Lessons Learned From the Analysis of System Failures at Petascale: The Case of Blue Waters

Catello Di Martino, Zbigniew Kalbarczyk, Ravishankar K. Iyer
University of Illinois at Urbana-Champaign
Email: {dimart,kalbarcz,rkiyer}@illinois.edu

Fabio Baccanico[1], Joseph Fullop[2], William Kramer[2]
[1]*Federico II University of Naples*
[2]*National Center for Supercomputing Applications*
Email: fbaccanico@unina.it,{fullop,wkramer}@ncsa.illinois.edu

Abstract—**This paper provides an analysis of failures and their impact for Blue Waters, the Cray hybrid (CPU/GPU) supercomputer at the University of Illinois at Urbana-Champaign. The analysis is based on both manual failure reports and automatically generated event logs collected over 261 days. Results include i) a characterization of the root causes of single-node failures; ii) a direct assessment of the effectiveness of system-level failover as well as memory, processor, network, GPU accelerator, and file system error resiliency; and iii) an analysis of system-wide outages. The major findings of this study are as follows. Hardware is not the main cause of system downtime. This is notwithstanding the fact that hardware-related failures are 42% of all failures. Failures caused by hardware were responsible for only 23% of the total repair time. These results are partially due to the fact that processor and memory protection mechanisms (x8 and x4 Chipkill, ECC, and parity) are able to handle a sustained rate of errors as high as 250 errors/h while providing a coverage of 99.997% out of a set of more than 1.5 million of analyzed errors. Only 28 multiple-bit errors bypassed the employed protection mechanisms. Software, on the other hand, was the largest contributor to the node repair hours (53%), despite being the cause of only 20% of the total number of failures. A total of 29 out of 39 system-wide outages involved the Lustre file system with 42% of them caused by the inadequacy of the automated failover procedures**

Index Terms—**Failure Analysis, Failure Reports, Cray XE6, Cray XK7, Supercomputer, Machine Check, Nvidia GPU errors.**

I. Introduction

Failures are inevitable in large-scale high-performance computing systems. Error resiliency (i.e., the ability to compute through failures) is strategic for providing sustained performance at scale. In this context it is important to measure and quantify what makes current systems unreliable. Such analyses can also drive the innovation essential for addressing resiliency in exascale computing.

In this paper, we study the failures of Blue Waters, the Cray sustained petascale machine at the National Center for Supercomputing Applications (NCSA), at the University of Illinois at Urbana-Champaign during a period of 261 days (March 1, 2013 to November 17, 2013) starting just a few days before Blue Waters went into official production (March 27 2013). Blue Waters is a new-generation supercomputer, which provides sustained petaflops (with a peak of 13.1 petaflops) for scientific and engineering applications. That high performance is achieved using a hybrid architecture, which includes general-purpose computing nodes (22,640 Cray XE6 machines) and graphic accelerators (3,072 Cray XK7 hybrid nodes equipped with cutting-edge Nvidia GPU accelerators). In support of computation, Blue Waters provides access to the largest Lustre parallel file system installed to date, with more than 26 petabytes of available storage (36 PB raw).

Failures of large-scale systems have been analyzed in the past [1]–[8]; this study focuses on a detailed characterization of single node failures as well as system-wide outages of a sustained petascale machine. Blue Waters is a configuration corner case well beyond the scales at which hardware and software are methodically tested; as such this study's findings have significant value. Specifically, our analysis i) characterizes single-node failures as well as system-wide outages by analyzing the manual system reports compiled by the Blue Waters maintenance specialists, ii) quantifies the effectiveness of protection mechanisms (e.g., network and file system failover), and iii) based on the manual failure report and automatically collected syslogs (about 3.7 TB), provides an in-depth understanding of hardware error resiliency features and measure the rates of correctable and uncorrectable errors for all cache levels and RAMs (including system and GPU accelerator RAM memories). The key findings of this work are reported in Table I. In summary, our major findings are:

- Software is the cause of 74.4% of system-wide outages (SWOs). Close to two-thirds (62%) of software-caused SWOs resulted from failure/inadequacies of the failover procedures, such as those invoked to handle Lustre failures. More research is needed to improve failover techniques, including the way failure recovery software modules are tested in large-scale settings.
- Hardware is highly resilient to errors. Out of 1,544,398 analyzed machine check exceptions, only 28 (0.003%) resulted in uncorrectable errors, showing the value of the adopted protection mechanisms (Chipkill and ECC). The GPU accelerator DDR5 memory, protected only with ECC, is 100 times more sensitive to uncorrectable errors than DDR3 node RAM is. This finding shows the need for a better technique to be employed to protect the memory of GPU accelerators memory when both CPUs and GPU accelerators are used to create future large-scale hybrid systems.

The findings presented in this paper are of interest to a broad audience, including practitioners and researchers. Our findings demonstrate that the resiliency features of Blue Waters differ from those observed in former generations of supercomputers (e.g., [4], [9], [10]), highlighting new challenges to address in order to approach the next-generation exascale systems.

TABLE I: Findings and implications

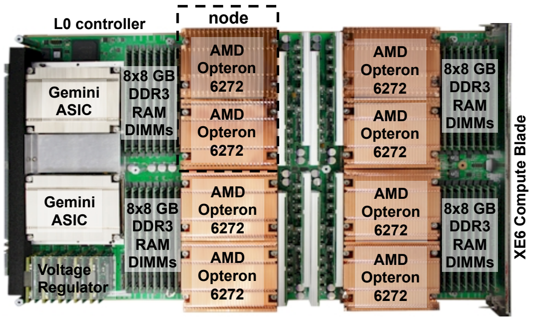
Findings from overall failure characterization (Section IV)	***Implications***
Hardware caused more failures than any other factor (51% of single/multiple node failures and 42% of total failures), but those failures were responsible for only 23% of the total node repair hours.	Hardware failures are well-managed by the Cray system and are easily diagnosed through machine check logs and POST checks. The programmed failover techniques are effective in recovering from hardware failures.
Software was the largest contributor to the total node repair hours (53%), despite being the cause of only 20% of the failures.	Software is critical in systems at the scale of Blue Waters and beyond. Failures due to software root causes are more likely to affect multiple nodes.
The Lustre distributed file system account for 46% of the software-caused failures.	Lustre problems were the most common software-caused failures. Lustre plays a crucial role in Blue Waters, since all the compute nodes are diskless and rely on Lustre for file system access and failures caused by Lustre are potentially critical.
Findings from characterization of hardware errors (Section V)	***Implications***
Error-correcting code (SEC-DED and Chipkill) techniques detected and corrected 99.997% of memory and processor errors. Only 28 out of 1,544,398 single and multiple bit errors were uncorrectable.	This finding shows that hardware is not a major problem for system-wide reliability despite a measured rate of 250 errors/h. This shows that it is worthwhile to invest in state-of-the-art detection and recovery to achieve sustained performance in the presence of high error rates.
DDR5 RAM of Nvidia GPU accelerator, protected by ECC, showed a rate of uncorrectable error per GB 100 times higher than the rate of uncorrectable errors for the nodes' DDR3 memory protected by x8 Chipkill.	The impact of memory errors in the GPU accelerator memory could represent a serious menace for the development of future large-scale hybrid supercomputers. Improved protection techniques would be needed when increasing the size of the GPU memory.
The hardware failure rate decreased with time. Software failure rates did not decrease over time.	Hardware tends to become mature (i.e., flat part of the bathtub curve) earlier than software. Defective hardware can be replaced in a few months, contributing to a decrease of the failure rate. More work is needed to improve the way large-scale software systems are tested.
Findings from characterization of system-wide outages (SW, Section VI)	***Implications***
Software was the cause of most (74.4%) SWOs and caused 68% of Blue Waters downtime hours. Close to two-thirds (62%) of software-caused SWOs resulted from failures of the failover procedure invoked to handle Lustre problems.	The failure recovery of Lustre is a complex procedure that can even more than 30 minutes. The mechanisms behind Lustre failover (e.g., timeouts and distributed lock management) operate at their limits at the scale of Blue Waters. New failover solutions at scale may be required to support future supercomputers.
SWOs caused by single-node failures were rare; they occurred in 0.7% of all reported single-node failures. 99.3% of hardware-related failures did not propagate outside the boundary of a single blade. Software-related failures propagate to multiple blades in 14.8% of the cases, i.e., 20 times more often than for hardware-related failures.	The error-protection mechanisms were effective in containing failures within the boundaries of a single node.

