PostgreSQL server on the SMW. Update the `hostname`, `database`, and `username` parameters. A password is not required.

```
[pmdbconn]
hostname=dh0
database=pmdb
username=pmdbuser
```

6. Edit the `/opt/cray/hss/default/etc/xtpmd.ini` file on the SMW to configure the XT Power Management Daemon, `xtpmd`, to start up in passive mode with database logging disabled.

Starting up in passive mode prevents `xtpmd` from responding to total system power requests from `xtpowerd` or broadcasting total system power to cabinet or blade controllers. In this mode, `xtpmd` will continue running any custom data output plugins if so configured.

```
[xtpmd]
passive=true
enable_database=false
```

7. Restart `rsms` to restart the ERD and `xtpmd`, and to read the new configurations.

```
smw# rsms restart
```

8. Push the changes out to the blade and cabinet controllers.

a. Shut down the system.

   smw# **`xtbootsys –s last –a auto.xtshutdown`**

b. Reboot the cabinet controllers to use the new image.

```
smw# xtcli power down s0
smw# xtccreboot –c all
smw# xtcli power up s0
smw# xtalive –l cc
```

c. Run `xtbootsys` with the site-appropriate autofile to boot the system with the new image.

```
smw# xtbootsys -a autofile
```

The SMW is configured to work with the off-SMW database node.

12.2 Configure and Create a Remote PMDB Image

Prerequisites

Requires the KIWI imaging tool, included in the SLES 12 SDK. This tool is installed on the SMW.

About this task

The SMW has been configured to support the database node, as described in *Configure the SMW to Support a Remote PMDB*. Note that whenever the operating system on the SMW is updated it is necessary to build a new PMDB image, as the software for the SMW and the database node are tightly coupled.

An RPM called `cray-pmdb-image`, installed with the SMW software, contains a pre-configured template for building the node image and a makefile that calls the appropriate KIWI command to build a bootable installation ISO for deployment on the database node. This template is installed to `/usr/share/cray-pmdb-image`.

Before building the image, certain parameters such as packaging list, credentials, and timezone need to be configured. This is done by editing the xml configuration file, `cray-pmdb/config.xml.in`.

To customize the image content, edit the node image configuration files located in an overlay directory at `cray-pmdb/root`. Any files placed in this directory hierarchy are copied in archive mode after KIWI has finished installing packages in the temporary root filesystem, but before building the image. For example, if the `eth0` network interface settings need to be customized, edit the system configuration file `cray-pmdb/root/etc/sysconfig/network/ifcfg-eth0`.

Note that the HSS configuration directory is not placed under the default symlink in the image overlay hierarchy. The script `config.sh` takes care of moving the override files into the expected HSS configuration directory `/opt/cray/hss/default/etc` prior to constructing the image.

Procedure

1. Install source control software on the SMW if it does not already exist. This procedure uses `git`, which can be installed from the SLES 12 SDK repository on the SMW, using the `zypper` command.

```
smw# zypper install git
```

2. Create a repository for the current pmdb image template.

   a. Copy the pmdb image template to a directory containing the required free space.

   ```
   smw# cp -a /usr/share/cray-pmdb-image .
   ```

   b. Change to the image directory and initialize a `git` repository.

   ```
   smw# cd cray-pmdb-image
   smw# git init
   Initialized empty Git repository in /home/crayadm/cray-pmdb-image/.git/
   ```

   c. Add all of the files from the default template and perform the initial commit.

```
smw# git add *
smw# git commit -a -m "Initial Commit"
[master (root-commit) 5942d43] Initial Commit
47 files changed, 2782 insertions(+)
create mode 100644 Makefile
create mode 100755 cray-live/config.sh
create mode 100644 cray-live/config.xml.in
...
smw# git branch
* master
```

Note that the `git branch` command shows that there is a single branch called `master`. This branch contains the unmodified pmdb template. When the system is upgraded new versions of the default pmdb image template will be committed to the master branch, then merged into the customization branch.

Customizations can include setting the timezone, adding additional packages, and configuring passwordless ssh. Do not make these changes to the master branch. Instead, create a new branch that uses the master as a starting point.

