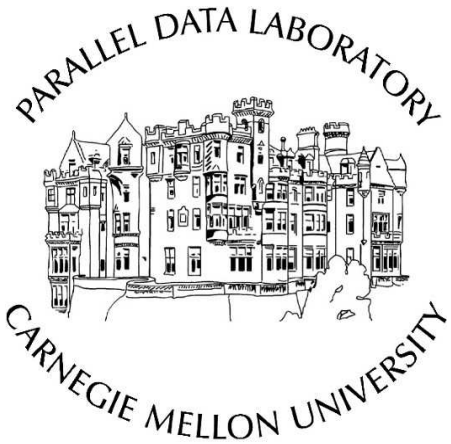

A large-scale study of failures in high-performance-computing systems

Bianca Schroeder, Garth A. Gibson

CMU-PDL-05-112
December 2005

Parallel Data Laboratory
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

Designing highly dependable systems requires a good understanding of failure characteristics. Unfortunately little raw data on failures in large IT installations is publicly available, due to the confidential nature of this data. This paper analyzes soon-to-be-public failure data covering systems at a large high-performance-computing site. The data has been collected over the past 9 years at Los Alamos National Laboratory and includes 23000 failures recorded on more than 20 different systems, mostly large clusters of SMP and NUMA nodes. We study the statistics of the data, including the root cause of failures, the mean time between failures, and the mean time to repair. We find for example that average failure rates differ wildly across systems, ranging from 20-1000 failures per year, and that time between failures is modeled well by a Weibull distribution with decreasing hazard rate. From one system to another, mean repair time varies from less than an hour to more than a day, and repair times are well modeled by a lognormal distribution.

Acknowledgements: We thank the members and companies of the PDL Consortium (including APC, EMC, Equallogic, Hewlett-Packard, Hitachi, IBM, Intel, Microsoft, Network Appliance, Oracle, Panasas, Seagate, and Sun) for their interest, insights, feedback, and support.

Keywords: Failure data, Lifetime data, Root cause, Repair time, High performance computing.

1 Introduction

Research in the area of dependable computing relies in many ways on a thorough understanding of what failures in real systems look like. For example, knowledge of failure characteristics can be used in resource allocation to improve cluster availability [3, 18]. The design and analysis of checkpoint strategies relies on certain statistical properties of failures [6, 15, 16]. Creating realistic benchmarks and testbeds for reliability testing requires an understanding of the characteristics of real failures.

Unfortunately, obtaining access to failure data from modern, large-scale systems is difficult, since such data is often sensitive or classified. Existing studies of failures are often based on only a few months of data, covering typically only a few hundred failures [14, 17, 12, 13, 11, 5]. Many of the commonly cited studies on failure analysis stem from the late 80's and early 90's, when computer systems where significantly different from today [1, 2, 4, 10, 14, 7, 8]. Finally, none of the raw data used in the above studies has been made publicly available for use by other researchers.

This paper foreshadows the public release of a large set of failure data. The data was collected over the past 9 years at Los Alamos National Labs (LANL) and covers 22 high-performance-computing (HPC) systems used at the site, adding up to a total of 4750 machines with 24101 processors. The data contains an entry for any failure that occurred during the 9-year time period and that required the attention of a system administrator. For each failure the data includes start time and end time, the system and node affected, as well as categorized root cause information. To the best of our knowledge this is the largest set of failure data studied in the literature to date, both in terms of the time-period it spans, and the number of systems and processors it covers.

The goal of this paper is to provide a description of the statistical properties of the data, as well as information for other researchers on how to interpret the data. We first describe the environment the data comes from, including the systems and the workloads, the process used to collect the data, and the structure of the data records (Section 2). Section 3 describes the methodology we use in the data analysis. We then study the data with respect to three important properties of system failures: the root causes (Section 4), the time between failures (Section 5) and the time to repair (Section 6). Section 7 compares our results to related work. Section 8 summarizes and concludes.

2 Description of the data and environment

To provide the necessary context for analyzing the data, we describe below the systems the data was collected on, the workloads those systems are running, and the process used to collect the data.

2.1 The systems

The data spans 22 high-performance-computing systems that have been in production use at LANL between 1996 and November 2005. Most of these systems are large clusters of either NUMA nodes, or 2-way and 4-way SMP nodes. In total the systems add up to 4750 nodes and 24101 processors. Table 1 gives an overview description of the 22 systems.

The left half of Table 1 provides high-level information for each of the 22 systems, including the hardware architecture (NUMA vs SMP), the total number of nodes and processors in the system, and a system ID used throughout this paper to refer to a particular system. Unfortunately, we are not able to release vendor specific information on the hardware used in each system. Instead we label system types using capital letters, such that two systems have the same label when they use identical processor and memory chip models. We refer to a system's label as its *hardware type*.

As the table shows, the LANL site has hosted a very diverse set of systems. Systems vary widely in size with the number of nodes ranging from 1 to 1024 and the number of processors ranging from 4 to