II. About Blue Waters

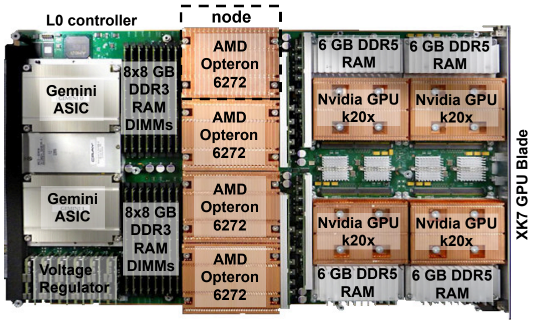
Blue Waters is a sustained petaflop system capable of delivering approximately 13.1 petaflops (at peak) for a range of real-world scientific and engineering applications. The system is equipped with:

- 276 Cray liquid-cooled cabinets hosting 26,496 nodes and 1.47 PB of RAM across 197,032 RAM DIMMs. Each cabinet consists of an L1 cabinet controller, several fan trays, power conversion electronics, breakers, a blower and chiller, and related piping. Each cabinet is organized in 3 chassis, and each chassis hosts 8 blades;
- 22,640 compute nodes (based on AMD Opteron processors) with a total of 724,480 cores;
- 3,072 GPU hybrid nodes equipped with Nvidia K20X GPU accelerators and AMD Opteron processors;
- 784 service nodes with a total of 5,824 available cores;
- The high-speed Cray Gemini network, to provide node connectivity;
- The online storage system, consisting of 198 Cray Sonexion 1600 storage units equipped with 20,196 disks, and 396 SSDs (used to store file system metadata) that provide access to 26 petabytes (36 raw) of usable storage over a Lustre distributed file system; and
- 300 petabytes (380 raw) of usable near-line tape storage.

Compute node hardware. Compute nodes are hosted in 5,660 Cray XE6 blades (see Figure 1.(a)), 4 nodes per blade. A compute node consists of 2 16-core AMD Opteron 6276 processors at 2.6 GHz. Each Opteron includes 8 dual-core AMD Bulldozer modules, each with an 8x64 KB L1 instruction cache; a 16x16 KB L1 data cache; an 8x2 MB L2 cache (shared between the cores of each Bulldozer module), and a 2x8 MB L3 cache (shared among all the cores). Each compute node is equipped with 64 GB of DDR3 RAM in 8 GB DIMMs. System memory is protected with x8 Chipkill [11], [12] code that uses eighteen 8-bit symbols to make a 144-bit ECC word made up



(a)



(b)

Fig. 1: Blue Waters blades: (a) compute (Cray XE6), (b), GPU (Cray XK7).

of 128 data bits and 16 check bits for each memory word. The x8 code is a single-symbol correcting code, i.e., it detects and corrects up to 8-bit errors. L3, L2, and L1 data caches are protected with ECC, while all the others (tag caches, TBLs, L2 and L1 instruction caches) are protected with parity.

GPU node hardware. GPU nodes are hosted in 768 Cray XK7 blades, 4 nodes per blade (see Figure 1.(b)). A GPU node consists of a 16-cores Opteron 6272 processor equipped with 32 GB of DDR3 RAM in 8 GB DIMMs and a Nvidia

K20X accelerator. The accelerators are equipped with 2,880 single-precision Cuda cores, 64 KB of L1 cache, 1,536 KB of dedicated L2 cache, and 6 GB of DDR5 RAM memory with the latter protected with ECC. After this study, 1152 additional XK7 GPU nodes were added to Blue Waters.

Service node hardware. Service nodes are hosted on 166 Cray XIO blades and 30 XE6 blades, 4 nodes per blade. Each XIO service node consists of a 6-core AMD Opteron 2435 Instanbul working at 2.3 GHz and equipped with 16 GB of DDR2 memory in 4 GB DIMMs protected by x4 Chipkill (with single symbol error correction and dual-symbol error detection capabilities, with 4-bit per symbol). Service nodes host special PCI-Express cards such as Infiniband and fiber-channel cards. A service node can be configured as i) a boot node to orchestrate system-wide reboots; ii) a system database node to collect event logs; iii) a MOM node for scheduling jobs; iv) a network node to bridge external networks through Infiniband QDR IB cards; or v) as an Lnet (Lustre filesystem network) node to handle metadata (via Lustre metadata servers, or MDSes, to keep track of the location of the files in the storage servers) and file I/O data (via Lustre Object Storage Servers, or OSSes, to store the data stripes across the storage modules) for file system servers and clients.

All the blades are powered though 4 dual-redundant Cray Verty voltage regulator modules (see Figure 1), one per node, fed by a power distribution unit (PDU) installed in the cabinet and attached to the blade through a dedicated connector.

Network. Blue Waters high-speed network consists of a Cray Gemini System Interconnect (see Figure 1). Each blade includes a network mezzanine card that houses 2 network chips, each one attached on the HyperTransport AMD bus [9] shared by 2 CPUs and powered by 2 mezzanine dual-redundant voltage regulator modules. The topology is a three-dimensional (3D) 23x24x24 reentrant torus: each node has 6 possible links towards other nodes, i.e., right, left, up, down, in, and out.

Hardware Supervisor System (HSS) and System Resiliency Features. Every node in the system is checked and managed by the HSS. Core components of the HSS system are i) the HSS network; ii) blade (L0 - see Figure 1) and cabinet (L1) controllers in charge of monitoring the nodes, replying to heartbeat signal requests and collecting data on temperature, voltage, power, network performance counters, runtime software exceptions; and iii) the HSS manager in charge of collecting node health data and executing the management software. Upon detection of a failure, e.g., a missing heartbeat, the HSS manager triggers failure mitigation operations. They include i) warm swap of a compute/GPU blade to allow the system operator to remove and repair system blades without disrupting the workload; ii) service node and Lustre node failover mechanisms, e.g., replacement of IO nodes with warm-standby replicas; and iii) link degradation and route reconfiguration to enable routing around failed nodes in the topology. The procedure in the communication path route consists of i) waiting 10 seconds to aggregate failures; ii) determining which blade(s) is/are alive; iii) quiescing the Gemini network traffic; iv) asserting a new route in the Gemini chips; and v) cleaning up and resuming Gemini. The total time to execute that procedure is around 30 to 60 seconds. In the case of Gemini link failures, applications running on the affected blades are either killed or, in the case of a warm swap-out, allowed to complete. In the case of inactivity or miscommunication of one of the agents, the HSS manager switches node state to ”suspect” or ”down”. The job currently running on the node, if not failed, is allowed to complete.

System Software. Compute and GPU nodes execute the lightweight kernel Compute Node Linux (CNL) developed by Cray. The operating system is reduced to a bare minimum to minimize the overhead on the nodes and includes only essential components, such as a process loader, a Virtual Memory Manager, and a set of Cray ALPS agents for loading and controlling jobs. Service nodes execute a full-featured version of Linux, the Cray Linux Environment (CLE), which is based on the Suse Linux Enterprise Server 11 kernel 3.0.42.

Jobs are scheduled through a suite of cooperating software that includes i) Cray ALPS (the Application Level Placement Scheduler) [13] for the placement, launch, and monitoring of all the applications composing a single job, and ii) Moab and TORQUE [14] for resource management and scheduling decisions. Jobs are assigned at the granularity of the node.

File System Features. All blades are diskless and use the shared parallel file system for IO operations. Blue Waters hosts the largest Lustre installation to date. It consists of parallel file system used to manage data stored in Cray Sonexion 1600 [15] storage modules. Each Sonexion module has i) 2 SSD of 2 TB in a RAID 1 configuration for journaling and logging, ii) 22 disks of 2 TB for metadata storage, and iii) 80 disks of 2 TB for data storage, organized in units of 8 disks in RAID 6. All disks are connected to two redundant RAID controllers. In each unit, two additional disks serve as hot spares, which automatically provide failover for a failed drive. Data are accessed transparently from the nodes via the Lustre service nodes (Lnet), which interface with the Sonexion storage modules. Blue Waters includes three file systems i.e., project, scratch, and home, and provides up to 26 PB of usable storage over 36 PB of raw disk space. Each Lustre service node and Sonexion module is configured as an active-passive pair connected to a shared storage device. In this configuration, the passive replica node becomes active when the HSS detects a failure of a Lustre node; the shared storage device is then mounted in a read-only mode to avoid data inconsistency until the failed node has been replaced with the standby replica. After failure recovery, clients reconnect and replay their requests serially in order to reconstruct the state on the replaced node. Until a client has received a confirmation that a given transaction has been written to stable storage, the client holds on to the transaction (waiting on a timeout), in case it needs to be replayed. If the timeout is reached, all the jobs waiting to reconnect fail. The recovery process (i.e., the boot-up of the warm standby and the reconstruction of the state lost in the failure) may take 5–30 minutes (60 for MDS), depending on the number of clients using the file system at the moment of the failure.