3. Create a new branch called `pmdb`, then customize the image in that branch.

```
smw# git checkout -b pmdb master
        Switched to a new branch 'pmdb'
```

4. Verify that the current working branch is `pmdb`.

In `git` the current branch is identified by a leading asterisk.

```
smw# git branch
          master
        * pmdb
```

5. Edit the time zone parameter in the image configuration file, `cray-pmdb-image/cray-pmdb/config.xml.in`, to set the local time. Acceptable values are file paths relative to `/usr/share/zoneinfo`. The default is UTC.

To change the time to America/Chicago (Central Time Zone)

```
<bootloader-theme>SLE</bootloader-theme>
<locale>en_US</locale>
<keytable>us.map.gz</keytable>
<timezone>America/Chicago</timezone>
<hwclock>utc</hwclock>
<rpm-excludedocs>false</rpm-excludedocs>
</preferences>
```

6. Configure the RAID controller to specify the system storage space.

By default the RAID controller must be configured to provide a system disk of 500GB or more to install the operating system on and another disk to install the database on. This can be done by entering the BIOS setup or by using the web interface to iDRAC on the database node, as described in *Install and Deploy a Remote Power Management Database*. Alternatively, if the node does not have a hardware RAID controller and the OS system disk size must be changed, customize the `config.xml.in` file. The actual disk size must be greater than or equal to the sum of the `oem-systemsize` and `oem-swapsize`, specified in megabytes.

```

      <oemconfig>
      <oem-systemsize>120000</oem-systemsize>
```

```
      <oem-swapsize>2000</oem-swapsize>
      <oem-swap>true</oem-swap>
```

7. Change the default passwords. Use the `kiwi -createpassword` command to generate a password hash. Do this step twice, for the `root` and the `crayadm` passwords.

```
smw# /usr/sbin/kiwi --createpassword
Feb-26 17:45:14 <1> : Enter Password: <enter root password>
Feb-26 17:45:16 <1> :Reenter Password: <enter root password>
Feb-26 17:45:18 <1> :Your password:
       6KZRSlfiFsFRs
Feb-26 17:45:18 <1> :KIWI exited successfully
smw# /usr/sbin/kiwi --createpassword
Feb-26 17:45:14 <1> : Enter Password: <enter crayadm password>
Feb-26 17:45:16 <1> :Reenter Password: <enter crayadm password>
Feb-26 17:45:18 <1> :Your password:
       mr2qO/OF02b8U
Feb-26 17:45:18 <1> :KIWI exited successfully
```

8. Edit the image configuration file to replace the clear text passwords with the generated password hash and to change the password format attribute from cleartext to encrypted.

```
smw # vi cray-pmdb/config.xml.in
    <users group="root">
        <user password="6KZRSlfiFsFRs" pwdformat="encrypted"
              home="/root" name="root"/>
    </users>
    <users group="crayadm">
        <user password="mr2qO/OF02b8U" pwdformat="encrypted"
              home="/home/crayadm" name="crayadm"/>
    </users>
```

9. If installing the PMDB on a Dell R630 platform that contains an integrated Broadcom BCM5720 four port Ethernet adaptor, edit the `70-persistent-net.rules` file in the image overlay directory to map the `eth0` and `eth1` interface names in accordance with the physical Ethernet cabling guidelines.

   (These rules are copied from the example `udev-rules/70-persistent-net.rules.R630` directory in the RPM.)

```
smw# mkdir -p
smw# mkdir -p cray-pmdb/root/etc/udev/rules.d
smw# cp udev-rules/70-persistent-net.rules.R630 \
cray-pmdb/root/opt/cray/hss/etc/udev/rules.d/70-persistent-net.rules
smw# view cray-pmdb/root/opt/cray/hss/etc/udev/rules.d/70-persistent-net.rules
KERNELS=="0000:01:00.0", SUBSYSTEMS=="pci", DRIVERS=="tg3", NAME="eth0"
KERNELS=="0000:01:00.1", SUBSYSTEMS=="pci", DRIVERS=="tg3", NAME="eth1"
KERNELS=="0000:02:00.0", SUBSYSTEMS=="pci", DRIVERS=="tg3", NAME="eth2"
KERNELS=="0000:02:00.1", SUBSYSTEMS=="pci", DRIVERS=="tg3", NAME="eth3"
```

   IMPORTANT: The database node image has iptables enabled by default. The configuration considers `eth1` the internal HSS management connection and allows TCP connections on any port number. All other interfaces are considered external interfaces. The only port number that is open on external interfaces is for incoming SSH connections.

