SAND2015-5981C

New Systems, New Behaviors, New Patterns: Monitoring Insights from System Standup

J. Brandt and A. Gentile
Sandia National Laboratories
Albuquerque, NM.
Email: (brandt|gentile)@sandia.gov

C. Martin
Los Alamos National Laboratory
Los Alamos, NM.
Email: c_martin@lanl.gov

J. Repik
Cray, Inc.
Email: jjrepik@cray.com

N. Taerat
Open Grid Computing
Austin, TX.
Email: narate@ogc.us

***Abstract*—Disentangling significant and important log messages from those that are routine and unimportant can be a difficult task. Further, on a new system, understanding correlations between significant and possibly new types of messages and conditions that cause them can require significant effort and time. The initial standup of a machine can provide opportunities for investigating the parameter space of events and operations and thus for gaining insight into the events of interest. In particular, failure inducement and investigation of corner case conditions can provide knowledge of system behavior for significant issues that will enable easier diagnosis and mitigation of such issues for when they may actually occur during the platform lifetime.**

In this work, we describe the testing process and monitoring results from a testbed system in preparation for the ACES Trinity system. We describe how events in the initial standup including changes in configuration and software and corner case testing has provided insights that can inform future monitoring and operating conditions, both of our test systems and the eventual large-scale Trinity system.

I. Introduction

A primary source of information available to system administrators for troubleshooting problems in High Performance Computer (HPC) systems is the aggregate set of log files. These range from messages of interest from a security perspective, to warnings of sensor thresholds crossed, to error and failure messages for both hardware and software. On a large system there may be hundreds of millions of such log entries per day with the majority being benign. Over time, a system administrator will learn the ramifications of new features of the machine and which log entries are meaningful and what vendor specific tools can be utilized to expose problems and causes. However, each new system typically has enough differences to make initial troubleshooting and root cause analysis a challenge and new problem indicators may be discovered over a machines lifetime. Such differences stem from causes including but not limited to different and/or upgraded Operating Systems (OS), compilers, applications, hardware, file systems, network infrastructure, etc. Different developers encode diagnostic messages differently. Vendor specific hardware related messages will necessarily be different due to different device names and vendor terminology.

While new systems present challenges with respect to understanding behavioral and diagnostic characteristics, the standup and acceptance process provides unique opportunities to enable necessary insights. This testing phase is particularly dynamic with respect to software installs, configuration, hardware replacement, stress testing, and corner case investigation. Appropriate investigation and analysis in this controlled environment can provide insights into behaviors and issues, and into what log information is useful for warning and diagnosis in the future.

In this paper, we present insights gained during the acceptance testing and standup for one of the advance test platforms (Mutrino) of the upcoming New Mexico Alliance for Computing at Extreme Scale (ACES) Trinity Cray XC40 platform. While specific results address the new functionalities and behaviors of this platform, we more generally target the issues of new features in hardware, software, and HPC subsystems and how those manifest themselves in higher level concepts in thermal, electrical, and performance areas.

The rest of this paper is arranged as follows. In Section II we describe more about new features of our system and how they are anticipated to impact funcationalities of interest. In Section III we describe the testing during our standup process. Machine layout is described in V. In Section IV we describe complexities of the log data sources on the Cray XC platforms and how we utilize our tools for analyzing such data. In Section VI we present insights gained from the standup and acceptance process and how they have enhanced our understanding and informed future monitoring. Finally, in Section VII we summarize and discuss future work.

II. Reference System Description

While the concepts presented in this paper are generally applicable to new HPC systems, we exemplify their applicability by presenting our own *new system* scenario. This section describes the new system we will be using as a reference in this work, with particular emphasis on the differences between this and our previous system that on the surface might be viewed as insignificant. These differences result in new and different failure mechanisms as well as give rise to new log messages and behaviors that can be utilized to detect and understand system problems. The following five subsections each present a differentiating feature as well as some of the characteristics that may change due to that feature. The remainder of the paper addresses how these features are tested for acceptance and how this testing exposes associated problem indicators that can then be used throughout the system lifetime to quickly

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

flag such problems during normal operation. In particular Section VI presents use cases of how these differences impact our monitoring of thermal, network, and electrical characteristics using logs from the various subsystems. Note that each of the differences described below may drive new log patterns while previous patterns may or may not disappear.

A. New Processors

Our current HPC capability system, Cielo, is a ten thousand node Cray XE6 with dual 8 core AMD Interlagos processors per node. The initial deployment of our new system, Trinity, is a ten thousand node Cray XC40 with dual 16 core Intel Haswell processors. While we have other clusters with Intel processors, this is the first instance we have of Haswell. The Intel Haswell processor is significantly different from both the AMD and previous Intel processors with respect to its thermal and thermal throttling characteristics (Section VI-A3) and its power capping capabilities (Section VI-C1). With these new characteristics, come a whole new set of log messages to assist in diagnosis. Use of new feature related messages is explored in each of the aforementioned sections. *Characteristics affected: performance, thermal, and electrical due to packaging and new features (e.g., power capping, p-state setting)*

B. New Interconnect Features

While Cielo utilized the Cray Gemini interconnect configured in a 3D torus, Trinity's interconnect uses the Cray Aries interconnect configured in a Dragonfly [1], [2] configuration. While some of the features of these two interconnect technologies are similar, there are large differences in message logging, performance counters, error handling, adaptive routing, and hot swapability. Additionally the network topology being so different will necessitate new tool development for understanding congestion related performance changes. With respect to the interconnect, this paper focuses on the Aries related log messages experienced thus far in our testing and how they relate to possible problems in the system (Section VI-B1). *Characteristics affected: performance*

C. New Packaging

Packaging can drive different observed behaviors. As the HPC market has matured we have progressed from using standard desktop boxes on shelves in racks to horizontal 1U servers in racks to blades in chassis in racks. The trend is toward higher and higher densities. With these high densities packaging and resultant airflow become important concerns. In Cielo a rack consisted of three chassis of vertical blades all cooled by bottom to top airflow. This resulted in increasing temperatures for nodes as their rack height increased. Trinity, by comparison, consists of racks of blades placed horizontally in the rack with transverse airflow. The layout is described in Section V. Blades are half rack width so there is still a temperature differential between the intake air of left and right hand compute node blades (Section VI-A3). *Characteristics affected: thermal and performance due to CPU throttling*

