SAND2005-2987C

Red Storm Capability Computing Queuing Policy

by
James A. Ang, Robert A. Ballance, Lee Ann Fisk,
Jeanette R. Johnston, and Kevin T. Pedretti
Sandia National Laboratories*

Presented at:
Cray Users Group 2005
Albuquerque, NM
May 17, 2005

* Sandia is a multi-program laboratory operated by Sandia Corporation, a Lockheed Martin Company for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Presentation Outline

- ASC Policy Directive for Capability Systems
- Configuration of Red/Black Sections: small, large, jumbo
- Red Storm Resource Allocation Parameters:
  - Determination of node-hour division of Red Storm between classified and unclassified computing
  - Determination of node-hour allocations
    - Red Section among the Tri-Labs and
    - Black Section among Tri-Labs and ASC Alliance Centers
- PBS Queuing Policy
  - Fair Share Scheduling Algorithm
  - Definition of Four Queue Structure
  - Backfill
- Concluding comments
  - Capability Utilization vs. Utilization: Alternate Definition for Capability Usage Performance metric
  - Improvements to the ASC Capability Utilization Policy
  - Caution: Unintended Consequences
  - Next steps

ASC Policy Directive for Capability Systems

- 80% (or more) of the node-hours of utilization must be allocated to jobs that run on 40% or more of the system

$$\mathrm{CUP}_{40\%} \geq 80\%$$

- The CUP metric is a ratio of utilizations:

$$\mathrm{CUP}_{40\%} = \frac{\text{node-hour}_{\text{Large Jobs}}}{\text{node-hour}_{\text{Large Jobs}} + \text{node-hour}_{\text{Small Jobs}}}$$

Where "Large Jobs" by definition use 40% or more of the system and "Small Jobs" by definition use less than 40% of the system

- The CUP metric will be measured and tracked over long periods, e.g., one year
- This is a very challenging goal and may be ill-posed …

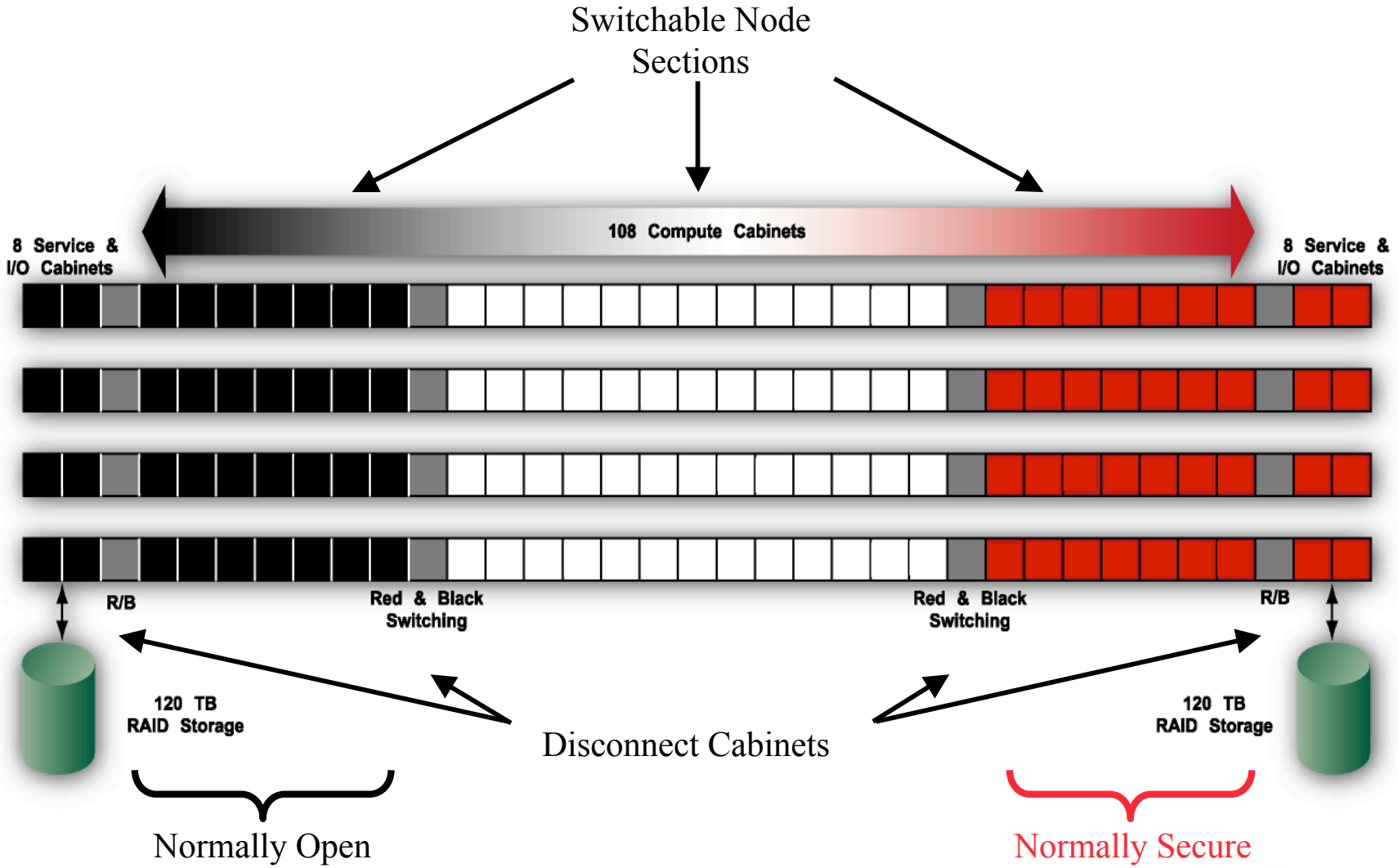
National Nuclear Security Administration