10. Modify the `opt/cray/hss/etc/erd.broadcast.conf` file to optimize the ERD broadcast settings to limit only enviromental data to flow up into the database node.

    The default configuration enables all traffic that would normally flow into the SMW to flow into the database node as well. This example optimizes traffic so that only environmental data is allowed to flow up into the database node. Command and control events that would otherwise flow downstream are blocked.

```
ec_sedc_data  = up_smw
ec_power_data = up_smw
```

11. Edit the `/root/opt/cray/hss/etc/postgresql/pg_hba.conf` file in the image overlay directory to add additional users.

    By default, PostgreSQL is configured to allow a single user, `pmdbuser`, to connect to database `pmdb` locally from the database node or remotely from the SMW.

12. Use the `git add` command to stage the changed file for commit.

```
smw# git add cray-pmdb/config.xml.in
```

13. Configure passwordless `ssh`. This is needed to allow the SMW to communicate with the PMDB.

14. Create a `.ssh` directory in the `crayadm` home directory relative to the image overlay.

```
smw# mkdir -p cray-pmdb/root/home/crayadm/.ssh
```

15. Append any `ssh` public keys to the `authorized_keys` file. This example appends the ssh public key for the `crayadm` account on the SMW to the `authorized_keys` file in the `crayadm` account on the database node.

```
smw# cat /home/crayadm/.ssh/id_dsa.pub >> cray-pmdb/root/home/crayadm/.ssh/authorized_keys
```

16. Update directory and file permissions. Note that, as part of the image build process `chown crayadm:crayadm` will be applied recursively to the `crayadm` home directory.

```
smw# chmod 700 cray-pmdb/root/home/crayadm/.ssh
smw# chmod 600 cray-pmdb/root/home/crayadm/.ssh/authorized_keys
smw# git add cray-pmdb/root/home/crayadm/.ssh/authorized_keys
```

17. Use the `git status` command to display a list of the files that will be modified with the next commit.

```
smw# git status
On branch pmdb
Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

modified:   cray-pmdb/config.xml.in
new file:   cray-pmdb/root/home/crayadm/.ssh/authorized_keys
```

18. Commit the changes to the customization branch. Specify the commit message interactively, or directly on the command line, as shown in this example.

```
smw# git commit -m "Initial PMDB Customization"
[pmdb cbfb1df] Initial PMDB Customization
2 files changed, 4 insertions(+), 3 deletions(-)
create mode 100644 cray-pmdb/root/home/crayadm/.ssh/authorized_keys
```

19. Build the image using the KIWI imaging tool included in the SLES 12 SDK. This step must be run as root.

    In this example, the installation ISO is placed in the current directory and is named `cray-pmdb.x86_64-2016.02.26.iso`.

```
smw/cray-pmdb-image# time make cray-pmdb-image
...Find build results at: /root/cray-pmdb-image/cray-pmdb-image done
KIWI exited successfully
```

```
Complete logfile at: /root/cray-pmdb-image/cray-pmdb-root.log
mv cray-pmdb-image/cray-pmdb.x86_64-2016.02.26.install.iso .
real 23m43.253s
user 19m14.468s
sys 2m29.396s
```

The custom image is now ready to be installed on the database hardware.

12.3 Install and Deploy a Remote Power Management Database

Prerequisites

Dedicated hardware exists and a new image was configured and built as described in the topics *Configure the SMW to Support a Remote PMDB* and *Configure the Database Node for Remote PMDB Image*. This hardware is referred to in the documentation as the database node.