Workload. The workload processed by Blue Waters consists of large-scale scientific simulations, including molecular dynamics, hurricanes and tornadoes (the Weather Research and Forecasting run on Blue Waters is the largest WRF simula-

TABLE II: Example of Blue Waters Incident Report. Showed entries include only a subset of fields used in this study

Start Date	End Date	Category	Facility	Subject	Failed Nodes	SWO	FRU
3/6/13 7:38	3/7/13 17:00	Single/Multiple Node		c2-8c1s3n3(RQ12110251) Memory Error	1	0	DIMM
3/11/13 21:11	3/12/13 0:21	Interrupt	Sonexion	SCSI and SAS errors on snx1003n181 - crashed and automatic failure failed	26563	1	
3/28/13 2:52	3/29/13 14:52	Failure (No Interrupt)	Sonexion	snx11003n354 Slot 47 Disk Failure	0	0	Disk
4/22/13 7:28	4/22/13 18:08	Interrupt		c5-8c0s6 (rq11520016) node failure made the HSN non-routable	26534	1	Comp. Blade
4/29/13 16:28	4/29/13 16:49	Failure (No Impact)	Sonexion	snx11003n[150-151] Warning Fan at speed 0 RPM.	0	0	Fan Tray
5/30/13 4:50	5/30/13 9:48	Interrupt	Gemini	Blade failed. Replaced, warm swapped then there was a throttle and routes hung	26573	1	

tion ever documented), supernovae, the formation of galaxies, earthquakes, and fluid dynamics. Blue Waters is capable of delivering above 1 PF/s of sustained performance and jobs may use compute nodes, GPU nodes, or both. Since the beginning of the production phase (March 2013), Blue Waters processed more than 190,408 distinct jobs that required a total of 254,925 billion node hours. The average load (utilization) in terms of used nodes, computed over 24h for the measured period is 71%.

III. Data and Methodology

This study is based on analysis of manual reports generated by Blue Waters system maintainance engineers over a period of 261 days, from January 3, 2013 to November 17, 2013. In addition, we use automatically collected event data logs to provide an in-depth understanding of specific failures, such as uncorrectable machine checks, e.g., hardware exceptions caused by parity or ECC errors.

Failure Reports. System failure reports are human-written documents that record each event in the system that required the attention of the system managers, such as failures, maintenance, system upgrades, and bugs signaled to the manufacturer. Each entry contains several distinct fields, including i) start time of the event, ii) repair time and date, iii) category of the event, iv) affected facility, v) textual description of the event, vi) number of nodes offline because of the event, vii) System-Wide Outage (SWO) flag, and viii) Field Replaced Unit (FRU). Over 1,978 distinct entries (including scheduled maintenance, system expansions, and failure events) were been reported during the measurement interval of 261 days. Table II shows a snippet of the reports.

Failure Categories. Entries in the failure reports are produced upon arrival of alerts from automated detection mechanisms and failure-reporting tools. The Blue Waters maintenance specialists classify each entry added to the failure report using the below categories.

1) Failure (No Interrupt): failures that are naturally tolerated by the architecture, i.e., do not cause node/system downtime or trigger failover actions (e.g., cooling hardware or performance problems).
2) Interrupt (Failover): a critical problem that is successfully handled by the automatic failover mechanisms.
3) Link and Node Failure (Job Failed): the failure of one or more nodes and one or more links that cause the failure of one or more user jobs.
4) Link Failure (No Job Failed): Failures of a single or multiple node(s) that cause a link failure that is successfully handled by the automatic network failover mechanisms;
5) Link Failure (Job Failed): Link failures that cause job loss.
6) Single/Multiple Node Failure: Failures of single or multiple node(s) that do require repair, but do not impact core system functionalities.
7) Interruption (System-Wide Outage): The whole system is unavailable, e.g., because of a system-wide repair or restart. A system-wide outage occurs if specific requirements cannot be met, such as: i) the ability to access all data blocks of all files in the file system; ii) user ability to log in; iii) full interconnection between nodes; iv) access to external storage server (esDM, or external data mover); v) ability to support user applications submission, scheduling, launch, and/or completion; vi) ability to recover from file system or network failures through automated failover operations; and vii) performance (e.g., network bandwidth or file system throughput) above acceptable levels.

Human-generated reports present several challenges. They contain textual descriptions in natural language and cannot be readily analyzed by automatic tools. Failure reports must be filtered from reports on non-failure events in order to avoid biasing the analysis. Therefore, a first step of the analysis consists of purging the non-failure entries and reorganizing the content into a structured database. We (in close partnership with NCSA and Cray engineers) manually reviewed each data record in the failure reports and combined redundant or overlapping records. Finally, manual failure reports may suffer from misdetection and underreporting of noncritical failures. We extensively investigated the potential for underreporting, and concluded that the incidence of failure undereporting is negligible. The reasons are that each manual report is based on an automatic failure/alert trigger provided by the HSS's extensive system-logging mechanisms (see Section II) and the technical staff of Blue Waters is meticulous about recordkeeping. After fixing the problem, the staff update the error report with the identified cause of the event, the fixes applied, and, where applicable, the field replaced unit.

Marchine Check Data. Machine checks are errors detected by the machine check architecture in the northbridge of the processor. Machine check errors may be related to several causes, such as hard or soft memory errors (e.g., bit flips) as well as processor or memory voltage ripple or motherboard problems. Machine check data include a timestamp, the ID of the node that experienced the machine check, and information on the type (e.g., correctable/uncorrectable or detected by the hardware scrubber), physical address, error syndrome, and involved operation (e.g., reading from the memory, or loading the L2 data cache) encoded into a 64-bit word in the logged data obtained after fetching the content machine check status register [11]. In the case of an uncorrectable error, the operating system is configured to panic. Consequently, the event is detected by the system console, which switches the status of the node from up to down.

Failure root causes identified from the reports are classified into the following exclusive categories: *hardware, software,* *missing heartbeat, network, environment*, and *unknown* (i.e., failures for which the cause could not be determined). We consider this a separate category of failures detected by the system but automatically recovered from before they can be diagnosed (see discussion in the Section IV-A.

The assignment of an outage to the hardware or software category is guided by the type of corrective action (e.g., replacement using a spare part, or installation of software patches) and based on interactions with system administrators. After assigning categories, we reduced the number of reports from 1,978 to 1,490 entries, which include only failures events.

A. Characterization Methodology

In characterizing failures from the manual failure reports, we i) associate failures with the corresponding root cause categories, i.e., hardware, software, network, environment, heartbeat/node down, and unknown, and we measure the failure rate and repair times across them; ii) determine how the hardware and software failures evolve in the considered measurement window; iii) evaluate the hardware error resiliency; and iv) evaluate system-wide failure and repair time distributions and their statistical properties. Computed metrics include the relative failure frequency, mean, and standard deviation. In addition, the density distributions of the TBF and relative frequency of root causes are evaluated. As part of the study, we also measure the error resiliency of memory and processors using the machine check data. Specifically, we compute i) failure rate in FITs (failures in 10^9 hours of operation) per GB of memory for RAM and GPU accelerator on-board memory; ii) mean time between failures (MTBF) and annualized failure rate (AFR) for the processor, memory, GPU accelerator, disks, and SSDs; and iii) probability of uncorrectable errors per GB of memory, for both GPU accelerator and memory errors.

The analysis of the data employed in this paper required us to develop several techniques and tools i) to handle the large amount of textual data (e.g., 3.7TB of system logs for the considered period), ii) to extract and decode specific types of system events (e.g., the 64-bit machine check status register for all the logged machine checks), and iii) to harmonize and mine the manual failure reports. Details of the developed toolset are deliberately not included in the paper because of both space limitation and the focus on measurements.

IV. Blue Waters Failure Causes

In this section, we analyze the data and information in the failure reports addressing how often Blue Waters fails and how much time is needed to fix the causes of a failure. We report on i) the distribution of failures' root causes across all the failure categories defined in Section III; and ii) the root causes of the failures, as identified by the Cray and Blue Waters maintenance specialists.

A. Breakdown of Reported Failures

Table III provides a breakdown of the failure reports, MTBF, and MTTR across the defined failure categories. In the measured period of 261 days, the system experienced 1,490 failures, of which 1,451 (97.4%) were tolerated and 39 (2.6%) resulted in system-wide outages (analyzed in Section VI)[1]. Key obser-

[1]The breakdown of Blue Waters failures and number of SWOs reflects NCSA view of the system and may not correspond to that provided by Cray.

TABLE III: Failure Statistics. The last row refers to the statistics calculated across all the failure categories.

Failure Category	count	%	MTBF [h]	MTTR [h]	σ_{TBF} [h]	σ_{TTR} [h]
1) Failure (No Interrupt)	164	11%	35.17	13.5	70.8	35.3
2) Interrupt (Failover)	99	6.6%	58	14.7	92	42.2
3) Link & Node Failure (Job Failed)	19	1.3%	297.7	6.1	427.3	5.4
4) Link Failure (No Job Failed)	285	19.1%	19.9	32.7	51.9	91.2
5) Link Failure (Job Failed)	19	1.3%	291.6	16	444	26.7
6) Single/Multiple Node Failure	868	58.2%	6.7	26.7	6.3	72
7) Interruption (system-wide outage)	**39**	**2.62%**	**159.2**	**5.16**	**174.2**	**8.1**
ALL	**1490**	100%	4.2	34.5	13.3	50.5

vations are:

- On average, there was a failure (across all categories) every 4.2 h, while the system suffered system-wide outages approximately every 160 hours.
- 58.3% of failures resulted in single/multiple node failures (category 6 in Table II) that caused node unavailability. Such failures were the most common, occurring every 4.2 h; they were allowed to accumulate in order to optimize the cost of field intervention, and they were repaired, on average, in 32 hours.
- About one-fourth (25.7%, categories 2 and 4) of failures were potentially severe events from which the system recovered by means of system-level failover (see Section II) without job failures. Only 2.6% caused job failures (categories 3 and 5) without resulting in a SWO.
- 11% of failures (category 1) consisted of noncritical events that were tolerated by the redundant design of Blue Waters without requiring any automatic failover operation. Such events included, for instance, failures of (redundant) power supplies in the blades, failures of cooling hardware (fan trays or water cooling valves), job-scheduling performance problems, and resource manager crashes. Such failures caused no machine downtime and little or no unavailability of software services (e.g., the job scheduler). They occurred on average every 35.12 h, and their inter-arrival times showed high variance due to the heterogeneity of the root.