Sandia National Laboratories

Capability Policy Implementation

- Sandia is the first ASC Lab to implement the new ASC Capability policy – this is not likely to be the final word
- Plan to implement via a *Large* queue with a min job size of 40% of the available nodes and give the *Large* queue priority over all jobs in the *Standard* queue with a max job size of 40% of the available nodes
- Shift from Institutional Allocation of nodes, to Allocation of node-hours
  - Need a fair share queuing policy to handle inter-lab job priority
  - Need to understand how node-hours allocations are treated with Red-classified/Black-unclassified system configuration switches

Red Storm Layout with Switch Cabinets



National Nuclear Security Administration

Sandia National Laboratories

Red Storm
System Configurations

Section Name	Red Section		Black Section		Section Name
	40% Goal Nodes	Compute Nodes	Compute Nodes	40% Goal Nodes	
–	0	0	10,368	4,147	Jumbo
Small	1,075	2,688	7,680	3,072	Large
Large	3,072	7,680	2,688	1,075	Small
Jumbo	4,147	10,368	0	0	–

National Nuclear Security Administration

Sandia National Laboratories

Red Storm Resource Allocation Parameters

- **The following global *Red Storm* shares are set by agreement of the ASC executives:**
  - $S_{sh} = 0.45$ — Host lab has 45%
  - $LA_{sh} = 0.225$ — Sister labs have 22.5%
  - $LL_{sh} = 0.225$ — Sister labs have 22.5%
  - $AC_{sh} = 0.10$ — Alliance Centers* have 10%
- **Each Lab is Free to choose what fraction of their allocation they want for classified vs. unclassified computing:**
  - $S_R = 0.75$ — Assume for example, some typical Lab allocations
  - $LA_R = 0.85$
  - $LL_R = 0.90$
  - $AC_R = 0$

* Note, the five ASC Level 1 Alliance Centers; Univ. of Illinois, Univ. of Chicago, Univ. of Utah, Stanford, and Caltech are each entitled to an equal share of the Alliance Center global shares. In the following discussion the Alliance Centers are treated as a single institution, but they are really five distinct universities so the AC share is further divided by five.