About this task

This procedure guides through the installation of the image for the database node using the resultant ISO file or burned DVD created in *Configure the Database Node for Remote PMDB Image*. Updates are the same as installations. The image is built such that it will install only over the OS disk.

Procedure

1. Gain access to the console of the machine being installed. This can be done by either physically connecting a keyboard, mouse and monitor to the machine or by connecting through iDRAC. Serial over LAN access is not sufficient.

2. If the node includes an H330 RAID Controller, configue it (using either the BIOS setup or the iDRAC web interface) to create two logical drives.

   The first logical drive is for the system disk. The default XML configuration expects a system disk of 500GB or greater. The system disk should be a `raid1` mirror made up of two physical hard disks.

   The second logical disk should be 1TB or greater, and will be partitioned and named in a later step. Configure this disk according to site policies. A `raid0` stripe of the remaining disks provides maximum storage but least reliability. A mirrored stripe with 4 disks (`raid10`) provides high performance and redundancy against drive failure at the expense of storage capacity. (A `raid5` stripe is not recommended.)

3. Insert the database node image disk into the DVD drive, either physically as a DVD or as a streamed ISO virtual CD through a BMC interface.

4. Power up the database node. If the machine is already started, reboot the machine.

   The database node should now boot from the DVD. It will bring up a boot menu. The options are:

   ```
   Boot from Hard Disk
   Install cray-pmdb
   Failsafe -- install cray-pmdb
   ```

5. From the boot menu, choose **Install cray-pmdb**/

The installation provides the name of the hard disk, or names of partitioned disks, and prompts to continue the install, which will overwrite the disk.

6. Select the disk desired for the database node OS and choose **Yes** to continue.

   The install procedure will write the raw image to the hard disk on the database node and then boot into a login prompt. This completes the installation of the database node OS.

7. Eject the physical DVD or virtual CD/ISO.

8. Log into the database node either through the existing console window or, from the SMW via `ssh`.

   At this point, the login prompt may display a default hostname, such as `databox`. This is because the DHCP has not yet assigned the correct one.

9. Initialize the storage array on the database node to create a single partition covering the entire database disk.

   When the system boots up, the default image looks for an ext4 filesystem containing the label `pg_disk`. If one exists, it is mounted on `var/lib/pgsql` before PostgreSQL starts up. This is a one-time manual procedure to partition, format, and label the disk. Use the GPT partitioning scheme with either `fdisk`, `cfdisk`, or `parted`, then format and label the disk.

   ```
   dh0# mkfs.ext4 /dev/sdb1
   dh0# e2label /dev/sdb1 pg_disk
   ```

10. Reboot the database node to begin using the newly provisioned storage.

    ```
    dh0# reboot
    ```

11. Optional: To verify that the system is actively reporting data to the remote PMDB, query the database for the most recent cabinet-level power reading. This should return a timestamp from within the last 15 seconds.

    ```
    dh0# psql pmdb pmdbuser -c "select max(ts) from pmdb.cc_data \
    where ts >now() - interval '1 minute'"
          max
    -------------------------------
    2016-05-25 18:00:54.228792-05(1 row)
    ```

The external database should now be provisioned and operable.

ATTENTION: Administrators should be aware that although certain HSS commands, such as `xtcli`, `xtbootsys`, and `xtdiscover`, are present on the remote PMDB they are unsupported, unlikely to function as expected, and should not be used.

12.4 About Upgrades and Patches to the PMDB Software

The PMDB node uses an image-based upgrade mechanism wherein the system disk is effectively reformatted at every upgrade. Therefore, any changes made locally to the PMDB node and any log messages messages stored on the system disk (the default) are deleted when an updated PMDB node image is deployed.

CAUTION: During an image-based upgrade, the *PostgreSQL data disk* is not reformatted. However, even though the database configuration settings are preserved, the tables that store telemetry information will have been reinitialized.

