Title

A study of the reliability of Internet sites

Authors

Long, DDE
Carroll, JL
Park, CJ

Publication Date

1991

DOI

10.1109/reldis.1991.145421

A Study of the Reliability of Internet Sites

D. D. E. Long*

Computer & Information Sciences
University of California
Santa Cruz, CA 95064

J. L. Carroll and C. J. Park

Mathematical Sciences
San Diego State University
San Diego, CA 92182

Abstract

Modeling the reliability of distributed systems requires a good understanding of the reliability of the components. Careful modeling allows highly fault-tolerant distributed data applications to be constructed at the least cost.

Failure and repair rates of components are often assumed to be exponentially distributed. This hypothesis is testable for failure rates, though the process of gathering and reducing the data to a usable form can be difficult. By applying an appropriate test statistic, some samples were found to have a realistic chance of being drawn from an exponential distribution, while others can be confidently classed as non-exponential.

Data were collected from a large number of hosts via the Internet. Almost all of the visible Internet (over 350,000 hosts) were considered, and more than 68,000 of these that were judged likely to respond were queried. These hosts were sampled several times to obtain up-times, and finally to determine average host availability. Estimates of availability, mean-time-to-failure (MTTF) and mean-time-to-repair (MTTR) were derived. The results reported here correspond with those commonly seen in practice.

1 Introduction

Many availability and reliability models assume that the failure and repair rates of components are exponentially distributed. This assumption is often made more for analytic simplicity than out of a conviction that it is the best model of reality. For example, recent studies of replicated data that employ Markov models [9, 6] depend on that assumption.

The exponential hypothesis is rigorously testable, although the process of gathering the data and the problem of interpreting it are non-trivial. Two of the important statistics that can be derived are *mean-time-to-failure* (MTTF) and *mean-time-to-repair* (MTTR). MTTF is not directly available from hosts, but it can be estimated using the length of time that hosts have been up, provided that the pattern of up-times is governed by a distribution that is approximately exponential. The problem of determining the length of time that a host has been down is an obvious example of indirect data acquisition, since a failed host is not in the position to immediately report its demise. An estimate of the MTTR can be derived by using the estimates of the MTTF and the average host availability.

While it might have been possible to install monitors at a large number of sites to collect these data, it was impractical to solicit the cooperation of the hundreds of system administrators[1] necessary to gather the desired data. Instead, the analysis was done using only data that could be obtained using the Internet[2] with no special privileges or added monitoring facilities. This was principally done by polling hosts using Sun RPC [13] to query rpc.statd to obtain up-times and to test availability, and by querying domain servers [7] to obtain host-specific information. A surprisingly rich collection of information can be gathered in this fashion, allowing several important parameters to be estimated.

Availability is difficult to estimate accurately using the Internet. This is due to the many possible reasons for a host not responding to a request, most of which are indistinguishable to the polling process. Among these are the host being down, the host not implementing the polling protocol, and both hard and soft network failures. To minimize these complications, this portion of the study was confined to polling machine types that have answered at least one of many queries over the past few months. This prevented the absence of Sun RPC capabilities being interpreted as a

*Supported in part by faculty research funds and a Regents Junior Faculty Fellowship from the University of California at Santa Cruz.

[1] A large number of system administrators answered an initial call for data. Unfortunately, the data they provided were too few and were often incomplete.

[2] It is interesting that these polling activities, encompassing more than 68,000 hosts repeated several times over a period of two months, elicited inquiries from only six system administrators.

failure, but at the expense of a small bias in favor of highly available hosts.

Analyses of MTTF and the causes of failure have usually been confined to specific systems. Recent studies include analyses of Tandem systems [4, 5] and the IBM/XA system [8]. Research covering heterogeneous systems is less common. Few such studies have appeared in the open literature, although it is certain that most companies perform reliability studies of their products internally. The difficulty in assembling sufficient data and applying the appropriate statistical tests has inhibited a thorough analysis of the shape of the failure distribution. In this study, the failure rate distributions of several common architectures are analyzed, and their MTTF are estimated.