(I) High-level system information					(II) Information per node category				
Archi-tecture	Hardw. Type	ID	#Nodes	#Procs	#Procs per node	Commiss. Date	Production Time	Mem (GB)	#NICs
SMP	A	1	1	8	8	N/A	N/A - Dec-99	16	0
	B	2	1	32	32	N/A	N/A - Dec-03	8	1
	C	3	1	4	4	N/A	N/A - Apr-03	1	0
	D	4	164	328	2	Mar-01	Apr-01 - now	1	1
					2	Dec-02	Dec-02 - now	1	1
	E	5	256	1024	4	Oct-01	Dec-01 - now	16	2
		6	128	512	4	Aug-01	Sep-01 - Jan-02	16	2
		7	1024	4096	4	Mar-02	May-02 - now	8	2
					4	Mar-02	May-02 - now	16	2
					4	Mar-02	May-02 - now	32	2
					4	Mar-02	May-02 - now	352	2
		8	1024	4096	4	Aug-02	Oct-02 - now	8	2
					4	Aug-02	Oct-02 - now	16	2
					4	Aug-02	Oct-02 - now	32	2
		9	128	512	4	Aug-03	Sep-03 - now	4	1
		10	128	512	4	Aug-03	Sep-03 - now	4	1
		11	128	512	4	Aug-03	Sep-03 - now	4	1
		12	32	128	4	Aug-03	Sep-03 - now	4	1
					4	Aug-03	Sep-03 - now	16	1
	F	13	128	256	2	Aug-03	Sep-03 - now	4	1
		14	256	512	2	Aug-03	Sep-03 - now	4	1
		15	256	512	2	Aug-03	Sep-03 - now	4	1
		16	256	512	2	Aug-03	Sep-03 - now	4	1
		17	256	512	2	Aug-03	Sep-03 - now	4	1
		18	512	1024	2	Aug-03	Sep-03 - now	4	1
					2	Mar-05	Mar-05 - Jun-05	4	1
NUMA	G	19	16	2048	128	Oct-96	Dec-96 - Sep-02	32	4
					128	Oct-96	Dec-96 - Sep-02	64	4
		20	49	6152	128	Nov-96	Jan-97 - now	128	12
					128	Nov-96	Jan-97 - Nov-05	32	12
					80	Apr-05	Jun-05 - now	80	0
		21	5	544	128	Oct-98	Oct-98 - Dec-04	128	4
					32	Jan-98	Jan-98 - Dec-04	16	4
					128	Nov-02	Nov-02 - now	64	4
					128	Nov-02	Nov-05 - Dec-04	32	4
	H	22	1	265	256	Nov-04	Nov-04 - now	1024	0

Table 1: *Overview of systems*

6152. Systems also vary in their hardware architecture. There is a large number of NUMA and SMP based machines, and a total of eight different processor and memory models (types A-H).

The nodes in a cluster system are not always identical. While all the nodes in a system have the same hardware type, they might differ in the number of processors and network interfaces (NICs), the amount of main memory, or in their commission/decommission dates. The right half of Table 1 categorizes the nodes in a system with respect to these properties. For example, the nodes of system 12 fall into two categories, differing only in the amount of memory per node (4 vs 16 GB).

While the table includes information on the time a system was commissioned and the time it actually went into production, the data includes failure records only for the production time.

2.2 The workloads

The majority of the workloads are large-scale scientific simulations, such as simulations of nuclear stockpile stability. These applications perform long periods (often months) of CPU computation, interrupted every few hours by a few minutes of I/O for check-pointing. Simulation workloads are often accompanied by scientific visualization of large-scale data. Visualization workloads are also CPU-intensive, but exhibit more reading of data from storage than compute workloads. Finally, some nodes are used purely as front-end nodes, and others run more than one type of workload, for instance, graphics nodes often run compute workloads as well.

At LANL failure tolerance is frequently implemented through periodic check-pointing. When a node fails, the job(s) running on it is stopped and restarted on a different set of nodes, either starting from the most recent checkpoint or from scratch if no check-point exists.

Starttime	Endtime	System	Node	Type of Node	Root cause	
					High-level	Detailed
...	...	...	...	...	...	...
2005-6-21 10:54	2005-6-21 11:00	6	1	graphics.fe	Hardware	Cooling Fan
2005-6-21 15:54	2005-6-21 16:41	10	93	compute	Software	OS Software
...	...	...	...	...	...	...

Table 2: *Sample failure records.*

2.3 Data collection

The data is based on a "remedy" database created at LANL in June 1996. At that time, LANL introduced a site-wide policy that for any system failure requiring the intervention of a system administrator a record describing the failure has to be entered into the remedy database. Consequentially, the database today contains a record for every failure that occurred in LANL's HPC systems since June 1996 and that required intervention of a system administrator.

Table 2 shows some sample records. A failure record contains the time when the failure started (Starttime), the time when it was resolved (Endtime), the system and node affected, the type of workload running on the node and the root cause. The workload is either *compute* for computational workloads, *graphics* for visualization workloads, or *fe* for front-end. Root causes fall in one of the following five high-level categories: *Human* error; *Environment*, including for example power outages or A/C failures; *Network* failure; *Software* failure; and *Hardware* failure. In addition, more detailed information on the root cause is captured, such as the particular hardware component affected by a *Hardware* failure. The failure classification and rules for assigning failures to categories were developed jointly by hardware engineers, administrators and operations staff at LANL.

Failure reporting at LANL follows the following protocol. Failures are detected by an automated monitoring system that pages operations staff whenever a node is down. The operations staff then create a failure record in the database specifying the Starttime of the failure, and the system and node affected, then turn the node over to a system administrator for repair. Upon repair, the system administrator notifies the operations staff who then put the node back into the job mix and fill in the Endtime of the failure record. If the system administrator was able to identify the root cause of the problem he provides operations staff with the appropriate information for the "root cause" field of the failure record. Otherwise the root cause is specified as "Unknown". Operations staff and system administrators have occasionally follow-up meetings for failures with "Unknown" root cause. If through those meetings or other ways the root cause becomes clear later on, the corresponding failure record gets amended accordingly.

Two implications follow from the way the data was collected. First, this data is very different from the error logs used in many other studies. Error logs are automatically generated and track any exceptional events in the system, not only errors resulting in system failure. Moreover, error logs often contain multiple entries for the same error event.

Second, since the data was created manually by system administrators, the data quality depends on the accuracy of the administrators' reporting. Two potential problems in human created failure data are underreporting of failure events and misreporting of root cause. For the LANL data we don't consider underreporting (i.e. a failure does not get reported at all) a serious concern, since failure detection is initiated by automatic monitoring and failure reporting involves several people from different administrative domains (operations staff and system administrators). While misdiagnosis can never be ruled out completely, its frequency depends on the skills of the system administrator. LANL employs highly-trained staff backed by a well-funded cutting edge technology integration team, often pulling new technology into existence in collaboration with vendors.