D. Water Cooling

Our Cielo system utilized an underfloor cold air plenum to supply cooling to the compute nodes and thus as long as long as the chiller plant was supplying cold air the nodes were cooled. With Trinity, the cold air is supplied via cooling coils attached to the side of each rack which contain active water valving and a feedback loop to maintain a setpoint temperature. This arrangement means a whole new layer of monitoring and control is needed. Thus additional information with respect to water pressures, flows, and temperatures as well as fan speeds and resulting air temperatures is present on Trinity that was not on Cielo. Additionally this information must be analyzed in the context of appropriate chiller plant information to ensure correct interaction and operation of the system as a whole. Water data is reported via SEDC [3] and includes valve openings, water pressure, and water temperature, in addition to air related data (temperatures, fanspeeds, etc). We consider the interaction of site facilities and machine and machine responses in Section VI-A1. *Characteristics affected: thermal and performance due to CPU throttling*

E. System Software

System software and firmware can undergo significant changes as vendors fix bugs and add features. These changes can thus change a system's behavioral characteristics including diagnostic messages. Section VI-C1 and Section VI-A2 present some observed changes in the behavior of Mutrino as it was transitioned to a new version of system software. Such changes accentuate the desirability of re-running the same battery of acceptance tests, including corner cases, to understand the new behavioral characteristics and associated indicators following significant upgrades. In general this testing can be accomplished on a representative test-and-development system such as Mutrino. *Characteristics affected: thermal, electrical and performance due to how cooling and power capping are handled*

III. Testing

This section presents a high level description of the set of tests used to characterize system behavior as well as identify weaknesses and failure modes. In particular we were interested in application and filesystem functionality and performance as well as the system's thermal and electrical properties under various workload scenarios.

A primary reason for building ever larger HPC systems is increased aggregate performance. Thus our application performance testing includes representative applications and we look for acceptable runtime performance. Additionally we exercise new features in the Aries network (warm swap) and Haswell processor (limiting power draw of an application through runtime modification of clock frequency). These experiments are designed to ensure functionality and enable understanding of the limitations of these capabilities as well as our visibility into how well they are working.

Understanding thermal characteristics of the system is crucial to efficient operation and overall application performance. Thus we identified log entries indicative of thermal problems that show up under both high computational load and during simulated failures. Sections VI-A3 and VI-A1 show how these logs can be used to identify when there are problems. Further such correlations, established during testing, can be utilized for quick root cause analysis during normal operations.

The two primary electrical related behaviors we present here are related to power capping and simulated power failures. Power capping behavior is becoming increasingly important as systems grow in size and the system power draw becomes a significant fraction of a site's aggregate power draw. This can have consequences with respect to cost and stability. Thus it is important to understand how well power can be controlled, its variation across components, and its impact on application performance. Power failures local to a set of nodes or system wide can occur without warning for a variety of reasons (e.g., power supply failure, utility power outage, lightning strike). Knowing how this will affect a system and how to properly bring it back online is the motivation for the electrical failure testing described in Section VI-A1.

IV. Log Data

Examination of log data is a standard approach for gaining information about system errors, states, and behaviors. Log data contains a variety of information including: standard OS related information (e.g., user logins, scheduler events, OOM messages), vendor/hardware specific information (e.g., network technology specific events), and occurrence data. The latter may include items that are clearly errors, notifications where the ramifications may not be readily apparent (e.g., CPU throttle, Processor Hot), and warnings that are clearly about problem conditions but without indication of the severity of the condition which induced them. Messages about components (e.g., mezzanine), may require knowledge of the architecture, which is not universal across platforms. Further, documentation covering the full scope and ramifications of these messages generally doesn't exist. As a result, discovery and interpretation of events of significance within logs can be difficult.

A. Cray Log Data

On Cray XE/XK/XC compute platforms, most log data resides on their System Management Workstation (smw). While this provides a single point of access to the data, the log file organizations and formats are not conducive to direct consumption by general analysis tools. There are at least 20 different event log files in directories and subdirectories. While most of these are text, some logs are in binary format and vendor specific utilities are required to parse them (e.g., hwerrlog file and the xthwerrlog utility). Many log messages are in multi-line format but require examination as a unit for understanding. Some text logs are not in rsyslog format (e.g., *netwatch*) while many log analysis tools depend on that format. While logs are typically divided by source and/or topic, we have found messages on certain topics in unexpected places (example in Section VI-C1).

In addition, Cray's System Environment Data Collections (SEDC) [3] consist of 10's of more logs of numerical component (e.g., CPU, DIMM) related characteristics such as temperatures, voltages, power, and cooling data. They also include environment related information including fan speeds, air velocities, water temperatures, valve openings etc.

An additional complexity to log analysis is that the smw controls a number of functions of the system and hence its access is restricted and it is not intended to support computationally intensive analysis as this could interfere with its control functions. Cray has implemented their Lightweight Log Manager (LLM) [4] functionality, which is a wrapper around rsyslog supporting a simple configuration command to forward Cray logs from the smw to a site log host. However, we have found that LLM does not result in an identical setup of the logs on the remote host (e.g., logs handled via specific network ports on the smw, like the *controller* logs, are not forwarded to those same ports on the site log host). Thus log analysis tools targeting specific file sources on the smw may have to be altered on the log host. In addition, LLM does not by default forward the SEDC data. Finally, because the Cray infrastructure reinstalls or rewrites some of the configuration, Cray does not recommend direct alteration of rsyslog configuration files. This makes additional user tuning of rsyslog on the smw (e.g., directing certain logs to specific ports only) more difficult and potentially not possible.