Software failures contributed to 53% of the node downtime hours. Figure 2.(a) shows how hardware, software, network, heartbeat, and environment root causes are distributed across the failure categories given in Table III. As seen in other systems [4], *failures with hardware root causes were the predominant cause of single/multiple node failures*. They occurred 442 (51%) times over 868 single/multiple node failures documented in the failure reports, and constituted 42% of the total number of failures across all the categories (rightmost bar in Figure 2.(a)). Conversely, failures with software root causes represented 20% of the total number of failures and only 7.9% of the single/multiple node failures. However, an interesting conclusion can be drawn from the relative impact that hardware and software causes have on the total number of node repair hours (hours required to repair failures due to the same root cause, multiplied by the number of nodes involved in the failure) shown in Figure 2.(b). The key observation is that *failures with software root causes were responsible for 53% of the total node repair hours, although they constituted only 20% of the total number of failures*. Hardware root causes, however, *despite causing 42% of all failures, resulted in only 23% of the total repair time*. As we shall see, hardware problems are well

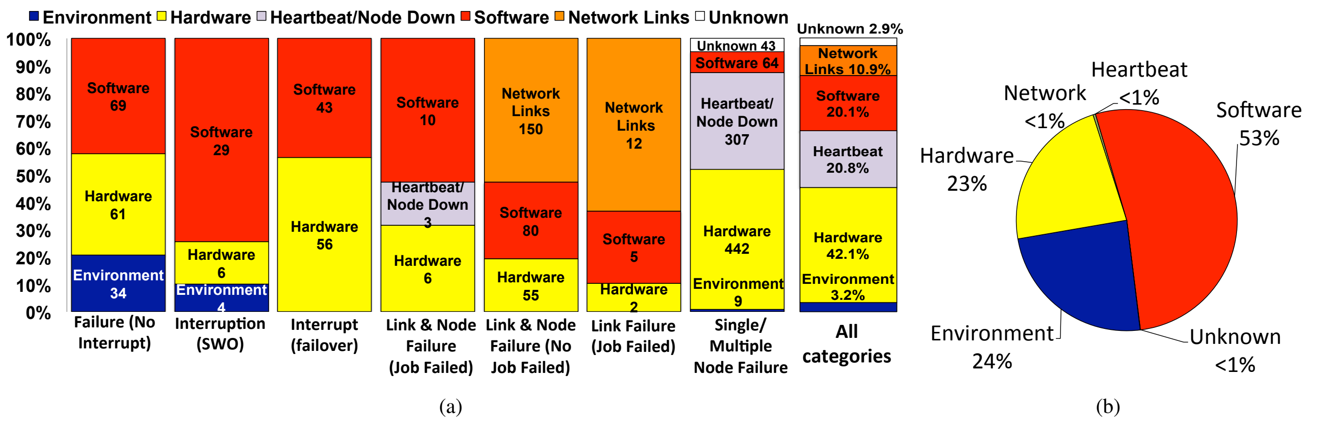


Fig. 2: Breakdown of the failure categories (a), and distribution of the cumulative node repair hours (b), across hardware, software, network, unknown, heartbeat, and environment root causes.

TABLE IV: Breakdown of the count of the top 3 hardware and software failure root causes

	Failure (No Interrupt)		**Interrupt (SWO)**		**Interrupt (Failover)**		**Link Failure (User Job Failed)**		**Link & Node Failure (User Job Failed)**		**Single/Multiple Node Failure**	
HW	PSU	20	EPO	1	Disks	45	Optic	12	GPU	2	Processor	160
	IPMI	15	Compute Blade	2	IPMI	5	RAM	9	Gemini ASIC	1	RAM	158
	Fan tray assy	14	Storage module	2	Storage module	2	Gemini voltage regulator	8	Compute blade	2	GPU	38
SW	Moab/TORQUE	33	Lustre	18	Lustre	29	Lustre net (Lnet)	2	Lustre	8	Lustre	30
	CLE/kernel	17	Moab/TORQUE	6	Sonexion/storage	8			CLE/kernel	1	CLE/Kernel	16
	Warm swap	5	Gemini	3	CLE/	4					Sonexion/Storage	5

managed by the Cray architecture.

To identify a reason for those differences, we analyzed the distribution of the number of nodes involved in failures with hardware or software root causes. We found that failures with hardware root causes that did not cause system-wide outages propagated outside the boundary of a single blade in only 0.7% of the cases. Cases in which failures caused by hardware impacted a full blade involved failures in the voltage converter module (VRM) of the mezzanine and/or in problems with the cabinet controller. Data show that failures with hardware root causes were limited to a single node or a single blade (i.e., 4 nodes) 96.7% and 99.3% of the times they occurred, respectively. Conversely, software failures, if they did not cause a system-wide failure, propagated to more than 1 node in 14% of the cases, i.e., 20 times more often than the hardware. In addition, hardware is easier to diagnose than software. Hardware failures are fixed in bulk to reduce the cost of field intervention, e.g., after a given number of node failures. Although that does not impact on the system MTTR (5.16 h), it does result in a larger MTTR for single nodes (32.7 h), as reported in Table III.

Figure 2.(b) also shows that failures whose root causes are unknown (2.9% of the total failures; Figure 2.(a)) and lack of heartbeat activity (20.8%) account for less than 2% of the total node repair hours. Similar observations have been made in other Cray machines [16]. In particular, based on discussions with the system maintenance engineers, we determined that only 2% of the failures detected by the heartbeat mechanisms were symptoms of real problems with the nodes. Periodically, each node receives a heartbeat request that triggers a number of specific tests. If the test fails or the heartbeat is not received/delivered for some other reason, the node is marked as down by the system management console. The test is automatically repeated, by default, every 60 seconds for 35 minutes following a failure. Hence, a node can be marked as "down" and later pass the test and return to the "up" state without any intervention.

Table IV shows the breakdown of the top 3 hardware and software root causes over the failure categories. Processors, memory DIMMs, and GPUs are the 3 most frequently replaced hardware components (together accounting for 72% of the replaced components), out of an extensive list of 69 replaced units. As discussed in the next section, processors are replaced when the maintenance specialists observe specific hardware exceptions (e.g., L1 cache parity errors) in the system console logs. In many cases, the replaced processors are tested and found to be free of problems, and the failure is attributed to a problem with the socket or the voltage regulator. Problems in the node or mezzanine (Gemini) voltage regulator accounted for about 14% of single/multiple node failures in our gathered data. In one case, the mezzanine voltage regulator failed because of a bug in the L0 node controller firmware, causing a system-wide outage and showing that *software can have an impact on low-level hardware failures*. Lustre, the CLE OS, and Sonexion/storage software are the top 3 software root causes among the failure categories. In particular, the Lustre file system and related software caused 44% of the single/multiple node failures attributed to software root causes, while the CLE OS and Sonexion/Storage caused as much as 28% and 9% of the total single/multiple node failures, respectively.

Lustre includes a rich software stack of more than 250k lines of code and plays a crucial role in Blue Waters, since all the compute nodes are diskless and rely on Lustre for file system access. Failures caused by Lustre problems are the most widespread cause of failures and are present in 6 out of 7 failure categories in Table IV, and, as detailed in Section VI, 18 of those 104 failures escalated to SWOs.

Figure 3 shows the breakdown of the root causes of Lustre failures. Blue Waters experienced a total of 104 different fail-

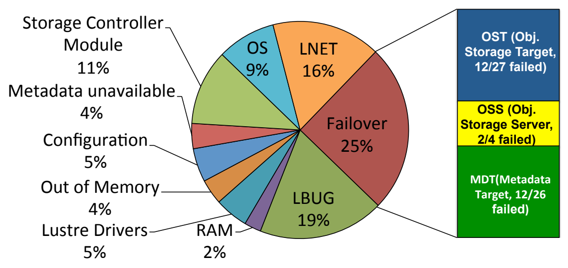


Fig. 3: Breakdown of the Lustre Failures

ures attributed to Lustre. The failure of any Lustre component triggers a failover operations which was successful 75% the times they were invoked (see Figure 3). In case of the inability of the automated failover procedures to recover the normal state of the system, the automated procedures are overridden by the technical staff managing Blue Waters and the system is manually recovered. A common cause of Lustre issues is detection of the so-called LBUG (19%), i.e., a panic-style assertion in the storage node kernel. The unavailability of the Object Storage Target (OST) or of the metadata (i.e., MDS, MDT, and MGS in Lustre) that provide low-level support for the storage system is another important Lustre failure category. Configuration problems contributed to 5% of the Lustre failures, but they were not critical and often manifested as performance issues, i.e., Lustre became slow. About 14% of Lustre failures were due to hardware problems (RAM, 3%, and Controller Module, 11%), and the failover mechanism recovered from all of them.