Throughout this study, "failure" is defined in a distributed-environment sense; that is, as an inability to access a host. The term encompasses both hardware and software faults attributable to the host, and can include power failures and scheduled down-time. It can also be caused by off-site communications failures, ranging from the transitory absence of accurate routing information to problems with the physical communications links. No attempt has been made to characterize the causes of failure, though it seems that most failures are brief and are caused by software faults or voluntary reboots.

The method of data acquisition and the problems encountered in its reduction are described in §2. The host MTTF can be estimated from the *up-times* reported by each host by using the *length-biased sampling technique* described in §3. Success depends on the exponential nature of the data, a hypothesis which is examined in §4. The resulting estimates of the MTTF are discussed in §5, followed by the average host availability in §6. These results are used to derive estimates of the MTTR in §7. A summary of the results of this study follows in §8.

2 Data acquisition and reduction

To acquire information about the status of Internet hosts, top-level domain servers were queried for the names of the hosts at each site and for secondary domain servers, and this process was applied recursively to the entire Internet name-tree. This resulted in over 350,000 hosts, a substantial fraction of the total Internet.

There are two significant problems with this approach. Several installations choose to shield their internal network behind a gateway; information about those subnetworks cannot be obtained by this method. Some large installations such as sun.com contributed only one data point to this study, since only the gateway machine was accessible.

Once lists of hosts were obtained, duplicate names were consolidated and the domain servers were again queried to determine the type and operating system of each host. This information was used to analyze the system status information returned by each operational host, and was crucial to constructing a manageable and meaningful sample space: purging host types that were unlikely to answer[3] sped the polling process considerably.

This approach led to another pair of significant difficulties. First, there are some sites that do not provide any information about the model or manufacturer of individual hosts. For example, while the University of California at San Diego has 3,052 hosts and about one tenth of those answered queries for system status, the information provided by these hosts is of little value due to an unfortunate lack of host-specific information. The second, and possibly most challenging, problem stemmed from the many ways a system administrator may describe a host. This made it difficult to precisely identify the manufacturer and model of queried hosts. Often it was impossible to determine more than the manufacturer, or perhaps the processor family.

Once the host list was determined, data were gathered by polling each host using Sun RPC [13] to determine the system status, including the length of time it had been up. A time-out period of 15 seconds was initially chosen, since a typical round-trip time for an ICMP [11] echo request is less than one second, and Sun RPC uses UDP [10], which, like ICMP, is layered on top of IP [12]. The remaining 14 seconds was judged to be sufficient for the host to respond to the request. If the response was not received within that time window, then the host was deemed to be too heavily loaded to be considered available.

These assumptions, and the performance measures derived from them, are sensitive to the context. In the continental United States, a 15-second time-out period is quite reasonable. When the data gathering was expanded to a global scale, the availability for a typical host (Sun 4/60) appeared to drop below 86%. All hosts were polled again using a 60-second time-out period. This showed that world-wide availability was closer to 91%, and a 120-second time-out period gave similar results of 90.5%. Consequently, results for the 120-second time-out period were deemed the best indicator of availability of local hosts. For distributed applications requiring more acceptable response times, it might be more appropriate to base the design on 86% availability if the hosts are sufficiently distant.

A host that does not respond to the Sun RPC request may indeed be unavailable, but there are also other reasons for not receiving the desired response. The host may have failed, or may be unavailable due to a network failure. These two failure modes are often distinguishable: the IP protocol will typically notify the polling process of an un-

[3] An early version of this experiment that did not attempt to purge unlikely candidates took almost two weeks to complete a single poll.

reachable network, and even if it does not, the network failure would also be manifested as a cluster of unresponsive hosts. Surprisingly, only a few such network failures occurred during the polling periods.