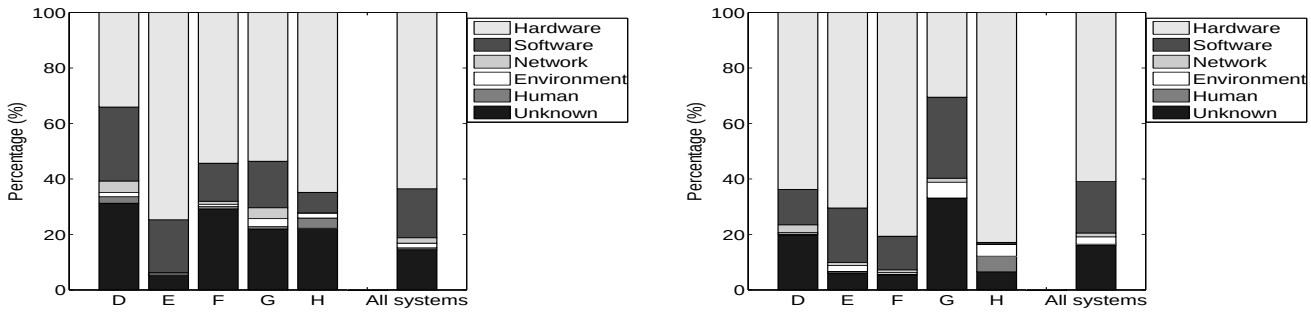

Figure 1: *The breakdown of failures into root causes (left) and the breakdown of downtime into root causes (right). Each graph shows the breakdown for systems of type D, E, F, G, and H and aggregate statistics across all systems (A-H).*

3 Methodology

We characterize an empirical distribution using three import metrics: the mean, the median, and the squared coefficient of variation (C^2). The squared coefficient of variation is a measure of the variability of a distribution and is defined as the squared standard deviation divided by the squared mean. The advantage of using the squared coefficient of variation as a measure of variability, rather than the variance or the standard deviation, is that it is normalized by the mean, and hence allows comparison of variability across distributions with different means.

We also consider the empirical cumulative distribution function (CDF) and how well it is fit by four probability distributions commonly used in reliability theory[1]: the exponential distribution; the Weibull distribution; the gamma distribution; and the lognormal distribution. We parameterize the distributions through maximum likelihood estimation and evaluate the goodness of fit both by visual inspection and the negative log-likelihood test.

Note that the goodness of fit that a distribution achieves depends on the degrees of freedom that this distribution offers. For example, a phase-type distribution with an arbitrary number of phases would likely give a better fit than any of the above standard distributions, which are limited to one or two parameters. Whenever the quality of fit allows, we prefer the simplest standard distribution because these are well understood, simple to use and can be generated and fit efficiently. In our analysis of this data we have so far not found any reason to depend on more degrees of freedom.

4 Root cause breakdown

An obvious question when studying failures in computer systems is what caused the failures. In this section we study the root causes as reported in the root cause field of the data.

We first look at the relative frequency of the six high-level categories for root cause: human, environment, network, software, hardware, and unknown. Figure 1 (left) shows the percentage of failures in each of the six categories. The right-most bar describes the breakdown across all failure records in the data set. Each of the five bars to the left presents the breakdown across all failure records for systems of a particular hardware type.

Figure 1 indicates that while the actual breakdown varies across systems with different hardware type,

[1]We also considered another distribution, which has recently been found to be useful in characterizing various aspects of computer systems, the Pareto distribution. However, we didn't find it to be a better fit than any of the four standard distributions for our data and therefore did not include it in these results.

the basic trends are similar. Hardware is the single largest component, with the actual percentage ranging from 30% to more than 60%. Software is the second largest contributor in all cases, with percentages ranging from 5% to 24%. Systems of type D differ most from the other systems, in that hardware and software are almost equally frequently reported as root cause.

It is important to observe that in most systems the root cause remained undetermined for 20-30% of the failures (except for type E systems, where less then 5% of root causes are unknown). Since the fraction of hardware failures is in all systems larger than the fraction of undetermined failures, and the fraction of software failures is close to that of undetermined failures, we can still conclude that hardware and software are among the largest contributors to failures. However, we can not conclude that any of the other failure sources (Human, Environment, Network) is insignificant.

In addition to the relative frequency of the different root causes, we also consider how much each of them contributes to the total downtime. Figure 1 (right) shows the total downtime per hardware type broken down into the root cause that caused the downtime. The basic trends are similar to the root cause breakdown by frequency: hardware tends to be the single largest component, followed by software. Interestingly, for most systems the failures with unknown root cause account for less than 5% of the total downtime, despite the fact that the percentage of unknown root causes is higher. Only systems of type D and G have more than 5% of downtime with unknown root cause.

The reason for the higher fraction of downtime with unknown root cause for systems of type D and G lies in the circumstances surrounding their initial deployment. Systems of type G were the first NUMA based clusters at the site and were commissioned at a time when LANL just started to systematically record failure data. As a result in the beginning of those systems' lifetime the fraction of root causes that remained undetermined was very high ($> 90\%$), but dropped to less than 10% within 2 years, as administrators' gained more experience with the new system and the root cause analysis involved with it. Similarly, the system of type D was the first large-scale SMP cluster at LANL, so initially the number of failures with unknown root cause was high, but then quickly dropped.

In addition to the breakdown of the root cause into the five high-level categories, we also looked at the more detailed failure categorization. We find that for all systems, memory related hardware failures make up a significant portion of all failures. For all systems more than 10% of all failures (not only hardware failures) were due to memory, in the case of system F and H even more than 25%. In fact, for all systems, except for system E, memory was the single most common "low-level" root cause. System E experienced a very high percentage (more than 50%) of CPU related failures, due to a major flaw in the design of the type of CPU used in systems of type E.

The detailed breakdown for software related failures varies more across systems. For system F the most common software failure was related to the parallel file system, for system H to the scheduler software and for system E to the operating system. For system D and G a large portion of the software failures were not specified further.

5 Analysis of failure rates

5.1 Failure rate as a function of system and node

This section looks at how failure rates vary across different systems, and across the nodes within the same system. Studying failure rates across different systems is interesting since it provides insights on the effect of parameters such as system size and hardware type. Knowledge on how failure rates vary across the nodes in a system can be utilized in job scheduling, for instance by assigning critical jobs or jobs with high recovery time to more reliable nodes.