B. *Effectiveness of Failover*

On average, each day, the system is subjected to 1 or 2 critical failures triggering automatic failover procedures. Hence, an important question is, how effective are the automated failover procedures in recovering from failures due to hardware and software causes?

About 28% of all failures triggered some type of failover mechanism. That included 39 SWOs; 99 interrupt-failovers; 285 link failures in which no user job failed; 19 link failures in which a user job failed; and 19 node failures in which a user job failed, out of a total of 1,490 failures. The data contain information on of 138 of triggered failover operations due to failures attributed to hardware or software causes. Out of 138 recorded failover operations, system-wide outages occurred in 24.6% (39 out of 138) of the times; the remaining 99 failures were successfully recovered (See Figure 2.(a)).

Lustre failover. The failure recovery of Lustre is a complex procedure that typically requires movement of a large volume of data over the network, and synchronization of the nodes participating in data movement. 26 out of 104 failover attempts failed; specifically, i) the failover of the Metadata Target (MDT) failed 12 times out of 26, ii) the failover of the Object Storage Server (OSS) failed 2 time out of 4, and iii) the failover of the Object Storage Target (OST) failed 12 times out of 27. As one might expect, the most critical failover procedures are those involving the storage and metadata server nodes. Under heavy job loads, those procedures are time-consuming (taking between 15 and 30 minutes). Recovery from a client failure in Lustre is based on revocation and lock of resources, so the surviving clients can continue their work uninterrupted. Because of the way the Lustre Distributed Lock Manager handles parallel operations, threads can block on a single resource. Moreover, threads must hold a resource until clients acknowledge completion of an operation. Even though the system is configured to support 512 service threads, they can quickly all become consumed because of the blocking. In particular, data shows that the Lustre failover fails causing a system-wide outages about 20% of the times because of lock timeouts firing due to the high volume of clients to recover from during the failover (e.g., the high number of active OSSes, see Section II). Those results indicate that *the mechanisms behind Lustre failover procedures (e.g., timeouts) operate at their limit at Blue Waters scale when dealing with high workload conditions and high number of clients involved in the recovery*. In addition, because to the long recovery time and of the dual-redundant configuration of Lustre servers, the chances that failures will occur during the time needed to recover (and hence of a SWO) are not insignificant.

Gemini Failover. The Gemini network failover mechanism seems to be the most resilient. Out of 323 documented link failures (categories Link and Node Failure (Job Failed), Link Failure (No Job Failed), and Link Failure (Job Failed) in Table III), 38 caused the loss of one or more user jobs, while the Gemini-supported failover mechanism succeeded in 285.

In the Gemini network, when a lane (a part of a Gemini network link) fails, the network is automatically able to run in degraded mode (i.e., diverting the traffic to the remaining links at the price of reduced speed). Such problems have 3 main causes: i) bad cable interconnections, ii) failure of a routing node or a Gemini mezzanine card, and iii) network congestion. Evidence shows that the network is agile and fault-tolerant. We speculate that more attention has been paid to Gemini software stack testing, while Lustre is naturally more vulnerable to failures because of i) the richer software stack, ii) the community-driven nature of product, and iii) the lack of effective methods and tools to test how critical modules such as the failover would behave over large-scale deployments like Blue Waters. Blue Waters the largest Lustre deployment to date, and our findings shows that more effort is required to create better failover and testing techniques for improving the resiliency of software working at the petascale and above.

V. Hardware Error Resiliency

Blue Waters maintenance specialists diagnose processor and memory related problems by looking at the machine check exceptions contained in the system logs. In looking at the system logs produced by Blue Waters nodes, we counted 1,544,398 machine check events in the measurement period (i.e., on average, a rate of 250 errors/h), of which only 28 consisted of uncorrectable errors, i.e., errors that cannot be corrected by either ECC or Chipkill and may cause loss of data, corruption of processor state, or both. That indicates an unusual degree of containment of hardware problems that we further investigate in this section. Table V shows the breakdown of the machine check errors over the different node types.

In total, 12,721 nodes (46% of the total Blue Waters nodes) experienced at least a memory error; 82.3% of the nodes that manifested machine checks were compute nodes; 19.4% were GPU nodes and 7.34% were service nodes. 6.6% of

TABLE V: Breakdown of the machine check errors.

	Compute		GPU		Service		All	
	Count	Error/ MB	Count	Error/ MB	Count	Error/ MB	Total	Error/ MB
L1	27,522	1.62	594	0.26	632	3.48	28,748	1.48
L2	18,590	0.05	1,098	0.02	1,566	0.17	21,254	0.05
L3	292,920	0.81	2,282	0.01	23,030	1.98	318,232	0.75
Memory	840,322	5.66E-4	97,974	9.73E-4	93,590	3.93E-3	1,031,886	6.42E-4
Other	101,388	-	1,102	-	41,788	-	144,278	
% CPU	34.39%		4.93%		41.73%		33.19%	
% RAM	65.61%		95.07%		58.27%		54.41%	
Total	1,280,742		103,050		160,606		1,544,398	

all the memory DIMMs generated at least a correctable error during the observation window. In our data, 55% of the nodes generated only 1 machine check, while 92% of the machine checks were generated by only 19% of the nodes.

Memory error breakdown. Table VI shows a breakdown of the memory errors in Blue Waters. The table shows that about 70.01% of the memory errors involved a single bit, and that 29.98% 2–8 consecutive bits (less or equal than two symbols, being a symbol 4 bit), similarly to the results in [17] for a smaller-scale Cray supercomputer. The Chipkill can correct errors affecting up to 2 symbols (x8 Chipkill on compute and GPU nodes) and that, without it, 30% of the analyzed memory errors would be uncorrectable (for instance, if only ECC were used). Hence, a key finding is that *ECC/Chipkill techniques were effective in correcting 99.997% of the memory errors that occurred*, i.e., we observed only 28 uncorrectable errors out of 1,544,398 errors. The data also show that about 8.2% of the DIMMs manifest correctable errors, matching with the data in earlier large-scale studies [9]. However, in our study, we found that fewer than 0.1% of the machines and 0.014% of the total DIMMs generated uncorrectable errors, i.e., 1 order of magnitude lower than the incidences of 1.3%–4% for the machines and 3.5–5.7 times lower than the number of DDR2 DIMMs with uncorrectable errors (0.05%–0.08%) reported in [9]. In particular, the number of uncorrectable errors over the total number of handled errors is more than 3 orders of magnitude lower than that for DDR2 memory systems reported in other large-scale studies [9], even though Blue Waters generates 2 orders of magnitude more machine checks than previous generations of HPC systems [4], [9]. That shows a substantial improvement of resiliency to multiple bit errors of DDR3 over DDR2. A key implication is that the joint use of ECC and Chipkill techniques in Blue Waters was able to fix 99.998% of the errors generated in 1.476 PB of DDR3 RAM and 1.5 TB of L1, L2, and L3 caches across all Blue Waters processors. Only 0.002% of errors were uncorrectable, compared to the 1.29% reported for the previous generation of HPC systems [9] employing only ECC and/or x4 Chipkill (single symbol correction, dual symbol detection). The expectation among hardware designers has been that both transient and permanent hardware failures may rise uncontrollably as device sizes shrink, especially in large-scale machines like Blue Waters. However, results indicate that we are far from that situation. Because of the high numbers of nodes and errors, we claim that the results provided in this section have a strong statistical significance for the characterization of processor and memory error resiliency features.

TABLE VI: Breakdown of the count of memory errors.

Type	Count	%
Total memory errors	1,031,886	66.81%
- ECC/chipkill single bit	722,526	70.01%
- Chipkill (more than 2 bit)	309,359	29.98%
- Uncorrectable ECC/Chipkill	28	2.71E-05%

A. Rate of Uncorrectable Errors Across Different Node Type

Table V also shows that the rates of correctable and uncorrectable errors per node vary across compute, GPU and service nodes. If we look at the rates of errors detected by the memory scrubber for compute, GPU, and service nodes (not reported in Table V) we notice that the nodes have similar levels of physical susceptibility to memory or cache errors, i.e., 29 errors per compute node, 28.47 errors per GPU node, and 30 errors per service node. Therefore, we ruled out the possibility that uncorrectable error rates might be related to the different hardware characteristics (i.e., total installed RAM and different Opterons; - see Section II), and we further investigate in the following.