SSH host keys are a special case. The PMDB node implements an `init` script such that when the system is shutdown, the sshd host keys are copied to an archive directory on the PostgreSQL data disk. When the system boots, the `init` script looks for any existing sshd host keys in the `/etc/ssh` directory. If the keys do not exist, the previously archived sshd host keys will be restored prior to sshd starting up. If sshd host keys do not exist in either location, then a fresh set of keys will be generated.

When to Upgrade

The PMDB node image should be upgraded and redeployed along side any SMW upgrades or any time a configuration change is required. Keeping the image up to date ensures a rapid recovery in the event of catastrophic hardware failure.

Template Management

Although not required it is strongly recommended to keep the image description under version control. Using a version control tool helps to keep track of configuration changes over time, and aids with merging in new template changes as SMW software upgrades are performed. The instructions provided in *Configure and Create a Remote PMDB Image* on page 46 and *Update the Remote Database Node Software* use `git` for version control.

Security Patches

To incorporate security updates into an updated PDMB image, place the updates in the appropriate update repository on the SMW, then rebuild and redeploy the image. Alternatively, if security updates must be applied immediately, they can be applied directly to a running PMDB node by using the `zypper update` command, as described in the topic *Apply Security Patches to the PMDB*. This uses the same on-line repository exported from the SMW to the Cray Linux Environment via the live updates mechanism.

> **IMPORTANT:** Do not attempt to update the Cray SMW software on the PMDB server by using live updates. This method can be used only for SUSE updates and security patches.

12.4.1 Apply Security Patches to the PMDB

Prerequisites

You must be logged into the PMDB node as `root`.

About this task

There is an immediate need to apply a SUSE security patch to a running standalone PMDB node.

Procedure

1. Verify that the desired repository does not exist on the PMDB by listing the repositories.

```
dh0# zypper lr
Warning: No repositories defined.
Use the 'zypper addrepo' command to add one or more repositories.
```

2. Add the appropriate repositories.

```
dh0# zypper ar http://smw:2526/repos/sle-server_12sp3 sle-server_12sp3
Adding repository 'sle-
server_12sp3' .................................................................................
```

```
[done]
Repository 'sle-server_12sp3' successfully added

URI : http://smw:2526/repos/sle-server_12sp3
Enabled : Yes
GPG Check : Yes
Autorefresh : No
Priority : 99 (default priority)

Repository priorities are without effect. All enabled repositories share the same priority.
```

```
dh0# zypper ar http://smw:2526/repos/sle-server_12sp3_updates sle-server_12sp3_updates
Adding repository 'sle-
server_12sp3_updates' .........................................................................
[done]
Repository 'sle-server_12sp3_updates' successfully added

URI : http://smw:2526/repos/sle-server_12sp3_updates
Enabled : Yes
GPG Check : Yes
Autorefresh : No
Priority : 99 (default priority)

Repository priorities are without effect. All enabled repositories share the same priority.
```

3. Verify that the repositories were added successfully.

```
dh0# zypper lr
# | Alias                   | Name                    | Enabled | GPG Check | Refresh
--+-------------------------+-------------------------+---------+-----------+--------
1 | sle_server_12sp3         | sle_server_12sp3         | Yes     | ( p) Yes  | No
2 | sle_server_12sp3_updates | sle_server_12sp3_updates | Yes     | ( p) Yes  | No
```

4. Refresh the repository to download package metadata preprocess the data for quick reading.

```
dh0# zypper refresh
Retrieving repository 'sle_server_12sp3' metadata .........................[done]
Building repository 'sle_server_12sp3' cache ..............................[done]
Retrieving repository 'sle_server_12sp3_updates' metadata .................[done]
Building repository 'sle_server_12sp3_updates' cache ......................[done]
```

5. Apply the patch or update.

```
dh0# zypper update
Loading repository data...
```

6. Reboot the PMDB node.

```
dh0# shutdown -r now
```

12.5 Update the Remote Database Node Software

Prerequisites

The Power Management Database is running on an external node, rather than on the SMW itself. The PMDB image template is located in a source repository. The examples in this procedure use `git` source control and follow the preparation steps described in *Configure and Create a Remote PMDB Image* on page 46.

About this task

The SMW software has been upgraded, or a configuration change is needed.