B. Log Data Pattern Extraction for Analysis

Effective log data analysis requires not only identification of events of interest but also making correlations and associations between events. These tasks become harder with increases in data volume and number of disparate log file types. The problem is further exacerbated by the existence of multiple line log messages. Unraveling the correlative relationships can be particularly challenging for new systems or new configuration scenarios where the scope and meaning of possible messages may be unknown. As a reference point, over the time frame of the data of this paper (< 100 days) our single cabinet Cray XC40 system ($2/3$ populated) produced over 120 million lines of text log data. The implications of this with respect to Trinity's final size of 200 cabinets is the possibility of over 2 billion log lines per day to wade through. This doesn't include the scheduler logs nor the ones in binary format. The number of lines per file type per day are shown in Figure 1. A shorthand has been used to designate files that reside under the `controller` and dated `p0` subdirectories of the main log directory.

In this work, we are primarily considering text log data. There are a number of open source tools for assisting with log file content discovery (e.g., Baler [5], HELO [6], Splunk [7]). While Splunk is widely used for analyzing log data, it requires the user to a-priori know their patterns of interest in order to write filters to analyze their occurrence patterns. A problem with this is that many such patterns are not known at system standup and are discovered over time as problems emerge. Both Baler and HELO perform deterministic discovery of patterns in log data allowing the user to identify those that bear further investigation as soon as the first occurrence. Baler uses identified word lists (e.g., English words, additional words of interest, e.g., `Cray`, `i2c`) from which to build the patterns. We utilize Baler in this work and show how this type of tool, in conjunction with further analysis (e.g., visual), can be used to provide insight into the behavioral characteristics of a new system where new log message types and patterns are still being generated and are not known a-priori. In order to obviate the issue of what types of logs reside in which files we stream all logs to Baler, from the smw, regardless of source.

Baler reduced our log files to a few thousand patterns which can be further reduced using similarity heuristics. We would expect about the same number of patterns for the full scale

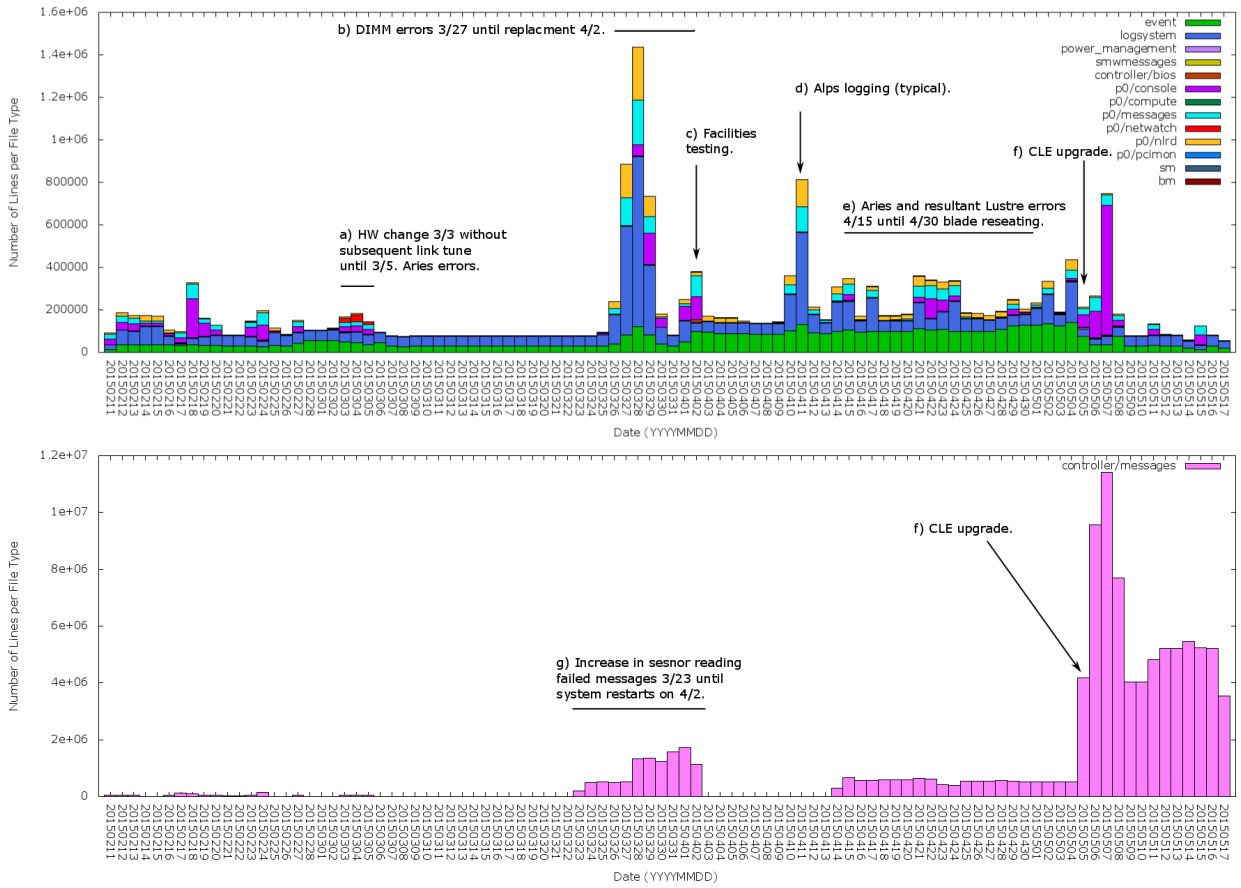


Fig. 1. Number of log lines per file type per day, for all syslog-formatted log files. Most files (top), *controller/messages* file (bottom). The latter is the major contributor to the number of log lines after the CLE upgrade. Occurrences of events of interest are marked and discussed in the text. Multi-day events are indicated by a line through the duration of the event.

Trinity system even with an aggregate daily volume in the billions. The pattern companion figure of Figure 1 (bottom) is Figure 2, showing the patterns for the same log source. There is an initial discovery of new patterns at system startup (h) which drops rapidly. New events and new software result in new log lines and thus new patterns. In particular, installation of a new version of the Cray Linux Environment (CLE 5.2 UP03) (f) resulted in millions of log lines but not a commensurate number of new patterns. Note that even when log lines contain understandable words that bear investigation, they may be difficult to find if one doesn't know of their existence (see Figure 3). Data reduction and testing under known conditions is intended to help discovery of patterns of interest.