A third possibility is that the host is reachable and available, but does not understand the Sun RPC protocol. This case is indistinguishable from a non-operational host, since UDP is a connectionless protocol and no response is returned if there is no server present. For hosts that were identified by their domain server as Suns, a lack of response was interpreted as a failed host. A fourth possibility is that a host with Sun RCP may have rpc.statd disabled. Such a host will decline to respond, but this is distinguishable from a failure and can be safely discounted.

Initially, responses from a random sampling of over 1,000 hosts were gathered, from which it was determined that the MTTF of a typical Internet host was on the order of 15 days. The plot of the sample cumulative distribution bore a striking resemblance to an exponential distribution. This hypothesis was tested in §4, and these tests [2] showed that while some data collected did indeed fit this pattern, other data did not.

In general, more than 75% of the hosts reported uptimes of less than 21 days. Consequently, a two-month period between the initial sampling and final sampling was deemed sufficient to allow for independence while minimizing the inaccuracies caused by changes to the namespace over time. Repeated polling could have been used to mitigate the bias in the availability data, but this was not attempted for this study due to time constraints.

An interesting complication arose in measuring availability since an unavailable host is not necessarily a host that has failed. A null response could instead be caused by a network failure. For many applications, current availability is indeed the proper measure, and the data supplied by Sun RPC is directly correlated to the expected availability of a typical host. The distinction is important when inferring absolute availability, but can be largely ignored when comparing the relative availability of different classes of hosts. The distinction is also irrelevant when local availability, rather than availability across the Internet, is analyzed.

To gauge the degree to which the communications medium affected the results, an experiment was conducted to obtain a measure of the reliability of the Internet. For this experiment, 24 geographically diverse Internet hosts were selected to be repeatedly polled using the ICMP echo protocol. Each poll consisted of a single datagram message sent from the host maple.ucsc.edu to the destination, and one reply datagram in response. One set of polls was collected for each host every 20 minutes over a 48-hour period. Each set of polls consisted of 50 ICMP echo requests issued at one-second intervals.

The data sets were examined to determine how often communication failures occurred. For hosts which remained available during the time of the experiment, most hosts responded to a poll more than 90% of the time. The one host which responded less often was continuously unavailable for 7 of the 48 hours of sampling. This would seem to indicate that message delivery, while not completely reliable, is highly likely.

The data sets were also examined to determine how long communication failures lasted. Failed polls were classified by the length of the run of failed messages of which they were a member. In more than half the cases where a message failed, the message was part of a run of only one or two failed messages. Discounting polls which failed due to host unavailability, more than 77% of the remaining failed messages were either single failures or part of a run of two failures. Consequently, a small number of retries of a poll seems sufficient to accurately determine whether a host is available.

3 Length-biased sampling

Randomly sampling the length of time since the last system initialization is distinct from sampling the length of time between initialization and failure. Sampling system up-time reports results in a skewed set of data, as hosts which have been up the longest are more likely to be polled. Analysis of the data must accommodate this effect. Let the length of the time interval from the reinitialization of a host until its next failure be denoted by the random variable X. This quantity is not directly observable, but must be inferred from data that is available. Let V represent the interval spanning the time between the last initialization and the time the sample was obtained. This value is observable, and can be obtained using Sun RPC. It is well-known [1, 3] that

$$E[V] = \frac{E[X^2]}{2E[X]}$$

and thus if X is assumed to have an exponential distribution with mean $\frac{1}{\lambda}$, then $E[X]$ matches the sample mean for V, that is, $E[V] = E[X]$. Intuitively, the degree to which X should exceed V is exactly counterbalanced by the length-biased sampling of V. Indeed, V will also be exponentially distributed with mean $\frac{1}{\lambda}$.

The contrapositive of this implication ensures that if V is not exponentially distributed, then neither is X. Since V is readily observable, a much richer sampling can be tested for exponentiality. If it is found that it is highly improbable that V is drawn from an exponential distribution, this

constitutes strong evidence that X is also not controlled by an exponential distribution.