One observation is that nodes employ different types of Chipkill, i.e., x4 for service nodes and x8 for compute and GPU nodes. In particular, service nodes are able to correct errors on 4 consecutive bits and detect errors on up to 8 consecutive bits, while compute and GPU nodes can detect and correct errors affecting up to 8 bits. That impacts the capacity of service nodes to tolerate multiple-bit errors (for more than 4 bits). In particular, we found that about 1% of the compute and GPU nodes' memory errors involve 4–8 bits, vs. 0.2% on service nodes. The difference is that while those errors are not critical for compute and GPU nodes, they cause uncorrectable errors on service nodes.

The difference in the Chipkill technique used explains only some of the measurements. In particular, as shown in Table VIII, the rate of memory errors per node for service node is 48.1 against 37.1 (23% fewer errors per node) and 31.1 (56% fewer errors per node) for compute and GPU nodes, respectively. An observation is that compute and GPU nodes execute a lightweight OS while service node run a full OS (see Section II). The level of multitasking is different for the different OSes, with 1 application per core for compute and GPU nodes, against several background services for service nodes. Service nodes also show a higher percentage of used memory because of the lower amount of installed memory, namely 16 GB against 64 GB for compute nodes. The high number of active services on service nodes translates into a more sparse memory access pattern than is found on compute or GPU nodes, which usually work on big chunks of data (such as matrices used in compute and GPU nodes, stored in consecutive memory locations, e.g., in the same memory chip or DIMM, because of compiler optimization). Hence, we speculate that the reason service nodes are more susceptible than compute/GPU nodes to uncorrectable memory errors is their higher levels of multitasking, disparities in the memory access patterns, and higher memory loads; however we plan to perform a more rigorous analysis in the future.

B. Hardware Failure Rates

The procedure enforced at NCSA for Blue Waters hardware replacement is to replace i) the processor when uncorrectable or parity errors are observed, and ii) memory when the rate

TABLE VII: AFR, MTBF and FIT for top-5 hardware root-cause.

	Total	AFR	MTBF	FIT/Device	FIT/GB
Processor	49,258	0.23%	3,771,492	265.15	NA
DIMMs	197,032	0.112%	7,821,488	127.84	15.98
GPU card [6 GB]	3072	1.732%	506,394	1974.11	329.02
Disks [2 TB]	20196	0.312%	2,807,692	356.16	0.174
SSD [2 TB]	392	0.717%	1,230,795	812.48	0.397

TABLE VIII: Breakdown of the uncorrectable memory errors (UE).

	RAM [GB]	Errors/node	UE	UE/GB	MTBF (UE)
Compute	1,448,960	37.1	14	1.08E-05	1617h
GPU	127,104	31.1	4	3.88E-05	768h
Service	23,232	48.1	10	6.22E-05	193h
GPU Card	18,432	9.76E-3	38	2.06E-03	80h

of corrected ECC errors over a single address is above a programmed threshold. A similar policy is replacement of storage devices and GPU accelerators when uncorrectable errors are detected, e.g., when a raid controller detects uncorrectable disk errors or when an Nvidia GPU accelerator manifests a double-bit error. In fact, GPU accelerator memory is protected only by ECC and therefore is vulnerable to multiple-bit errors.

Table VII reports the annualized failure rate (AFR, which is the percentage of failed unit in a population, scaled to a per-year estimation), MTBF, and FIT rate for processor, memory DIMM, GPU, and storage devices (including disks and SSDs). Table VII also reports the estimated failure rate expressed in FITs/device. The MTBF for a single component is computed as the total number of working hours divided by the AFR. Interestingly, the DDR3 DIMMs show the highest value of MTBF with 7,821,488 hours. The processors and the disks show a figure for the MTBF about half the DDR3 DIMM MTBF, specifically 3,771,492 hours for processor and 2,807,692 for the disks, while the GPU accelerators show an MTBF of 506,394 hours, i.e., 15 times smaller than the DDR3 DIMM MTBF (about 200 times smaller if comparing the FIT/GB). In fact, disks showed to provide high level of reliability. During the measured 261 days, only 45 disks were replaced from the pool of 20,196 devices. The computed AFR for disks is lower than the observed values of 2%–6% given in other studies of disk failures [5], although the population of disks in Blue Waters is smaller than that considered in other studies. Our numbers, however, confirm the MTBF values provided by the manufacturer and show no tangible evidence of defective disk units; we measured a SSD MTBF lower than the manufacturer's declared value of 2,000,000 hours.

Uncorrectable error over SSDs or disks often implies a permanent error on the device, but no permanent error was observed for processors or GPU accelerators. The NCSA replacement policy for processors and GPU is in fact very conservative. As discussed in the next section, we believe that the low MTBF calculated for the GPUs is the result of an assumption that they be more sensitive to uncorrectable error than processor and memory DIMMs in fact are. In addition, as discussed in section V-C, we believe that data in Table VII also contain measurements related to units that were replaced a few months after the system went into production because of detected defects (e.g., defective SSDs).

Rate of Uncorrectable errors for GPU cards. Table VIII reports the rates of uncorrected memory errors in Blue Waters nodes and Nvidia GPU accelerators. We note i) that for uncorrectable memory errors, the DDR5 memory on the GPU accelerator shows an MTBF 1 order of magnitude smaller than that for the DDR3 constituting the GPU node RAM and 2 orders of magnitude smaller than that of compute nodes (a similar comparison holds for the FIT/GB reported in Table VII); and ii) that the disparity is even higher if we look at the number of uncorrectable errors per GB reported in Table VIII. In particular, we note that the rate of uncorrectable errors per GB on GPU node DDR3 memory is 2 orders of magnitude smaller than that on the Nvidia accelerator DDR5 onboard memory and that the FIT rate per GB of memory of GPU accelerators is 10 times higher than that for the DDR3 RAM of the nodes. An implication is that enforcement of Chipkill coding for compute, and service and GPU node DDR3 RAM can decrease the rate of uncorrectable errors per GB by a factor of 100, compared to the rate for the ECC-protected memories (e.g., the DDR5 memory on the GPU card). The rate of uncorrectable ECC errors on supercomputer GPUs accelerators has not been successfully quantified by any former study for enterprise-level GPU accelerators. The impact of memory errors in GPU accelerator memory could represent a serious threat to creation of future large-scale hybrid systems. For instance, Titan [18] at ORNL adopted about 5 times the number of GPU nodes and cards as Blue Waters, making a substantial step towards fully GPU-based supercomputers.

C. *Hardware Failure Trends*

Our data cover the early months of production of Blue Waters. Therefore, it is important to assess how the failure rates evolved during the measured period. A first test consists of plotting the arithmetic mean of the TBF (Time Between Failures) for failures due to hardware and software root causes and observing whether there is any trend. Second, we want to use the Laplace test to determine whether the arrival of failures changed significantly over time [19]. The Laplace test evaluates a null hypothesis that failure inter-arrival times are independent and identically distributed. A score greater than zero means that there is an upward or increasing trend, and a score less than zero means there is a downward or decreasing trend. When the score is greater than (less than) +1.96 (–1.96), we are at least 95% confident that there is a significant trend upward (downward). That implies that successive inter-arrival failure times tend to become larger (smaller) for an improving (deteriorating) system. A score of zero means that the trend is a horizontal line. We compute the Laplace score for the number of failed nodes per day to analyze the presence of trends.

The failure rate distribution trended to a constant failure rate towards the end of the measured period. Figure 4.(a) shows that the TBF for failures with hardware root causes increased by a factor of 2 towards the end of the measured period. The ripples on the first months of the plot were caused by i) failures of defective units, and ii) pre-production tests run between March 1 and March 27, 2013 (the start of the production time). It is interesting to notice that after users started to use Blue Waters, there was a constant rate of discovery-and-fix of hardware problems. More specifically, looking at Figure 4.(a), the TBFs start to improve in April and continue to improve constantly until they stabilize towards the end of the period under study (the slopes of the charts decrease). At the end the observation period, the hardware MTBF grew

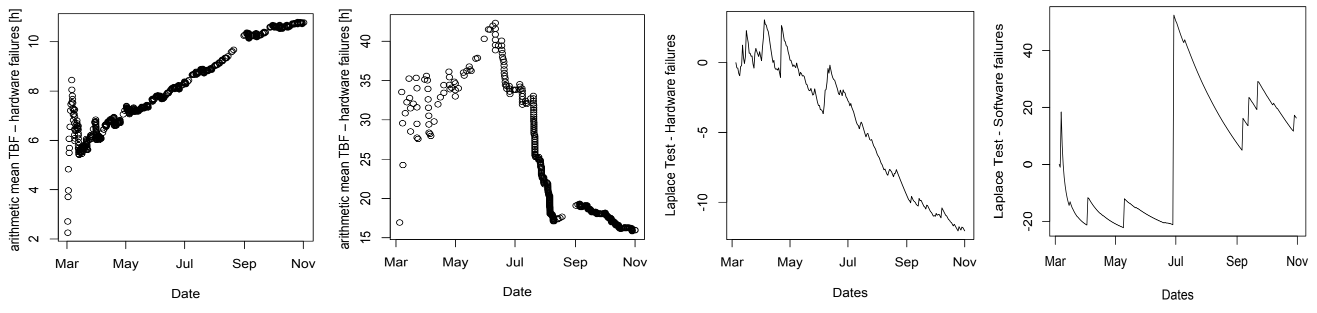


Fig. 4: Arithmetic mean of the TBF for failures due to (a) hardware, (b) software root causes. Laplace Test for the number of nodes failing per day because of (c) hardware, (d) software root causes.