National Nuclear Security Administration

Sandia National Laboratories

Red Storm Resource Allocation Parameters (cont.)

- The fraction of node-hours the system should be Red or Black is given by:

$$f_R = LA_R \times LA_{sh} + LL_R \times LL_{sh} + S_R \times S_{sh} = 73.1\%$$
$$f_B = 1 - f_R = 26.9\%$$

- For the Red Section the following formulas define the node-hour allocations among the three labs:

$$f_{LA,R} = [LA_R \times LA_{sh}] / f_R = 26.1\%$$
$$f_{LL,R} = [LL_R \times LL_{sh}] / f_R = 27.7\%$$
$$f_{S,R} = [S_R \times S_{sh}] / f_R = 46.2\%$$

- For the Black Section, the following formulas define the node-hour allocations among the three labs and the Alliance Centers:

$$f_{AC,B} = AC_{sh} / (1 - f_R) = 37.2\%$$
$$f_{LA,B} = [(1 - LA_R) \times LA_{sh}] / (1 - f_R) = 12.6\%$$
$$f_{LL,B} = [(1 - LL_R) \times LL_{sh}] / (1 - f_R) = 8.4\%$$
$$f_{S,B} = [(1 - S_R) \times S_{sh}] / (1 - f_R) = 41.8\%$$

National Nuclear Security Administration

Sandia National Laboratories

PBS Fair Share Algorithm

- When a user runs a job, the userID's node-hours are accumulated in a usage file. Usage for his or her institutionID is also incremented.
- The Fair Share Algorithm gives priority to userID and institutionID that have had lower amounts of usage. Since large amounts of usage should not be a permanent bias, the node-hour usage totals are *decayed* over time.
- The standard way to do this in PBS is to define a configuration parameter indicating the *half-life* of used node-hours. When this half-life, perhaps three to fourteen days or longer, has elapsed, PBS will divide all usage totals in half.
- On our Cplant System with OpenPBS we replaced this crude half life implementation with a more frequent and accurate half-life decay calculation

Establish Four PBS Queues

- ***Standard*** **– All users may submit jobs to the *Standard* queue. Jobs submitted to the standard queue must use less than 40% of the available nodes.**
  - ***Express*** **– Access to this queue is controlled by the Red Storm system management team. Jobs submitted to this queue must also use less than 40% of the available nodes. All jobs in the *Express* queue will be considered for execution by the PBS scheduler before any job in the *Standard* queue. The express queue exists to fast-track ordinary jobs under exceptional circumstances.**
- ***Large*** **– Any user may submit jobs to the *Large* queue. Jobs submitted to the *Large* queue must use at least 40% of the section's nodes. All jobs in the *Large* queue will be considered for execution before any job in the *Express* queue. *Gaming* the system by requesting large numbers of nodes for jobs that do not require large numbers of nodes will be considered a very serious offense.**
  - ***Expedited*** **– This queue is for any job which is deemed to be extremely urgent. Admission to the priority queue is granted by the Expedited Priority Run committee. Jobs submitted to this queue are not subject to any job size or job duration limits. All jobs in the *Expedited* queue will be considered for execution before any job in the *Large* queue.**

National Nuclear Security Administration

Sandia National Laboratories

Queue Priority Order and Backfill

- PBS will evaluate the queues in queue priority order. When evaluating the jobs in a queue against one another, PBS will use the fair share algorithm described above.
- The PBS scheduler for *Red Storm* has been modified to treat the highest priority job for which there are insufficient nodes as a *starving* job.
  - The scheduler will *Backfill* while waiting for the highest priority job to begin
  - No other job in the queues will be placed into execution unless it will complete before the starving job can run, or unless it will use nodes that the starving job will not use when it begins to run