In the case where X is exponentially distributed, the distribution of V matches X in both shape and mean. It is reasonable to assume that the shape of X approximates that of V when the distributions are approximately exponential. This implies that in typical cases, a sampling of up-time reports can be treated as a sampling of times-to-failure. This correspondence is the foundation of the estimation of the MTTF in §5.

4 Testing the exponential hypothesis

The cumulative distribution determined from an initial sampling of over 1,000 random hosts suggested that the underlying distribution is either exponential or a mixture of exponentials. As shown in figure 1, the graph of the logarithm of these values is remarkably straight. By designing an appropriate test statistic, the hypothesis that the sample values came from a single exponential distribution can be tested. A test statistic based on the parametric family of distributions with linear failure rate density has been shown to be applicable to a large class of nonparametric distributions as well, and has also been shown to be applicable to machine behavior [2]. This test does not depend on advance knowledge of the mean of the proposed governing distribution. For n samples t_1 through t_n with mean $\bar{t}$, the test statistic is given by

$$T = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \left[1 - \frac{1}{2}(t_i/\bar{t})^2 \right].$$

If the null hypothesis H_0 that the samples come from a single exponential distribution is true, the test statistic T has a standard normal distribution when the sample size n is large. Thus the null hypothesis H_0 can be rejected at a specified level of significance when the value of the equivalent formula

$$T = \frac{1}{2}\sqrt{n}\left[1 - \frac{\hat{\sigma}^2}{\bar{t}^2} \right]$$

is large, where $\hat{\sigma}^2$ is the sample variance.

The test statistic T can be also used in testing the null hypothesis that the samples come from a population with a linear failure-rate density as well as a population with a nondecreasing failure-rate average. For these cases, the null hypothesis can be rejected for large value of the test statistic T; the significance probability is calculated from the standard normal distribution.

No matter how large the sample size, no amount of testing can assure that a population distribution is exponential. By contrast, the test statistic T can quantify the prohibitively small probability that certain samples were derived from an exponential population distribution.

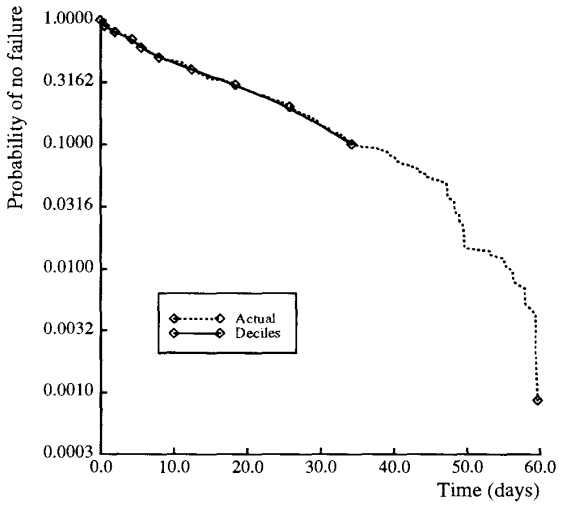

Figure 1: Semi-logarithmic Graph of Up-times for 1,154 Random Hosts.

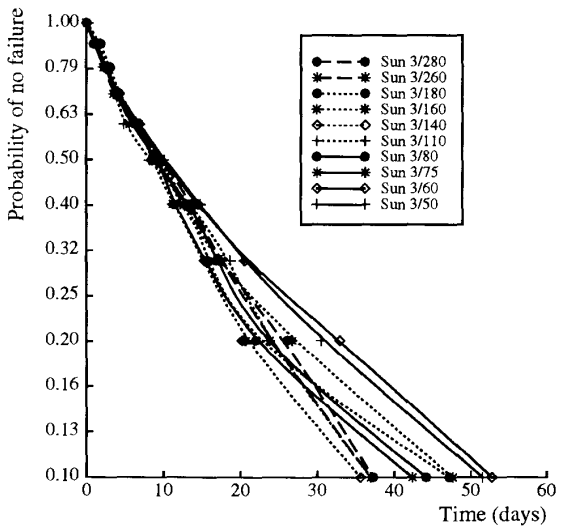

Figure 2: Semi-logarithmic Graph of Sun 3 Up-times.