to 10.9 h. Figure 4.(c) shows the Laplace score computed over the examined period. In particular, the figure confirms that in the first months, the hardware reliability was not stable (the score oscillates around zero). From July on, the reliability of the system improved (shown by a Laplace score lower than –1.96), because permanent and non-independent failures had been fixed so that most of the remaining failures were those that occurred with independent inter-failure times. The implication is that over those few months, Blue Waters' hardware configuration matured, going towards the flat part of the bathtub curve. It is interesting to note that Figure 4.(c) shows a peak around July. During that time the system was suspended for replacement of several blades; the new hardware added to the system caused a transient upward trend in the Laplace score showing that new defects were fixed soon after the installation of new hardware.

The reliability of the software deteriorated over the measured months. Figure 4.(b) shows that in the first months, the TBF of failures due to software improved until June, while 4.(d) shows that the number of nodes that failed per day decreased, despite being highly variable (scattered points in the chart). In June–August the TBF decreased sharply (by a factor of about 2.6), from about 42 h to 16 h. During that time, material changes were made to the Blue Waters stack/environment that perturbed the downstream reliability. Examples include major software upgrades, environmental and Gemini network changes. In particular, after a significant upgrade to the Blue Waters system software in July 2013 we noticed that there was a dip in the system reliability as a set of new (previously undiscovered) Lustre issues were excited. Some of the issues were attributed to Lustre (particularly in the area of failovers) and, as described in the reports, were partially caused by the maturity of the product of this scale as well as by Sonexion platform software and firmware defects. As Figure 4.(d) shows, after the major change in July the system experienced more failures, which were followed by a period of improved stability (decrease after the peak). Blue Waters is a configuration corner case at scales well beyond those at which Lustre is methodically tested and, hence, the findings provided in this study have a significant value. At the end of the observation period, the Laplace test shows a decreasing trend in reliability, while the TBF settles at around 16 h, showing that hardware and software follow different lifecycles in Blue Waters. The key implication is that hardware becomes mature (flat part of the bathtub curve) earlier than software. Software failure rate curves do not drop over time after new releases or patches (peaks in Figure 4). We speculate that in the long run, software will show MTBF values similar to these of the hardware, with the important difference that failures due to software causes can impact a larger number of nodes (e.g., file system software stack), as discussed in Section IV. As further detailed in the next section, one implication is that software is and will be the real problem at the scale of Blue Waters and above, further supporting our claim that more research is needed to improve the way large-scale software systems are tested.

TABLE IX: System-wide outage statistics

Availability	0.9688
Total Time	261 days
Unscheduled Downtime	8.375 days
MTBF (SWO)	6.625 days
MIN TBF	3.35 h
MAX TBF	37 days
MTTR	5.12 h
Min TTR	0.58 h
Max TTR	12 h

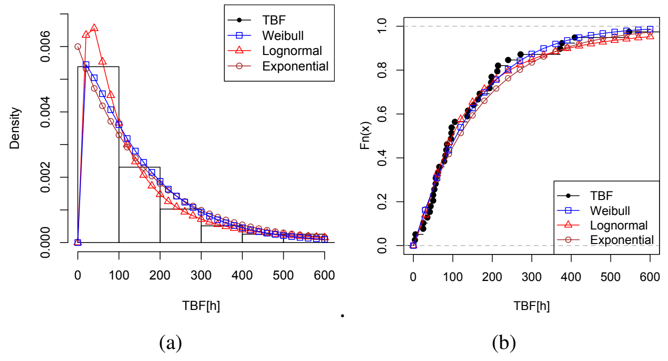


Fig. 5: Distribution fitting for time between SWOs: (a) PDF, (b) CDF

VI. Characterization of System-Wide Outages

In this section we analyze the causes and the time to repair for system-wide outages (*Interruption* failure category in Table III) in Blue Waters.

Basic Statistics. Table IX shows the statistics for 39 system-wide outages presented in the failure reports. On average, the system experienced a system-wide outage every 159.22 h. In 90% of those cases, the system was brought back to full capacity within 3.28 h from the start of the outage, with an MTTR of 5.12 h. The quick repair of system-wide outages contributed to a system availability of 0.9688.

TABLE X: Parameters estimated for the distributions in (Figure 5.

Distribution	G.O.F. (Kolmogorov)	Parameters
Lognormal	0.072	$\mu = 4.58, \sigma = 1.08$
Weibull	0.073	$\alpha = 1.07, \beta = 152$
Exponential	0.08	$\lambda = 0.006$

Figure 5 shows the PDF and CDF of the distribution of the times between SWOs. The exponential, Weibull, and lognormal distributions obtain an acceptable goodness of fit ($p-$value less than 0.1), as reported in Table X. Note that the good fit for the lognormal distribution indicates a hazard rate that first increases and then decreases ($\sigma > 1$), modeling the case in which a system-wide outage might depend on the preceding one. Interestingly, the reports document only one case in which a system-wide outage was attributed to the same cause as the preceding system-wide outage, which had occurred 3 h 51 min earlier (both outages were associated with Lustre), while the other cases were clearly unrelated. In addition, after each system-wide repair, the system is rebooted and cleared of remaining problems. That contributes to the good fit of the exponential distribution in Figure 5.

About 74.4% (29 out of 39) of the system-wide outages (SWOs) had software causes. Figure 6.(a) shows the breakdown of the system-wide outages across the three categories of causes: hardware, software, and environment. Hardware contributed to 15.4% of system-wide outages (6 out of 39), and environmental issues (power failures, in our case) caused 10.s% of system-wide outages (4 out of 39). Failures with network root causes brought down the system in 2 cases, i.e., 0.6% of the recorded network failures. However, as we will discuss in the next section, those failures resulted when the network became partitioned and was hence unable to route around failed nodes, because of a lack of redundant paths.

46% of SWOs were caused by Lustre failures (18 out of 39, about 62% of all software related SWOs). Figure 7 shows the types of Lustre failures that can escalate to system-wide outages. The key observations are i) that about 20% of Lustre failures (18 out of 104) cascade and manifest as system-wide outages, and ii) that for about half (47%) of them, a failure of the failover can be identified as a direct reason for the SWO.

About 0.7% (6 out of 831 cases) of single-node failures led to SWOs, because of an inability to reroute network traffic around the failed blades. Data show that node failures due to hardware problems become critical when they affect the Gemini routing or access to the Lustre file system. The full 3D torus topology is effective in providing a number of redundant paths, and the link degradation mechanisms and distributed routing tables ensure protection against transient errors, e.g., corruption of the routing table. The Gemini network uses dimension-ordered routing. In an error-free network, a packet first travels along the X dimension until it reaches the destination node's X coordinate. The packet then travels in the Y dimension to the destination node's Y coordinate. Finally, the packet moves in the Z dimension until it reaches the final destination. If the standard X-Y-Z ordering does not work, Gemini tries an alternative route until it finds a routable path.

Certain failure patterns, such as multiple errors in the same loop along the Z dimension, can leave Gemini unable to find a valid route. While such failures are rare, they require the longest time to recover from, with an MTTR of 8.1 h. The

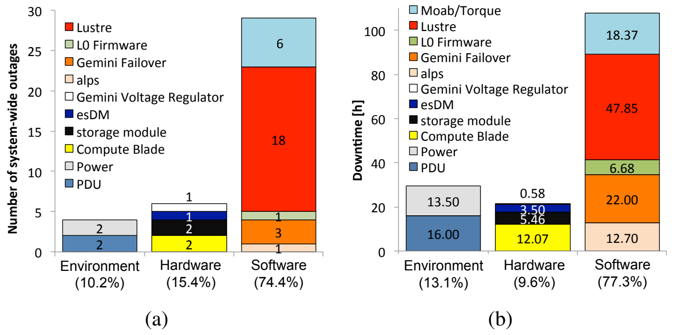


Fig. 6: Breakdown of the SWO root causes (a) and repair times (b)

failure reports contain three cases of system-wide outages due to routing problems: i) a mishap of the failover procedure during the warm swap of a failed service causing a system partition; ii) a partial failure of the Gemini network that did not trigger the link recovery procedure, causing all the jobs to abort because of lack of communication; and iii) a crash of the Gemini routing algorithm that caused the routes to hang. In the last case, while the Gemini crash's cause is unknown, system logs revealed a high rate of misrouted packets and routing data corruption within a 1 h time window before the incident.