Figure 2 (left) shows for each of the 22 systems the average number of failures recorded per year during the system's production time. The yearly failure rate varies widely across systems, ranging from only 17

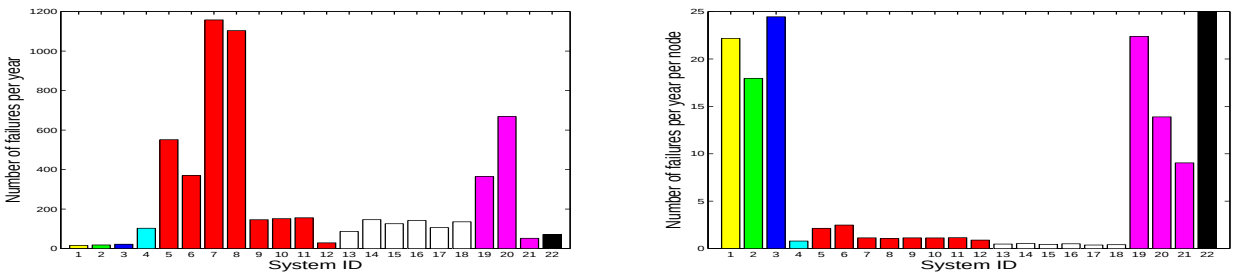

Figure 2: *Average number of failures for each system per year (left). Average number of failures for each system per year normalized by number of nodes in the system (right). Bars of systems that use the same hardware type have the same color.*

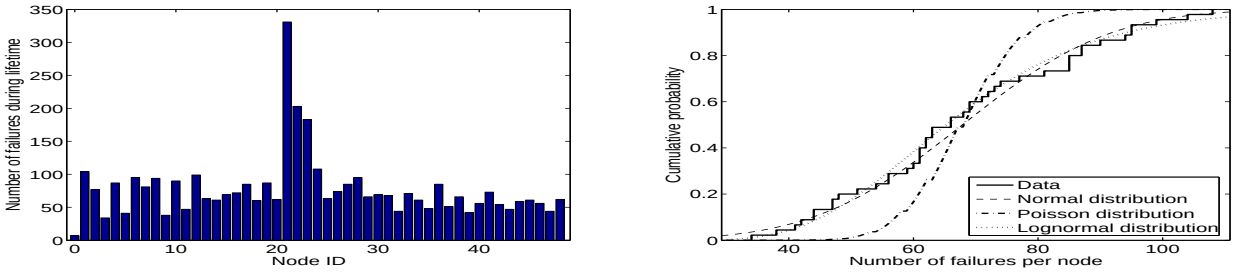

Figure 3: *Number of failures per node for system 20 as a function of node ID (left) and the corresponding CDF, fitted with a Poisson, normal and lognormal distribution (right).*

failures per year for system 2, to an average of 1159 failures per year for system 7. In fact, variability in the failure rate is high even among systems of the same hardware type.

The main reason for the vast differences in failure rate across systems is that the systems vary widely in size. Figure 2 (right), shows for each system the average number of failures per year normalized by the number of nodes in the system. The normalized failure rates have relatively small variability for systems of the same hardware For example, all systems of type E (systems 5-12) exhibit a similar average failure rate per year per node [2], despite the fact that they range in size from 128 to 1024 nodes. The same holds for type F systems. This indicates that failure rates don't grow significantly faster than linearly with the number of nodes in a system.

We next concentrate on the distribution of failures over the nodes within a given system. Figure 3 shows the total number of failures for each node of system 20 during the entire lifetime of the system. We find a relatively uniform distribution of failures across nodes, except for nodes 21-23, which experienced a significantly higher number of failures than the other nodes. While nodes 21-23 make up only 6% of all nodes, they account for 20% of all failures in the system.

One reason for the above non-uniformity might be that nodes 21-23 differ from the other nodes in the workloads they run. Node 21-23 are the only nodes used for visualization, as well as computation, resulting in a more varied and interactive workload compared to the other nodes. We make similar observations for other systems, where failure rates vary significantly depending on a node's workload. For example, for systems E and F, the front-end nodes, which run a more varied, interactive workload, exhibit a much higher failure rate than the other nodes in the same system.

While it seems clear from Figure 3 that the graphics nodes have a very different behavior from the other nodes, a remaining question is how similar the failure rates of the remaining (compute-only) nodes

[2]Except for system 6, which is an unusual case in that it was in production for only 5 months before it was decommissioned.

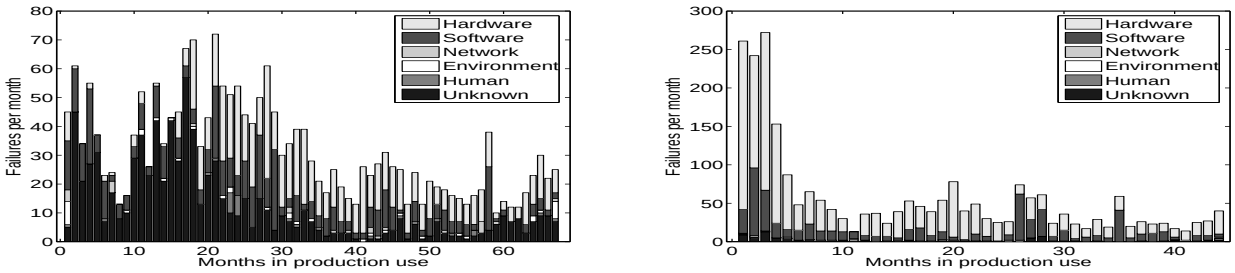

Figure 4: *Two representative examples for how the failure rate changes as a function of system age (in months). The curve on the left corresponds to system 19 which is representative for systems of type D and G. The curve on the right corresponds to system 5 which is representative for systems of type E and F.*

are to each other. Figure 3 (right) shows the CDF of the measured number of failures per node for compute nodes, with three different distributions fitted to it: the Poisson distribution, the normal distribution, and the lognormal distribution. If the failure rate at all nodes followed a Poisson process with the same mean (as often assumed e.g. in work on check-pointing protocols), the distribution of failures across nodes would be expected to match a Poisson distribution. Instead we find that the Poisson distribution is a poor fit, mostly because the measured data seems to have a higher variability than that of the Poisson fit. The normal and lognormal distribution are a much better fit, both visually as well as measured by the negative log-likelihood. This indicates that the assumption of Poisson failure rates with equal means across nodes is not likely to be realistic.