Procedure

1. Verify there are no uncommitted changes on the pmdb branch.

```
smw# git status
On branch pmdb
nothing to commit, working directory clean
```

2. Switch back to the master branch.

```
smw# git checkout master
Switched to branch 'master'
```

3. Remove old build artifacts from of the current directory. This will delete the image roots and any installer ISO or disk images. If needed, copy the previous installation ISO or disk image to a different directory.

```
smw# rm -Rf cray-pmdb
```

4. Copy the new image to the current directory.

```
smw# cp -a /usr/share/cray-pmdb-image/* .
```

5. Use the `status` command to show which files have been changed, added, or removed in the new default template.

```
smw# git status
On branch master
        Changes not staged for commit:
          (use "git add <file>..." to update what will be committed)
          (use "git checkout -- <file>..." to discard changes in working directory)

            modified:   cray-pmdb/config.sh
            modified:   cray-pmdb/config.xml.in
            modified:   cray-pmdb/root/usr/lib/systemd/system/xt-xtpmd.service

        Untracked files:
          (use "git add <file>..." to include in what will be committed)

            cray-pmdb/root/usr/lib/systemd/scripts/xt-sshd-key-sync.sh
            cray-pmdb/root/usr/lib/systemd/system/xt-sshd-key-sync.service
```

6. Add all the files to the master branch and verify the changes to be committed.

```
smw# git add *
smw# git status
On branch master
        Changes to be committed:
          (use "git reset HEAD <file>..." to unstage)
            modified:   cray-pmdb/config.sh
            modified:   cray-pmdb/config.xml.in
            new file:   cray-pmdb/root/usr/lib/systemd/scripts/xt-sshd-key-sync.sh
            new file:   cray-pmdb/root/usr/lib/systemd/system/xt-sshd-key-sync.service
            modified:   cray-pmdb/root/usr/lib/systemd/system/xt-xtpmd.service
```

7. Commit the files, adding a commit message in quotes. (Alternatively, create a more detailed commit message in a text editor.)

```
smw# git commit -m "update pmdb image template to release ..."
[master 29650f4] update pmdb image template to release ...
         5 files changed, 131 insertions(+), 3 deletions(-)
         create mode 100755 cray-pmdb/root/usr/lib/systemd/scripts/xt-sshd-key-sync.sh
         create mode 100644 cray-pmdb/root/usr/lib/systemd/system/xt-sshd-key-sync.service
```

8. Merge changes from the updated template into the local template.

a. Switch to the customization branch.

```
smw# git checkout pmdb
Switched to branch 'pmdb'
```

b. View the differences between the customized local template and the new template

```
smw# git diff master
```

The diff should show only site-specific changes.

c. Merge changes from the updated pmdb image template into the local customizations.

In this example the template package has a few modified files and a few new ones, when compared against the previous release.

```
smw# git merge master
Auto-merging cray-pmdb/config.xml.in
      Merge made by the 'recursive' strategy.
       cray-pmdb/config.sh                                            |  13 ++++-
       cray-pmdb/config.xml.in                                        |   4 +-
       cray-pmdb/root/usr/lib/systemd/scripts/xt-sshd-key-sync.sh     | 100 ++++++++++++++++++++++++++++++++++
       cray-pmdb/root/usr/lib/systemd/system/xt-sshd-key-sync.service |  16 ++++++
       cray-pmdb/root/usr/lib/systemd/system/xt-xtpmd.service         |   1 +
       5 files changed, 131 insertions(+), 3 deletions(-)
       create mode 100755 cray-pmdb/root/usr/lib/systemd/scripts/xt-sshd-key-sync.sh
       create mode 100644 cray-pmdb/root/usr/lib/systemd/system/xt-sshd-key-sync.service
```

Merge conflicts should be rare, and must be resolved manually. To verify the site-specific customizations, view another diff to the master branch.

d. Rebuild the updated image.

```
smw# make cray-pmdb-image
```

The updated PMDB image is created and ready to be deployed as described in *Install and Deploy a Remote Power Management Database* on page 50.