Capability Utilization vs. Utilization

- Backfill illustrates an example where our new capability computing policy directive can directly conflict with another common System-level performance goal – utilization rates should be maintained at or above 80%
- Backfill Increases Overall Utilization Rates
- However, since an inherent characteristic of backfill jobs is that they are small, Backfill Decreases $CUP_{40\%}$
  - Backfill jobs have to be small to fill in holes between currently running jobs

Suggest a modified $CUP_{40\%}$ Metric

- The $CUP_{40\%}$ metric does not distinguish between "normal" small jobs and Backfilled jobs. They are all part of the $\text{node-hour}_{\text{Small Jobs}}$

$$CUP_{40\%} = \frac{\text{node-hour}_{\text{Large Jobs}}}{\text{node-hour}_{\text{Large Jobs}} + \text{node-hour}_{\text{Small Jobs}}}$$

- One suggestion to remove the conflict between these two system performance metrics is to modify the $CUP_{40\%}$ metric to remove the penalty for Backfilled jobs, e.g.

$$\overline{CUP}_{40\%} = \frac{\text{node-hour}_{\text{Large Jobs}}}{\text{node-hour}_{\text{Large Jobs}} + \text{node-hour}_{\text{Small Jobs}} - \text{node-hour}_{\text{Backfill Jobs}}}$$

- This would require additional accounting capabilities to track node-hours used by backfill jobs

National Nuclear Security Administration

Sandia National Laboratories

Why The New ASC Capability Policy may be Ill-Posed

- Assume a user/group has a long-running Capability job that uses 70% of the system
  - If this job runs for an entire year and we have "reasonable" backfill, it is impossible to meet the $CUP_{40\%} \geq 80\%$ goal
  - The Utilization Rate of the remaining 30% of the system would have to fall to 70% to meet the $CUP_{40\%} \geq 80\%$ goal
- As defined, the $CUP_{40\%} \geq 80\%$ goal creates a "Death Valley" whenever a Capability Job uses 61-79% of the system
  - Below 61% it is possible to run a second Capability job
  - At 80% and above the CUP would be automatically met
- The Goal may be too stringent, perhaps we will find it needs to be reset to a modified goal of: $CUP_{40\%} \geq 60\%$
- These suggestions to modify the CUP metric will be the subject of further discussion with the ASC Executives

Unintended Consequences

Our user community is reasonably intelligent, and as is typically the case, while they are waiting in the queue for their jobs to start, they usually have time to think of ways to game the queuing policy

- **In their efforts to raise job priority, users move away from sizing jobs based on the amount of memory they need to contain their problem.**
- **In another effort to raise job priority, users may force applications to run at scales where they have poor parallel efficiency. This could be worse than using a capability system to run capacity jobs.**
- **Another factor is that some users may simply bundle 10, 100, or more small jobs into one big job that unfolds to run on 40% or more of the system.**
- **Allow nodes to idle, i.e., run while (1)**

Concluding Comments

- This is a work in progress
- We have neither implemented nor tested the proposed four level queuing structure, but it should be straightforward
- Once we have test data we may find that we need a more sophisticated implementation that factors Job size/Section size directly into the determination of fair share priority, e.g.,
  - If $CUP_{40\%} < 80\%$, then use
    $P_{CUP} = [0.80 - CUP_{40\%}] \times [JobSize/(0.40 \times Section\ Size)]^{p} \times$ [Standard PBS Fair share factor],
    where P_{CUP} is the capability metric-weighted fair share job priority, p is a real value number to vary the power law sensitivity of the capability job size weighting.
- At this time we have identified the following needed capabilities
  - Provide a nightly determination of the $CUP_{40\%}$ metric
  - Track usage by *Backfill* jobs
  - Correct the fair share calculation to refund "usage" tracked by jobs that actually failed due to node hangs