V. Machine Layout

This section describes the machine layout and Cray naming conventions used in the rest of this paper. An illustration of the Mutrino physical layout is shown in Figure 7 (top). It consists of one *rack* named `c0-0`. Each rack has 3 *chassis* which are vertically stacked and are named `c0-0c0` through `c0-0c2`. Each chassis consists of 16 *blades*, 8 on the left and 8 on the right. Blades reside in *slots*. The slots/blades in, for example, chassis 2 are named `c0-0c2s0` through `c0-0c2s7` on the left, and `c0-0c2s8` through `c0-0c2s15` on the right, bottom to top. Our system is not fully populated. Slots without blades are populated with diffusers and are shown as X's.

Blades are either *compute* or *service*. Compute blades have 4 nodes; service blades have 2. Service blades are in the left of each chassis and, in our layout, are in each of the first three slots, e.g., `c0-0c0s0` through `c0-0c0s2` and similarly for each of the three chassis. The nodes in service blade `c0-0c2s0` are `c0-0c2s0n1` and `c0-0c2s0n2`. The nodes in compute blade `c0-0c2s12` are `c0-0c2s12n0` through `c0-0c2s12n3`, however on the blade the nodes are laid out in the order `(2,1,3,0)`, front to back. The back of each blade has an Aries router chip; Aries related components have an `a` with subsequent identifiers after the blade name (e.g., `c0-0c2s12a0l16`). Compute nodes have 2 processors and service nodes have 1; these are shown as dots in the figure. The figure is not to scale.

VI. Use Cases: New Behaviors, New Patterns

This section presents use cases of how the testing and maintenance during system standup to investigate new features[1] has improved understanding of our reference system and

[1]For completeness features (b) and (d) in Figure 1, which are typical, are explained here. Feature (b): the increased verbosity in the *console* log is due to DIMM errors. The DIMM had to be replaced and was done so as part of the system shut down in the facilities testing on 4/2. The *logsystem*, *p0/nlrd* and *p0/messages* log behavior is as that of feature (d). This is typical of when there is an increase in the runs and hence the alps logging. There is alps-related logging in those files.

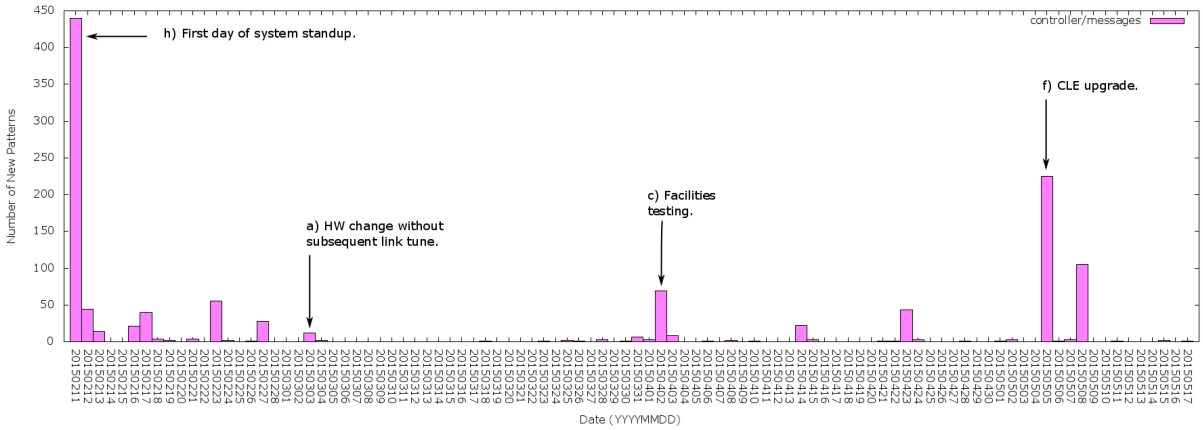


Fig. 2. Number of new patterns per day for *controller/messages* only. Patterns are expected to be stable through time until new events occur. CLE upgrade resulted in the discovery of new patterns, however, these are a small fraction of the number of new lines.

```
* *-*-* *:*:*|*|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[alert|Module Health Fault] \\
      |Cause:*|Text:*|ALERT|NULL|NULL|* is not Good!|NULL|NULL

kernel - - -  Oops: * [#*]

cray-*: **WARNING**  /*/security/limits.* * require hard and soft file limits
cray-*:              greater or equal to * for proper * operation
cray-*:              when large number of jobs are launched concurrently.

*: server *.*.*.* not responding, still trying

* database not setup
Perform the following steps by hand:

* *-*-* *:*:* *-* * *: adding component  to critical node fault list

*: Top * applications sorted by aggregate throttled ejection bandwidth
* *-*-* *:*:* *-* * *: Top * nodes involved with network congestion
* *-*-* *:*:* *-* * Congestion candidate node *: *-* (* flits/*) (* *; no * list)

* * of * blades reported timeouts during routing.
* You * need to reboot these blade controllers, or * the blades out,
* before trying again.

kernel - - -  FAT-* (*): bogus number of reserved sectors

kernel - *-* - Cable is unplugged...

WARNING: Your /*/* does not contain the * *
field.  * will kludge around things for you, but you
         should fix your /*/* file as soon as you can.

Are the lustre modules loaded?

*-*-*:*:* BUG: sleeping function called from invalid context at arch/*/*/fault.*:*

* setpoint was changed to *(*) to move down the dew point
```

Fig. 3. Example patterns possibly worthy of investigation, if one knew of their existence. Variable, non-word data is indicated by *. Long patterns are broken in figures where indicated by the double slash.

how that knowledge can impact our monitoring of thermal, network, and electrical characteristics using logs from the various subsystems.