13 Cray Advanced Platform Monitoring and Control Utility for Workload Managers

The Cray Advanced Platform Monitoring and Control Utility (CAPMC) provides workload managers (WLMs) and application schedulers with an API for remote power policy execution and monitoring by means of a secure network transport. The utility includes applets for querying system power data, powering off idle nodes, rebooting nodes into the resource pool, setting power caps, and applying power bias factors to individual nodes.

For detailed `capmc` usage information see the `capmc(8)` man page and see *CAPMC API Documentation S-2553* on pubs.cray.com.

Access to the `capmc` utility requires X.509 authorization. The system administrator provides the signed certificate authority, client certificate and private key privacy-enhanced mail (PEM) files.

14 Automatic Power Capping

Accelerated nodes containing a high thermal design power (TDP) processor are automatically capped to the maximum level supported by the power and cooling infrastructure. For example, nodes with 130 Watt CPU + GPU/MIC accelerator are capped at 425W. Rarely will the CPU, memory, and the GPU all be drawing maximum power at the same time. Therefore, it will be rare for the node level power cap to actually engage.

Nodes that contain Intel® Xeon Phi™ processors are capped at 425W and the Xeon Phi processors themselves are capped at 245W. To set a more restrictive power cap on either component see *Manage Power Consumption* on page 8.

If there is a need to disable automatic power capping, please contact Cray Service for guidance.

15 Troubleshooting

Typically any problems that arise in power management can be attributed to changes in the hardware or software, rendering the power profile(s) invalid.

- *Power Descriptors Missing After a Hardware Change*
- *Invalid Profiles After a Software Change*
- *Invalid Power Caps After Repurposing a Compute Module*
- *Local Properties Settings Missing After Software Update*

15.1 Power Descriptors Missing After a Hardware Change

A hardware replacement, such as swapping a blade or upgrading or expanding a system, can affect power profiles that were created for the original components. Cray recommends running the `xtpmaction -a validate all` command after any such changes to the system to verify that there are no missing power descriptors. If the validation process returns an error similar to:

```
ERROR: Profile thresh_75.p3 does not contain descriptor compute|01:000d:306e:
00e6:0014:0040:3a34:0000
```

it means a power descriptor for that component does not exist or is otherwise invalid. Running the `xtdiscover` to identify the hardware components and bounce the system should resolve this problem. It may also be necessary to delete the invalid profile and recreate it.

15.2 Invalid Profiles After a Software Change

When the Cray XC system software was updated or upgraded, the default properties file may have been updated. If so, the install program saved the existing properties file (`properties.local`) to `properties.local.`*`YYYY-MM-DD.HH:MM:SS`* and created a new `properties.local` file. Power management profiles, as well as any other site-specific changes to the original properties.local file, must now be merged into the new file.

Alternatively, run the `xtpmaction -a validate all` command to verify that the contents of the `/opt/cray/hss/default/pm/profiles` directory remain valid. Delete and recreate any profiles that failed validation.

15.3 Invalid Power Caps After Repurposing a Compute Module

Using the `xtcli mark_node` command to repurpose a node from compute to service or vice-versa has the same effect as adding new hardware to a system. In particular, repurposing a compute node or blade to be used as a service node or blade can produce inappropriate power caps on the repurposed module.

Cray recommends running the `xtpmaction validate` action after repurposing a node or nodes as described in *Validate a Power Profile* on page 10. Recreate or update any profiles that fail validation, as described in *Modify a Power Profile* on page 12

If the validation succeeds, or after recreating or updating a failed validation, reactivate the profile as described in *Activate/Deactivate Power Profiles* on page 12. This ensures that the module is capped properly.

15.4 Local Properties Settings Missing After Software Update

When the SMW software is updated, the new software may include an updated properties file. So that the site-specific changes are not lost, if a local properties file exists, that file is renamed `/opt/cray/hss/default/pm/properties.local.`*`yyyy-mm-dd.hh:mm:ss`*.

To recover the site-specific settings, copy the information from the renamed file to the newly created `properties.local` file at `/opt/cray/hss/default/pm`.