5.2 Failure rate at different time scales

This section looks at how failure rates vary across different time scales, from very large (system lifetime) to very short (daily and weekly). Knowing how failure rates vary as a function of time is important for generating realistic failure workloads and for optimizing recovery mechanisms.

We begin with the largest possible time-scale by looking at failure rates over the entire lifetime of a system. We find that for all systems in our data set the curve for the failure rate as a function of system age follows one of two shapes. Figure 4 shows a representative example for each shape.

Figure 4 (right) shows the number of failures per month for system 5, starting at production time. The basic characteristic of the curve is that failure rates are high initially, and then drop significantly during the first 3-4 months of production use. The shape of this curve is the most common one and is representative of all systems of type E and F.

The shape of this curve is intuitive in that the failure rate drops during the early age of a system, as initial hardware and software bugs are detected and fixed and administrators gain experience in running the system. One might wonder why the initial problems were not solved during the 1-2 months of testing before production time. The reason is that many problems in hardware, software and configuration are only exposed by real user code in the production workloads.

The curve in Figure 4 (left) corresponds to the failures observed over the lifetime of system 19 and represents the other commonly observed shape. The shape of this curve is representative for systems of type D and G, and is less intuitive: The failure rate actually grows over a period of nearly 20 months, before it eventually starts dropping. One possible explanation for this behavior is that getting these systems into full production was a slow and painful process.

The systems of type G were the first systems of the NUMA era at LANL and the first systems ever that arranged such a large number of NUMA machines in a cluster. As a result the first 2 years still involved a lot of development work among the administrators of the system, the vendors, and the users. Administrators

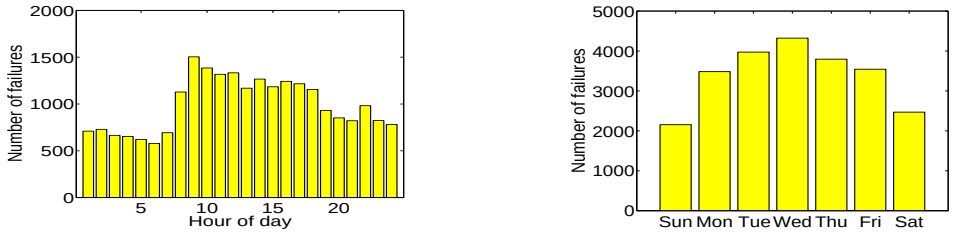

Figure 5: *Number of failures by hour of the day (left) and the day of the week (right).*

had to develop new software for managing the system and providing the infrastructure to run large parallel applications. Users developed new large-scale applications that wouldn't have been feasible to run on any of the previous systems. With the slower development process it took longer until the systems were running the full variety of production workloads and the majority of the initial bugs were exposed and fixed. The case for the system of type D was similar in that it was the first large-scale SMP cluster at the site.

Our above explanation for the failure rate curves of systems of type D and G is supported by two other observations. First, the failure rate curve for other SMP clusters (systems of type E and F) that were introduced later and were running full production workloads earlier in their life follows the more traditional pattern in Figure 4 (right). Second, system 21, which was introduced 2 years after the two other systems of type G, exhibits a failure rate curve much closer to Fig 4 (right).

Next we look at how failure rates vary over smaller time scales. It is well known that usage patterns of systems vary with the time of the day and the day of the week. The question we are interested in is whether there are similar patterns for failure rates. Figure 5 categorizes all failures in the data by hour of the day (left) and by day of the week (right). We observe a strong correlation in both cases. During peak hours of the day the failure rate is two times higher than during the night. Similarly the failure rate during weekdays is nearly two times as high as during the weekend. We interpret this as a correlation between the failure rate of a system and the workload run on the system, since we know based on general usage patterns (not specifically LANL) that the workload intensity and the variety of workloads is lower during the night and weekend.

Note that another possible explanation for the observations in Figure 5 would be that failure rates during the night and weekends are not lower, but that the detection of those failures is simply delayed until the beginning of the next (week-)day. We consider this explanation less likely, since failures are detected automatically by a monitoring system, and not by users or system administrators. Moreover, if delayed detection was the reason, one would expect a large peak on Mondays, and lower failure rates on the following days, which is not the case in this data.

5.3 Statistical properties of time between failures

In this section we view the sequence of failure events as a stochastic process and study the distribution of its inter-arrival times, i.e. the time between failures. We take two different views of the failure process: (i) the view as seen by an individual node, i.e. we study the time between failures that affect only this particular node; (ii) and the view as seen by the whole system, i.e. we study the time between subsequent failures that affect any node in the system.

Since failure rates vary across the lifetime of a system (recall Figure 4), the time between failures also varies accordingly. We therefore analyze the time between failures separately for the early production time during which failure rates are high and the remaining life of the system when failure rates have come down.

We begin with the view of the time between failures as seen by an individual node. Figure 6 shows the corresponding empirical distribution as seen by node 22 in system 20 during the years 1996-1999 (left) and

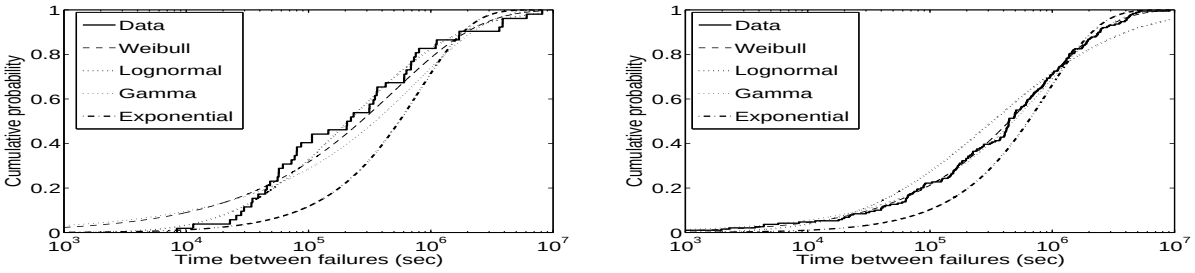

Figure 6: *Empirical CDF for inter-arrival times of failures on node 22 in system 20 early in production (left) and late in production (right).*