A. Thermal

1) Facilities Testing: HPC systems today require extensive facility cooling and power infrastructures. In order for HPC data centers to be efficient and effective, the management and monitoring must include not only the personnel who manage the platform, but also the facilities operations personnel. These individuals do not have direct access to the platform and in general are unaccustomed to sorting and searching log data. In this environment tools that can analyze data from multiple sources quickly and inform operations personnel of problems, when they arise, and the most probable causes is essential. Discovery and use of appropriate log patterns for use in dashboards utilizing filter-based tools (e.g., [7]) can thus provide the runtime feedback needed for timely mitigation of problems as they occur.

There were several tests performed on Mutrino that required analysis and visualization for confirmation of correct platform and facility operations. The areas of interest for operations included function of system during power loss, power factor fluctuations, and function of platform with a loss of cooling components. Note that the fan failure test resulted in unexpected system behavior (i.e., water valve opening) illustrating the value of corner case tests. These tests and their results are described in Table I.

In addition to determining how the machine would respond to the simulated failures, we further wanted to understand how the simulations would manifest themselves in the log files in order to facilitate detection of issues if they were to occur naturally. The tests resulted in feature (c) in Figure 1 and Figure 2, the latter indicating that new patterns occurred because of the new event. These patterns are shown in Figure 4. They refer to messages such as rectifier voltages too low, blower temps too low, air velocity too low, etc. These messages also gave us insight into the operating parameters of the machine. In addition they report suboptimal values for fans and rectifiers functioning and that the machine shutdown was in response to the fan/rectifier situation. Note that some of these cases would also result in changes in the SEDC numerical data (e.g., valve opening) which we can additionally include in monitoring.

2) DIMM Hot: DIMM Hot messages are of interest as high temperatures may result in memory errors. DIMM Hot messages occur in our logs only during boot or startup and only began after the CLE upgrade (Figure 1 (f)). In comparison, Processor Hot messages (Section VI-A3) are reported over other times. The investigated correlation with the booting/startup events makes it quite possible that this is not a problem event; however, it is not clear if this is a new behavior or just a new message. As in the induced facilities testing (Section VI-A1), system changes result in new patterns, so relying on user input for filter definition for items of interest discovery isn't an optimal monitoring approach.

Relevant patterns are shown in Figure 5 with association

TABLE I. FACILITIES TESTS PERFORMED

Test Description	Test Motivation	Results
Simulate power loss a) at boot time, b) at idle, and c) with machine at peak capacity. This was done by shunt tripping the breaker that supplies power to the system.	We expect to have power loss due to environmental factors such as lightning strikes. These are very common during the summer months in New Mexico.	Platform shut down as expected with no hardware fallout.
Removed 12 rectifiers from platform.	The power factor of the system fluctuates. This test was used to determine the impact to the power factor with fewer rectifiers.	There was an increase in the power factor when these rectifiers were removed.
Simulate one fans failure by shunt tripping the breaker providing power to the fan.	There was no sequence of operations documentation and the expectation was the other fans would speed up to compensate. This test was to validate that happened.	The fans did not increase in speed; however, the chilled water valve opened up to compensate. This led to questioning Cray on sequence of operations and they were able to validate this was the correct response to a failed fan.
Simulate two fans failing by shunt tripping the breaker providing power to the fan.	This test was to validate the cabinet platform would power down as expected due to multiple fan failures.	The system responded as expected.

```
* *-*-* *:*:*|ec_sedc_warning|src:::*:*|pri:*|seqnum:*|svc:::*-* *: *.* [below min] <min:*.* max:*.*> units: * | *: *.* [below min] <min:*.* max:*.*> units: * |
* *-*-* *:*:*|ec_sedc_warning|src:::*:*|pri:*|seqnum:*|svc:::*-* *: * [above max] <min:*.* max:*.*> units: * |
* * - -  micro log: '*|ALERT|FAN|NULL|Cannot detect blowers running, disabling rectifiers!|NULL|NULL'
* * - -  micro log: '*|WARN|FAN|NULL|Number of running fans in blower cabinet is too low! Expected fans:*, Actual running:*|NULL|NULL'
* * - -  micro log: '*|ALERT|NULL|NULL|Blower * not responding!|NULL|NULL'
* * - -  * * * *:*:*.* *:*:spread-*.*:*:*:WARNING:*: number of blowers running * is below the number of blowers to run *
* * - -  micro log: '*|*|Cannot detect blower fan *'
* * - -  * * * *:*:*.* *:*:spread-*.*:*:*:INFO:*: shutdown the cabinet because not enough number of blowers running
* * - -  * May * *:*:*.* *:*:*.*:*:*:INFO:Blower speed hardware value *(%) does not match the software value. set the speed to *(%)
* * - -  * May * *:*:*.* *:*:*.*:*:*:WARNING:Cannot get blower present state
* *-*-* *:*:*|*|src:::*-*|pri:*|seqnum:*|svc:::*-*|*|INFO|Rectifier not responding in slot:* on shelf:*.
* * - -  micro log: '*|ALERT|NULL|NULL|Insufficient rectifiers present:* Required:*!|NULL|NULL'
* *-*-* *:*:*|ec_ll_failed|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[alert|Cabinet Sensor Check Warning]|Cause:*|Text:*|ALERT|FAN|NULL| \\
    Cannot detect blowers running, disabling rectifiers!|NULL|NULL.
* *-*-* *:*:*|ec_ll_failed|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[alert|Cabinet Sensor Check Warning]|Cause:*|Text:*|ALERT|NULL|NULL| \\
    Insufficient rectifiers present:* Required:*!|NULL|NULL.
* *-*-* *:*:*|ec_ll_failed|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[warn|Cabinet Controller Air Speed Fault]|Cause:*|Text:*|WARN|AIRSPEED|/data/*/*/*/*:*| \\
    Minimum soft limit exceeded!|Data=*|Limit=*..
* *-*-* *:*:*|ec_ll_failed|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[alert|Cabinet Controller Air Speed Fault]|Cause:*|Text:*|ALERT|AIRSPEED|/data/*/*/*/*:*| \\
    Minimum hard limit exceeded!|Data=*|Limit=*..
* *-*-* *:*:*|ec_sedc_warning|src:::*:*|pri:*|seqnum:*|svc:::*-* *: * [health fault] <expected:*> |
* *-*-* *:*:*|*|src:::*-*|pri:*|seqnum:*|svc:::*-*(*)[alert|Module Health Fault]|Cause:*|Text:*|ALERT|NULL|NULL|* is not Good!|NULL|NULL
```