VII. Related Work

Several studies have attempted to evaluate large-scale systems [1]–[8]. They have addressed one or more of the following issues: basic error characteristics [1], [2], [6], [8], modeling and evaluation [7], [20], [21], failure prediction and proactive checkpointing [10], [22], [23]. This study is the first to characterize the failures and related root causes of a sustained petascale system. Following a process similar to [4], we base our study on the manual failure reports produced by NCSA technical staff managing Blue Waters. In addition to the manual reports, we also use system logs to conduct deeper analysis and characterize the resiliency of the hardware protection mechanisms.

A number of researchers have classified failures in relation to their root cause, e.g., hardware and software [4], [24]. Hardware failures are identified in [24] as contributing 6% of the total failures whereas a 42% were due to software failures. Hardware is identified in [4] as the single largest cause of node failures, with percentage ranging from 30% to more than 60% with respect to the analyzed systems. Software percentages ranges from 5% to 24%. The Authors focus only on single node downtime and failure causes, and do not explicitly characterize system-wide

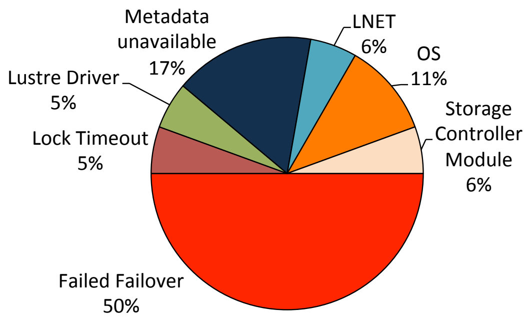


Fig. 7: Lustre Failures cascading in system-wide outages

outages. In our work we demonstrate that there has been a strong technological improvement in the hardware resiliency features by comparing the earlier generation with current HPC systems. Data show that the majority of system outages in Blue Waters are due to software errors (74.4%), while the hardware contributes only to single node/blade downtime node hours in 99.4% of documented failures. In addition, because of their important role in large-scale systems, we characterizes effectiveness of the failover and protection mechanisms at different scales, i.e., at the network, filesystem, processor and memory, and provide the first study to date measuring the rate of uncorrectable errors in the current generation of high-performance GPU accelerators.

VIII. Conclusions

This paper presents an in-depth study of manual failure reports collected over a period of 261 days for a large-scale computing system. This is the first failure study of a sustained petaflop system. The overall failure characterization indicates that failures due to software are the major contributor to the total repair time (53%), although they represent only 20% of the total number of failures. More importantly, software is also the primary cause (74.4%) of the SWOs. The analysis points out that the real system bottleneck is the inadequacy of mechanisms behind complex failover operations (e.g., timeout and distributed locks managers), such as those employed in the Lustre file system. Blue Waters is a configuration corner case on a scale well beyond that at which hardware and software have been methodically tested, so our findings have a significant value. To our surprise, hardware, regardless of its complexity, is highly resilient. In particular, use of error-correcting codes (including Chipkill) provides extremely high coverage (99.997%) for memory and processor errors, and hence high overall system resiliency, even in presence of high error rates (250 errors/h in our case). In the future, we plan to collect data from similar systems and conduct a detailed analysis of Lustre error resiliency at different scales. We also plan to look closely into automatically collected system error logs and workload data to obtain further insights into the impact of errors and failures on user computations.

Acknowledgements

This work is partially supported by the National Science Foundation grants CNS 10-18503 CISE, Air Force Research Laboratory and the Air Force Office of Scientific Research under agreement No. FA8750-11-2-0084, by a IBM faculty award and by a grant from Infosys Ltd.. This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (award number ACI 1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. The work of Fabio Baccanico has been supported by "Programma di scambi Internazionali con Università ed Istituti di Ricerca straniera per la mobilità di breve durata di docenti, ricercatori e studiosi" at Federico II University of Naples. We also acknowledge the anonymous reviewers and the Cray engineers for comments provided during the preparation of this paper and Jenny Applequist for the careful reading of this manuscript.

References

[1] R. K. Sahoo, A. Sivasubramaniam, M. S. Squillante, and Y. Zhang. Failure data analysis of a large-scale heterogeneous server environment. In *DSN '04: Proc. of the 2004 Int. Conference on Dependable Systems and Networks*, pages 772–781, 2004.

[2] Y. Liang, A. Sivasubramaniam, J. Moreira, Y. Zhang, R.K. Sahoo, and M. Jette. Filtering failure logs for a bluegene/l prototype. In *DSN '05: Proc. of the 2005 Int. Conference on Dependable Systems and Networks*, pages 476–485, 2005.

[3] Y. Liang, Y. Zhang, M. Jette, Anand Sivasubramaniam, and R. Sahoo. Bluegene/l failure analysis and prediction models. In *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, pages 425–434, 2006.

[4] B. Schroeder and G.A. Gibson. A large-scale study of failures in high-performance computing systems. *Dependable and Secure Computing, IEEE Transactions on*, 7(4):337–350, 2010.

[5] B. Schroeder and G. A. Gibson. Disk failures in the real world: what does an mttf of 1,000,000 hours mean to you? In *Proceedings of the 5th USENIX conference on File and Storage Technologies*, FAST '07, Berkeley, CA, USA, 2007. USENIX Association.

[6] A. Oliner and J. Stearley. What supercomputers say: A study of five system logs. *Dependable Systems and Networks, 2007. DSN '07. 37th Annual IEEE/IFIP Int. Conference on*, pages 575–584, June 2007.

[7] C. Di Martino, M. Cinque, and D. Cotroneo. Assessing time coalescence techniques for the analysis of supercomputer logs. In *In Proc. of 42nd Annual IEEE/IFIP Int. Conf. on Dependable Systems and Networks (DSN), 2012*, pages 1–12, 2012.

[8] A. Pecchia, d. Cotroneo, Z. Kalbarczyk, and R. K. Iyer. Improving log-based field failure data analysis of multi-node computing systems. In *Proceedings of the 2011 IEEE/IFIP 41st International Conference on Dependable Systems&Networks*, DSN '11, pages 97–108, Washington, DC, USA, 2011. IEEE Computer Society.

[9] B. Schroeder, E. Pinheiro, and W. Weber. Dram errors in the wild: a large-scale field study. *SIGMETRICS Perform. Eval. Rev.*, 37(1):193–204, June 2009.

[10] A. Gainaru, F. Cappello, M. Snir, and W. Kramer. Fault prediction under the microscope: A closer look into hpc systems. In *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, pages 1–11, 2012.

[11] AMD Inc. Bios and kernel developers guide, for amd family 16th.

[12] T. Dell. IBM Microelectronics Division. A white paper on the benefits of chipkill-correct ecc for pc server main memory, 1997.

[13] M. Karo, R. Lagerstrom, M. Kohnke, and C. Albing. The application level placement scheduler. In *Cray User Group - CUG*, 2008.

[14] http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-suite-enterprise-edition.

[15] http://www.cray.com/Products/Storage/Sonexion/Specifications.aspx.

[16] J. Stearley, R. Ballance, and L. Bauman. A state-machine approach to disambiguating supercomputer event logs. In *proc. of Workshop on Managing System Automatically and Dynamically 2, 155-192*, Berkeley, CA, 2012. USENIX.

[17] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi. Feng shui of supercomputer memory: Positional effects in dram and sram faults. In *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 22:1–22:11, New York, NY, USA, 2013. ACM.

[18] http://www.olcf.ornl.gov/titan/,number2ontop500.org.

[19] M. R. Lyu, editor. *Handbook of software reliability engineering*. McGraw-Hill, Hightstown, NJ, USA, 1996.

[20] E. Heien, D. Kondo, A. Gainaru, A. LaPine, W. Kramer, and F. Cappello. Modeling and tolerating heterogeneous failures in large parallel systems. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 45:1–45:11, New York, NY, USA, 2011. ACM.

[21] C. Di Martino. One size does not fit all: Clustering supercomputer failures using a multiple time window approach. In JulianMartin Kunkel, Thomas Ludwig, and HansWerner Meuer, editors, *International Supercomputing Conference - Supercomputing*, volume 7905 of *Lecture Notes in Computer Science*, pages 302–316. Springer Berlin Heidelberg, 2013.

[22] Xin Chen, Charng-Da Lu, and K. Pattabiraman. Predicting job completion times using system logs in supercomputing clusters. In *Dependable Systems and Networks Workshop (DSN-W), 2013 43rd Annual IEEE/IFIP Conference on*, pages 1–8, June 2013.

[23] A. Gainaru, F. Cappello, and W. Kramer. Taming of the shrew: Modeling the normal and faulty behaviour of large-scale hpc systems. In *Parallel Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*, pages 1168–1179, 2012.

[24] D. Oppenheimer and D. A. Patterson. Studying and using failure data from large-scale internet services. In *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, EW 10, pages 255–258, New York, NY, USA, 2002. ACM.