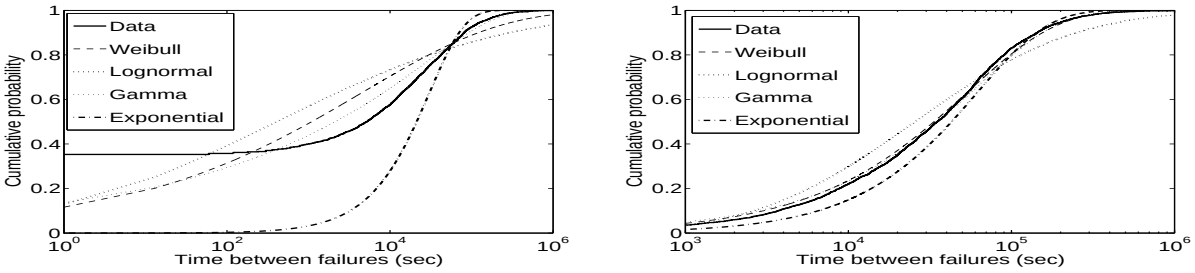

Figure 7: *Empirical CDF for inter-arrival times of failures for the system wide view of failures in system 20 early in production (left) and late in production (right).*

the years 2000-2005 (right), fitted by four standard distributions. We observe that during years 2000-2005 the distribution between failures is well modeled by a Weibull or gamma distribution. Both distributions create an equally good visual fit and the same negative log-likelihood. The simpler exponential distribution is a poor fit, as its C^2 of 1 is lower than the data's C^2 of 1.9.

For a given failure interarrival distribution it is useful to know how the time since the last failure influences the expected time until the next failure. This notion is captured by a distribution's hazard rate function. An increasing hazard rate function predicts that if the time since a failure is long then the next failure is coming soon. And a decreasing hazard rate function predicts the reverse. Figure 6 (right) is well fit by a Weibull distribution with shape parameter 0.7, indicating that the hazard rate function is decreasing, i.e. not seeing a failure for a long time decreases the chance of seeing one in the near future.

During years 1997-1999 the empirical distribution of the time between failures at node 22 looks quite different (Figure 6 (left)) from the 2000-2005 period. During this time period the best fit is provided by the lognormal distribution, followed by the Weibull and the gamma distribution. The exponential distribution is an even poorer fit here than it was during the second half of the node's lifetime. The reason lies in the higher variability of the time between failures with a C^2 of 3.9. This high variability might not be surprising given the variability in monthly failure rates we observed in Figure 4 for systems of this type during this time period.

Next we move to the system wide view of the failures in system 20, shown in Figure 7. The basic trend for the years 2000-05 (Figure 7 (right)) is similar to the per node view during the same time. The Weibull and gamma distribution are the best fit, while the lognormal and exponential fits are significantly worse. Again the hazard rate function is decreasing (Weibull shape parameter of 0.78).

The system wide view during years 1997-1999 (Figure 7 (left)) exhibits a distribution that is very different from the others we have seen and is not well captured by any of the standard distributions. The reason is an exceptionally large number ($> 30\%$) of inter-arrival times that are zero, indicating that two

	Unknown	Human	Env.	Netw.	Softw.	Hardw.	All
Mean (min)	398	163	572	247	369	342	355
Median (min)	32	44	269	70	33	64	54
Std. Dev. (min)	6099	418	808	720	6316	4202	4854
Variability (C^2)	234	6	2	8	293	151	187

Table 3: *Statistical properties of time to repair as a function of the root cause of the failure.*

failures occurred at the same time. Since for a given node there are never two failures recorded at the same time, a zero inter-arrival time is caused by simultaneous failures in two different nodes. While we did not perform a rigorous analysis of correlations between nodes[3], this high number of simultaneous failures indicates the existence of a correlation.

6 Analysis of repair times

This section considers a second important metric in system reliability, the time to repair. We first study how parameters such as the root cause of a failure and system parameters affect repair times. We then study the statistical properties of repair times, including their distribution and variability.

Table 3 shows the median and mean of time to repair as a function of the root cause, and as an aggregate across all failure records. We find that both the median and the mean time to repair vary significantly depending on the root cause of the failure. The mean time to repair ranges from less than 3 hours for failures caused by human error, to nearly 10 hours for failures due to environmental problems. The mean time to repair for the other root cause categories varies between 4 and 6 hours. In comparison, the mean repair time across all failures (independent of root cause) is close to 6 hours. The reason is that it's dominated by hardware and software failures which are the most frequent types of failures and exhibit mean repair times around 6 hours.

An important observation is that the time to repair for all types of failures is extremely variable, except for environmental problems. For example in the case of software failures the median time to repair is about 10 times lower than the mean, and in the case of hardware failures it is 4 times lower than the mean. This high variability is also reflected in extremely large C^2 values, as shown in the bottom row of Table 3.

One explanation for the extreme variability in the repair times of software and hardware failures might be the diverse set of problems that can cause these failures. For example, the root cause information for hardware failures spans 99 different sub-categories, compared to only two (power outage and A/C failure) for environmental problems. To test the validity of this explanation we determined the C^2 for several particular types of hardware problems. We find that even within one type of hardware problem the variability can be high. For example, the C^2 for repair times of CPU, memory, and node interconnect problems is 36, 87, and 154, respectively. This leads us to conclude that there are other factors contributing to the high variability.

Figure 8 (left) shows the empirical CDF for the repair times across all failures in the data, as well as four standard distributions fitted to the data. The exponential distribution is a very poor fit to the data, which is not surprising given the high variability in the repair times. Among all distributions the lognormal distribution is the best fit, both visually as well as measured by the negative log-likelihood. The Weibull distribution and the gamma distribution are weaker fits than the lognormal distribution, but still considerably better than the exponential distribution.

Finally, we consider how repair times vary for different systems. Figure 9 shows the mean and median time to repair for each of the 22 systems. The figure indicates that the hardware type has a major effect on

[3]The reason is that simply computing the cross-correlation over the data is not sufficient, since other correlations, e.g. with time of the day, would blur the results.