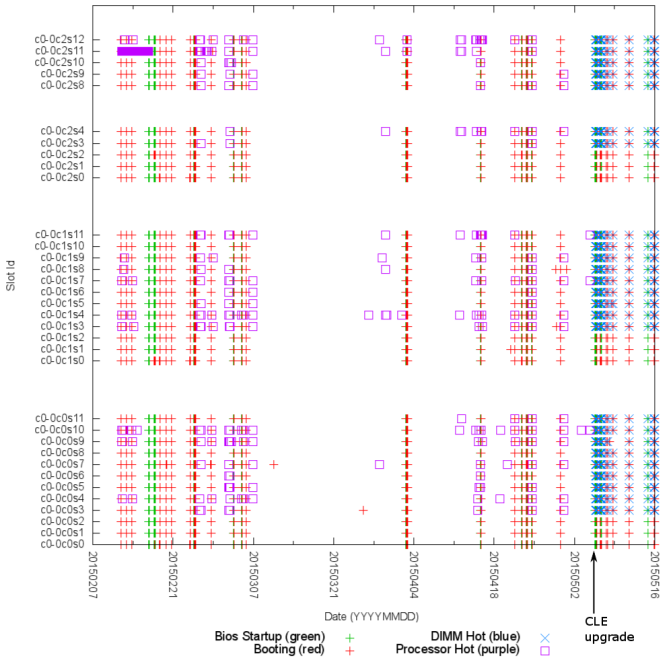
Fig. 4. Facilities testing patterns. The first two are representative of a number of variable values above or below min. Pattern variable data includes particular sensor measurements (e.g., particular voltage sensors, fan speeds etc) relevant to the tests.

at the blade level since the source of this data is the controller associated with the blade with the node/processor information provided in the message where relevant. Only the populated slots are labeled. Slots 0-2 for any chassis are populated with service, not compute nodes, and do not report the hot patterns.

3) Performance and Temperature: As part of the normal acceptance testing, HPL, a computationally intensive MPI implementation of the high performance Linpack benchmark [8], was run in two different CPU core allocation configurations (16 core, and all 32 cores) with the goal of validating performance numbers associated with the Haswell processors and Aries interconnect. As part of our monitoring investigation, we focused on logs indicative of thermal issues.

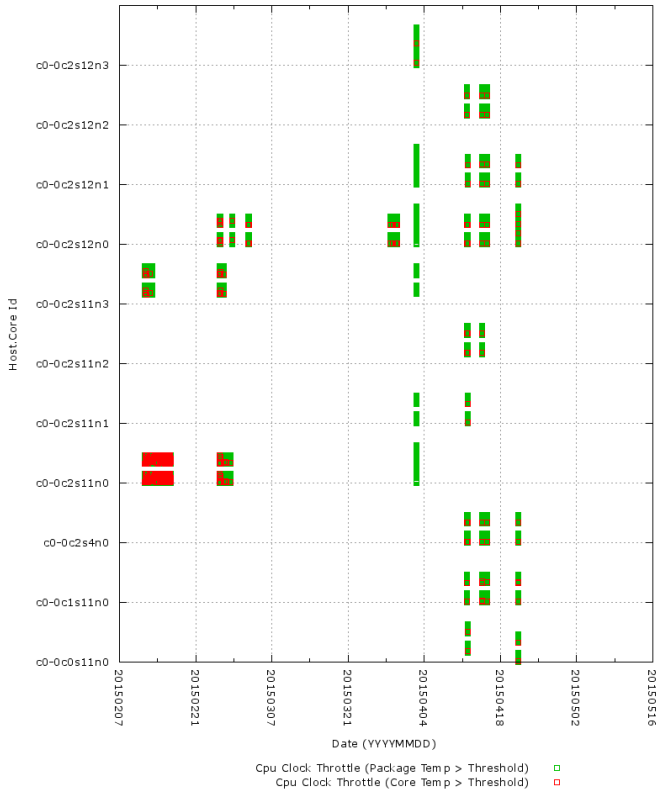
Perhaps more significant than the Processor Hot occurrences reported in Section VI-A2, is temperature based CPU clock throttling. Relevant log patterns with time and location occurrences are shown in Figure 6. Across the 100 compute nodes these patterns have only occurred on 11. Total throttles are shown in the machine layout in Figure 7. In this figure, the anticipated concerns of the airflow in Section II-C are evident as there are substantially more throttles in the rightmost cores and also in nodes where the left and right slots are both populated with compute blades. For some HPL runs, we have seen CPU temperature variation of around 25° C across the system and up to 10° C in the same blade even with no thermal throttling occurring [9]. However, the hottest spots are in accordance with the general characteristics of where the throttling is occurring.

This investigation has further motivated monitoring in order to more fully understand the performance and aging effects of the temperature and throttling events. In addition, such monitoring also motivates operational decisions, for example, what




```
Patterns:
 * * - [*@*]  BIOS startup *
 * * - -  *: node *-* is  booting.
 * * - -  Node * interrupt *=*, *=*, *=* *[*]: * * * * * DIMM * & * Memory Hot *[*]: \\
          * * * * * DIMM * & * Memory Hot *[*]: * * System Management *[*]: \\
          * * * * * DIMM * & * Memory Hot *[*]: * * * * * DIMM * & * Memory Hot *[*]: \\
          * * Alert Over Temperature
 * * - -  Node * interrupt *=*, *=*, *=* *[*]: * * * * * DIMM * & * Memory Hot *[*]: \\
          * * * * * DIMM * & * Memory Hot *[*]: * * * * * DIMM * & * Memory Hot *[*]: \\
          * * * * * DIMM * & * Memory Hot
 * * - -  Node * interrupt *=*, *=*, *=* *[*]: * * * Processor Hot
 * * - -  Node * interrupt *=*, *=*, *=* *[*]: * * * Processor Hot *[*]: \\
          * * Alert Over Temperature
```