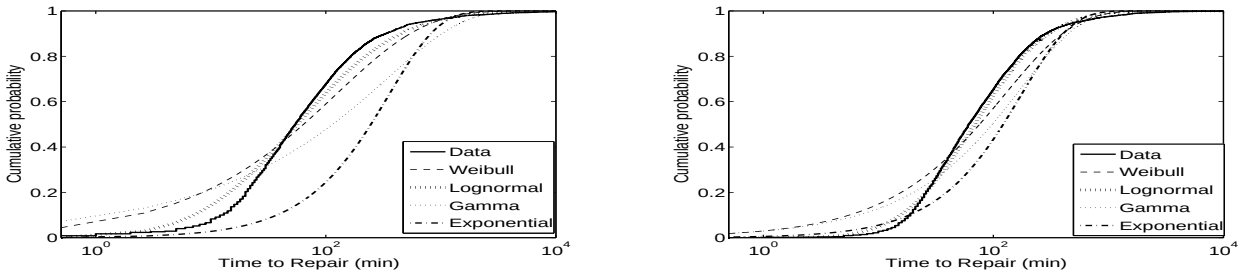

Figure 8: *Empirical CDF of repair times across all systems (left) and for systems of type E (right).*

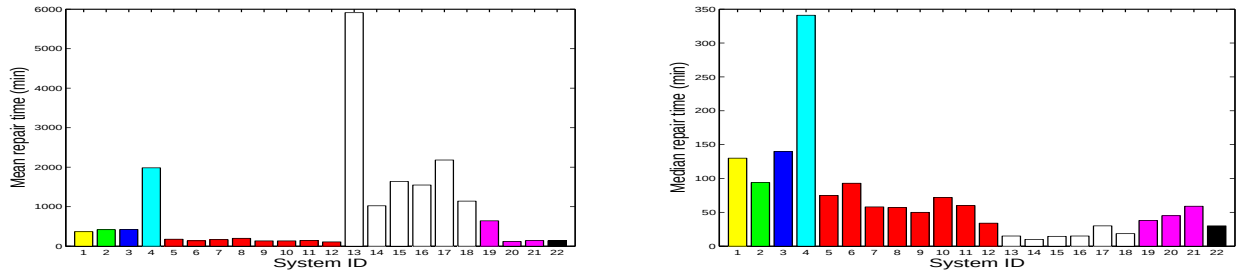

Figure 9: *Mean repair time for each system (left) and median repair time for each system (right).*

repair times. While systems of the same hardware type exhibit similar mean and median time to repair [4], repair times vary significantly across systems of different type,

Figure 9 also indicates that system size is not a significant factor in repair time. For example, type E systems range from 128 to 1024 nodes, but still exhibit similar repair times. In fact, systems 7 and 8, the largest systems of type E, are among the ones with the lowest median repair time.

The relatively consistent repair times across systems of the same hardware type are also reflected in the empirical CDF. Figure 8 (right) shows the CDF for the repair times of all systems of type E. The CDF is less variable than that taken across all systems (compare with Figure 8 (left)) which results in an improved (albeit still sub-optimal) fit by the exponential distribution.

7 Comparison with related work

Work on characterizing failures in computer systems falls in different categories depending on the type of data used; the type and number of systems under study; the time of data collection; and the number of failure or error records in the data set. Table 4 gives an overview of several commonly cited studies of failure data.

Four of the above studies include root cause statistics [2, 10, 12, 5]. The percentage of software-related failures is reported to be around 20% [1, 10, 12] to 50% [2, 5]. Hardware is reported to make up 10-30% of all failures [2, 10, 12, 5]. Environment problems are reported to account for around 5% [2]. Network problems are reported to make up between 20% [12] and 40% [5]. Gray [2] reports 10-15% of problems due to human error, while Oppenheimer et al. [12] report 14-30%. The main difference to our results is the lower percentage of human error and network problems in our work. There are two possible explanations. First, the root cause of 20-30% of failures in our data is unknown and could lie in the human or network category. Second, the LANL environment is an expensive and very controlled environment with national safety obligations and priorities, in which greater effort may be put into these parts of the infrastructure than in commercial environments.

[4] With the exception of system 13, which experienced three unusual months-long downtimes

Study	Date	Length	Environment	Type of Data	# Failures	Statistics
[1, 2]	1990	3 years	Tandem systems	Customer data	800	Root cause
[5]	1999	6 months	70 Windows NT mail server	Error logs	1100	Root cause
[12]	2003	3-6 months	3000 machines in Internet services	Error logs	501	Root cause
[10]	1995	7 years	VAX systems	Field data	N/A	Root cause
[14]	1990	8 months	7 VAX systems	Error logs	364	TBF
[7]	1990	22 months	13 VICE file servers	Error logs	300	TBF
[4]	1986	3 years	2 IBM 370/169 mainframes	Error logs	456	TBF
[13]	2004	1 year	395 nodes in machine room	Error logs	1285	TBF
[3]	2002	1-36 months	70 nodes in university and Internet services	Error logs	3200	TBF
[17]	1999	4 months	503 nodes in corporate envr.	Error logs	2127	TBF

Table 4: *Overview of related studies*

Several studies analyze the time between failures [13, 14, 3, 17]. Three of the studies use distribution fitting and find the Weibull distribution to be a good fit [3, 17, 7], which agrees with our results. All four studies looked at the hazard rate function, but come to different conclusions. Three of them [3, 17, 7] find decreasing hazard rates (Weibull shape parameter < 0.5). Others find that hazard rates are flat [14], or increasing [13]. We find decreasing hazard rates with Weibull shape parameter of 0.7-0.8.

Two other studies [4, 13] report correlation between workload and failure rate. Sahoo et al. [13] conclude that there is a correlation between the type of workload and the failure rate, while Iyer et al. report a correlation between the workload intensity (CPU utilization) and the failure rate. We find evidence for both correlations, in that we observe different failure rates for compute, graphics, and front-end nodes, and different failure rates for different hours of the day and days of the week.

Sahoo et al.[13] also study the correlations of failure rate with hour of the day and the distribution of failures across cluster nodes and find even stronger correlations than we do. They report that less than 4% of the nodes in a machine room experience almost 70% of the failures and find hourly failure rates during the day to be four times higher than during the night.