Fig. 5. DIMM Hot patterns occur only during booting/startup and only occur after the CLE upgrade. Processor Hot is reported throughout. (top). Representative patterns relevant to this figure (bottom).




```
Patterns:
 *: Core temperature above threshold, * clock throttled (total events = *)
 *: Package temperature above threshold, * clock throttled (total events = *)
```

Fig. 6. Occurrences of CPU clock throttled due to Core or Package temperature above threshold. Only nodes for which a throttle occurs are shown. Host format is `Host.Core` as a floating point. Each processor has 16 cores each with 2 hardware threads. In the case of Package temperature above threshold messages, all 32 threads in that package send a message. Since the core threads are numbered 0-15 and 32-47 for processor 0 and 16-31 and 48-63 for processor 1, a split vertical green line represents a single processor and a solid vertical green line represents both processors. The Core temperature above threshold reports for both hardware threads but only on a core granularity. Patterns are shown at bottom.

guidance should we give users about resource utilization and location-based placement for highly computationally intensive and performance sensitive applications?

B. Network

1) Aries: As part of the temperature based throttling (Section VI-A3) investigations, the thermal grease on a node on a blade was checked by pulling the blade, without a subsequent network link re-tuning upon reinsertion. While this was not a planned corner-case test, it is reflective of the fact that there are many system checks and changes during standup that can result in informative issues. We found a substantial number of log lines in the *netwatch* files reporting link issues with the onset of the errors traced back to this hardware checking event. The occurrence is shown as feature (a) in Figure 1 with the log lines in Figure 8. This was particularly informative for finding out about network errors as we expect that it is unlikely that we would see many network errors in the normal course of events on a machine of this size.

The relevance of the first pattern in Figure 8 was discovered via its frequency and proximity to other patterns.

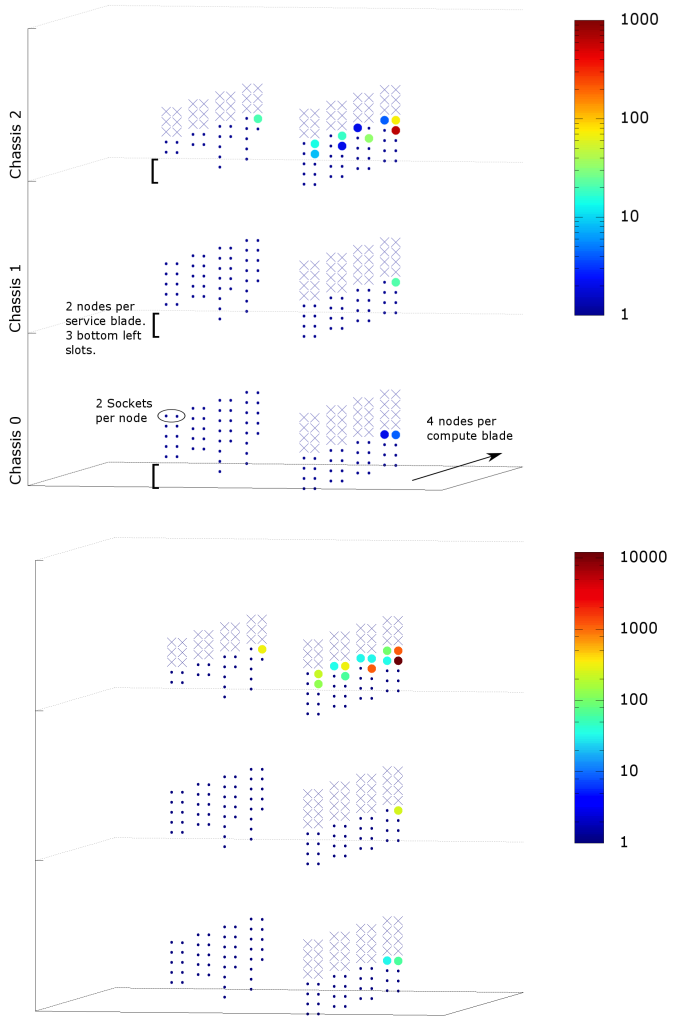


Fig. 7. Location of throttles in machine layout colored by total number of occurrences. Smallest circles indicate no throttles. `Core temperature above threshold` (top); `Package temperature above threshold` (bottom). Package temperature related throttles should result in all threads per core for all cores throttling (e.g., 32 lines per occurrence) although we have seen that not happen; however, as a rough indicator of distinct occurrences divide the package occurrences values by 32. There are generally more occurrences in the expected hotter locations.

The full text is `cb_hw_error: failed_component c0-0c2s12a0l16, type 37, error_code 0x62ff, error_category 0x0`. In the future, adding the now known `failed_component` or `cb_hw_error` to the dictionary, would make this pattern more obviously meaningful. An analogous add could address the first pattern in the bottom which is `Time of last linktune`.

All Aries issues do not result in lines in the same log. Feature (e) in Figure 1 is a result of Aries errors on a particular blade that also resulted in Lustre errors (thus affecting concurrent Lustre testing) in the *console* log. These cleared up when that blade was reseated.

C. Electrical

1) Power Capping and Sensor Data Availability: Power capping is a new feature in our capability HPC systems. From

```
* *-*-* *:*:* *-* * *: * *-*, type *, * *, * *
* *-*-* *:*:* *-* * *: MMR[*]=*
* *-*-* *:*:* *-* * *: link *-* reported as failed due to showing too many soft errors
* *-*-* *:*:* *-* * *: handling failed link *-*
* *-*-* *:*:* *-* * *: adding link *-* to * list


Time of last *: * * * *:*:* *
* *:*:*  *-*     *-*    Info      Router Input Queue saw */* error
* *:*:*  *-*     *-*    Info      * Lane Degrade, *: *
* *:*:*  *-*     *-*    Info      Reinitialized By Peer
* *:*:*  *-*     *-*    Info      * CRC Error, *: *  \\
                                  *: [*] *.*-* [*] *.*-* [*] *.*+* [*] *.*-*
* *:*:*  *-*     *-*    *         Aries * operating badly and will * shut down
* *:*:*  *-*     *-*    *         Link Alive Went Away
* *:*:*  *-*     *-*    Info               Multi-Lane * CRC Error
```

Fig. 8. Aries (reduced) error patterns. *nlrd* (top) *netwatch* (bottom).

a facilities and operations perspective, we are interested in power utilization control. Thus power profiling and power cap testing by interested researchers was included during Mutrino's standup. This team reported failure of the power cap command, with substantially more frequent failure occurring after the CLE upgrade. From a monitoring perspective, we are interested in how to detect such failures and their causes from the logs. In this case patterns relating to failure in the power capping occurred, as expected in the *power_management* log, but also in the *controller/messages* log. This is because the communication to set the power cap is issued to the blade; power capping then fails due to a communication failure [10].