We are not aware of any studies that report failure rates over the entire lifetime of large systems. However, there exist commonly used models for individual software or hardware components. The failures over the lifecycle of hardware components are often assumed to follow a "bathtub curve" with high failure rates at the beginning (infant mortality) and the end (wear-out) of the lifecycle. The failure rate curve for software products is often assumed to drop over time (as more bugs are detected and removed), with the exception of some spikes caused by the release of new versions of the software [10, 9]. We find that the failure rate over the lifetime of large-scale HPC systems can differ significantly from the above two patterns (recall Figure 4).

8 Summary

Many researchers have pointed out the importance of analyzing failure data and the need for a public data repository of failure data [12]. In this paper we study a large set of failure data that was collected over the past decade at a high-performance computing site and will soon be made publicly available. We hope that this data might serve as a first step towards a public data repository and encourage efforts at other sites to collect and clear data for public release.

Below we summarize a few of the findings of our study.

- Mean failure rates vary widely across systems, ranging from 20 to more than 1000 failures per year. The failure rate depends mostly on the size of a system and less on the particular hardware.
- There's evidence of a correlation between the failure rate of a system and the type and intensity of the workload running on the system.

- The curve of the failure rate over the lifetime of an HPC system looks often very different from lifecycle curves reported in the literature for individual hardware or software components.
- Interarrival times of failures at individual nodes, as well as at an entire system are fit well by a gamma distribution or a Weibull distribution with decreasing hazard rate.
- Mean repair times vary widely across systems, ranging from 1 hour to more than a day. Repair times depend mostly on the type of the system, and are relatively insensitive to the size of a system.
- Repair times are extremely variable, even within one system, and hence poorly modeled by an exponential distribution. We find the best fit to be a lognormal distribution.

Our study is only a first step in analyzing the wealth of information provided by the data. There are many different ways in which the data could be used in future work.

An important question we haven't touched on is what the underlying correlation structures in the measured failure processes look like, including for example correlation between failures at the same node and correlations across nodes. An interesting study would be to use statistical methods from time-series analysis to identify the underlying correlation structures.

Another interesting question is how the data could be used to create realistic simulators or benchmarks. This involves the question of whether using simple distributions under the i.i.d. assumption is sufficient to achieve realistic results, or whether more complex models are necessary.

Finally, an interesting avenue for future work would be a detailed study of the relationship between the workload of a system and its failure rate. Our results indicate that the workload intensity as well as the type of workload affect failure rates. It would be interesting to systematically characterize this relationship and develop models that capture it.

9 Acknowledgments

We thank Gary Grider, Laura Davey, and the Computing, Communications, and Networking Division at LANL for their efforts in collecting the data and clearing it for public release. We also thank Roy Maxion, Priya Narasimhan, and the participants of the ISSRE'05 "Workshop on dependability benchmarking" for the many useful comments and questions.

References

[1] J. Gray. Why do computers stop and what can be done about it. In *Proc. of the 5th Symposium on Reliability in Distributed Software and Database Systems*, 1986.

[2] J. Gray. A census of tandem system availability between 1985 and 1990. *IEEE Transactions on Reliability*, 39(4), 1990.

[3] T. Heath, R. P. Martin, and T. D. Nguyen. Improving cluster availability using workstation validation. In *Proc. of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2002.

[4] R. K. Iyer, D. J. Rossetti, and M. C. Hsueh. Measurement and modeling of computer reliability as affected by system activity. *ACM Trans. Comput. Syst.*, 4(3), 1986.

[5] M. Kalyanakrishnam, Z. Kalbarczyk, and R. Iyer. Failure data analysis of a LAN of Windows NT based computers. In *Proc. of the 18th IEEE Symposium on Reliable Distributed Systems*, 1999.

[6] G. P. Kavanaugh and W. H. Sanders. Performance analysis of two time-based coordinated checkpointing protocols. In *Proc. Pacific Rim International Symposium on Fault-Tolerant Systems*, 1997.

[7] T.-T. Y. Lin and D. P. Siewiorek. Error log analysis: Statistical modeling and heuristic trend analysis. *IEEE Transactions on Reliability*, 39, 1990.

[8] J. Meyer and L. Wei. Analysis of workload influence on dependability. In *Proc. International Symposium on Fault-tolerant computing*, 1988.

[9] B. Mullen and Dave R. Lifecycle analysis using software defects per million (swdpm). In *Presentation at the 16th international symposium on software reliability (ISSRE'05)*, 2005.

[10] B. Murphy and T. Gent. Measuring system and software reliability using an automated data collection process. *Quality and Reliability Engineering International*, 11(5), 1995.

[11] Daniel N., John B., and R. Wolski. Modeling machine availability in enterprise and wide-area distributed computing environments. In *Euro-Par'05*, 2005.

[12] D. L. Oppenheimer, A. Ganapathi, and D. A. Patterson. Why do internet services fail, and what can be done about it? In *USENIX Symposium on Internet Technologies and Systems*, 2003.

[13] R. K. Sahoo, R. K., A. Sivasubramaniam, M. S. Squillante, and Y. Zhang. Failure data analysis of a large-scale heterogeneous server environment. In *Proc. of the 2004 international Conference on Dependable Systems and Networks (DSN'04)*, 2004.

[14] D. Tang, R. K. Iyer, and S. S. Subramani. Failure analysis and modelling of a VAX cluster system. In *Proc. International Symposium on Fault-tolerant computing*, 1990.

[15] N. H. Vaidya. A case for two-level distributed recovery schemes. In *Proc. of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, 1995.

[16] K. F. Wong and M. Franklin. Checkpointing in distributed computing systems. *J. Parallel Distrib. Comput.*, 35(1), 1996.

[17] J. Xu, Z. Kalbarczyk, and R. K. Iyer. Networked Windows NT system field failure data analysis. In *Proc. of the 1999 Pacific Rim International Symposium on Dependable Computing*, 1999.

[18] Y. Zhang, M. S. Squillante, A. Sivasubramaniam, and R. K. Sahoo. Performance implications of failures in large-scale cluster scheduling. In *Proc. 10th Workshop on Job Scheduling Strategies for Parallel Processing*, 2004.