Related patterns are shown in Figure 9. Log patterns resulting directly from power capping related issues are shown in the top (failure to occur, power budget exceeded). Possible communication issues are shown in the middle. The latter 2 (too long for master) occur only after the CLE upgrade and produced a volume of over 9 million lines in 5 days relating to just 5 of the slots. The bad i2c value message occurs even before the CLE upgrade, and accounts for 50 million lines. These lines combined account for the majority of the increase in log lines for the *controller/messages* after the CLE upgrade (Figure 1 (bottom) feature (f)).

While investigating thermal issues, we noticed that some of the same nodes that failed to power cap also failed to report thermal data via SEDC. We also learned from Cray that some of the SEDC data is obtained over the same network interface that the power capping commands use [10]. These thus were related symptoms of the same failure though the association was only made due to investigation of both behaviors at the same time. Patterns we believe are related to the failure to get the sensor data are in Figure 9 (bottom) and account for 25 million lines. A similar occurrence of these lines is shown in Figure 1 (bottom) feature (g) occurring before the CLE upgrade, though not as frequently. In that case rebooting for our facilities testing temporarily cleared up the error.

VII. Conclusion and Future Work

In this paper, we have shown that, while new systems present challenges with respect to understanding behavioral characteristics due to new features and unknown logging, system understanding can be enhanced by appropriate monitoring and analysis during new system standup, testing and corner case stress testing. Specifics presented in use cases here illustrate features of interest common to HPC systems, such as processor technologies, interconnects, packaging, cooling infrastructure and software changes, and impact thermal, electrical, network, and performance characteristics.

```
* - - -  Error enabling the * power monitor!
* * - -  Node * power budget exceeded! Power=*, Limit=*, Max Correction Time=*
* * - -  Node * now within power budget. Power=*, Limit=*
* - - -  Node * Warning: deleting temporary power policy ID #* failed
* - - -  Error setting power policy ID # * to * watts
* - - -  Node * Error setting power budget
* - - -  Error disabling the * power monitor!Node * power budget set to * watts
* - - -  Node * Error: setting temporary power policy ID # *
* * - [*@*] *: : ERROR: error sending power profiles..
* * - [*@*] *: : ERROR: Error sending power limit..


kernel - - -  *: *:* bad i2c * value, i2c *=*
kernel - - -  *: *:* waited too long for master, *=-* (* *, * *, * *, * *)
kernel - - -  *: *-* waited too long for master, *=-* (* *, * *, * *, * *)


* * - -  get_sensor_reading() failed with *
* * - -  *: get_sensor_reading failed with (*)
```

Fig. 9. Power capping related patterns (top). Possible diagnostic patterns for communications issues (middle); the latter two occur only after the CLE upgrade. Related inability to get sensor data suspected patterns (bottom).

We have recently installed a patch for the power capping issue that Cray has backported to CLE 5.2 UP03 for us. We will investigate the effects of this patch on the reliability of the SEDC data. We continue to investigate the potential performance effects of the temperature issues, with and eye to how they may vary in Trinity due to the fully populated racks.

Practical application of the specific insights from this work will influence further operations and monitoring of the test system and Trinity. The monitoring framework in design by the ACES Trinity team incorporates all log and numerical data, expanding on the sources mentioned here, including node level and facilities data to obtain a more complete picture of system behavior under normal and abnormal conditions.

Acknowledgments

We would like to thank Kevin Pedretti (SNL) of the ACES power team for power and thermal related testing and issue reporting; Dave Morton (LANL) of the integration team for discussions on temperature issues; and Dave Martinez (SNL) and Ron Velarde (LANL) of the facilities team and Greg Hamilton (Cray) who performed the Mutrino facility testing.

References

[1] J. Kim *et al.*, "Technology-driven, highly-scalable dragonfly topology," *SIGARCH Comput. Archit. News*, vol. 36, no. 3, pp. 77–88, Jun. 2008.

[2] G. Faanes *et al.*, "Cray Cascade: A Scalable HPC System Based on a Dragonfly Network," in *Proc. Int'l Conference on High Performance Computing, Networking, Storage and Analysis (SC12)*, 2012.

[3] "Using and Configuring System Environment Data Collections (SEDC) Cray Doc S-2491-7001," 2012. [Online]. Available: http://docs.cray.com/books/S-2491-7001/S-2491-7001.pdf

[4] "intro_LLM man page," Jan 2014.

[5] N. Taerat, J. Brandt, A. Gentile, M. Wong, and C. Leangsuksun, "Baler: deterministic, lossless log message clustering tool," *Computer Science - Research and Development*, vol. 26, no. 3-4, pp. 285–295, 2011.

[6] A. Gainaru, F. Cappello, S. Trausan-Matu, and B. Kramer, "Event log mining tool for large scale HPC systems," in *Proc. Euro-Par*, 2011.

[7] "Splunk: Operational Intelligence, Log Management, Application Management, Enterprise Security, and Compliance." [Online]. Available: http://www.splunk.com

[8] "HPL." [Online]. Available: http://www.netlib.org/benchmark/hpl/

[9] J. Brandt *et al.*, "Enabling Advanced Operational Analysis Through Multi-Subsystem Data Integration on Trinity," in *Proc. Cray User's Group*, 2015.

[10] S. Martin and D. Rush, private communication, April 2015, Cray, Inc.