The analysis of the initial random sampling of over 1,000 host responses was instructive. The sample mean was 15 days, and the median was 7.5 days. The raw data failed the exponentiality test rather spectacularly with a test statistic of 6.6. If the sample is drawn from an exponential distribution, the probability of observing a test statistic

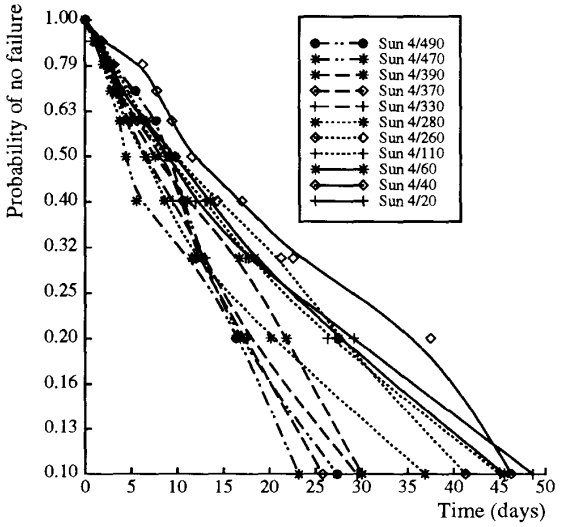

Figure 3: Semi-logarithmic Graph of Sun 4 Up-times.

value as large as 6.6 is on the order of one in ten billion: the probability of an exponentially-governed sample producing a test statistic of 6.6 or greater is equal to the probability that a sample point drawn from a normal distribution will be at least 6.6 standard deviations away from the mean.

Since several hosts were known to more than one domain server, approximately three percent of the sample points were repetitive; an additional two percent advertised improbably large up-times. When these repetitions were consolidated and the large numbers purged, the test statistic based on the modified sample shrank to 0.4. Under the assumption that the extremely large numbers are invalid data, the evidence against the exponential hypothesis is much weaker. If this modified sample is controlled by an exponential distribution, the probability of observing a test statistic value as large as 0.4 is just 2 in 3. This extremely small test statistic value provides strong evidence of the exponential nature of the sample.

Testing a collection of 27,022 raw data points showed that the sampled up-time V for this larger set of samples is definitely not exponentially distributed. This is certainly not surprising, for several reasons. The data includes some reports of hosts being up longer than their underlying hardware has been in existence; this seems unlikely.

A curiously large number of hosts reported up-times slightly over 20 years. These hosts were institutionally and geographically diverse, and hence the evidence points to an anomaly in the software that generates those numbers. As the dawn of time[4] was about 20 years ago, the most likely explanation is that these hosts maintain the correct current time but believe they were booted at time zero.

More importantly, the collection is not comprised of truly independent samples, as it is quite common to find entire sets of clients that are reinitialized within minutes of each other. This naturally occurs as a result of the failure of a common server, and may also be a consequence of common maintenance, back-up, or other administrative procedures. Classifying the time at which hosts reboot by the time of day and day of the week revealed striking differences in site procedures. Reboots at some organizations took place almost exclusively on Friday and Saturday, while other sites were limited to week-day work hours. It follows that reboots are not spontaneous events, but occur during periods of heavy use. This also supports the theory that most failures are software related.

In an attempt to ensure independent samples, one set of test data was built by taking at most one datum from each second-level domain. The resultant sample populations had test statistics that indicated a higher probability of exponential behavior, but most could still be distinguished from true exponential distributions.

As shown in §5, the MTTF of the different classes of hardware are distinct. This would suggest that the behavior of the hosts comprising the Internet might best be modeled as a sum of exponentials. Such a hyperexponential distribution would also exhibit straight-line behavior on a semi-logarithmic scale, similar to that in figure 1.

Figures 2 and 3 illustrate the distribution shapes of various Sun 3 and Sun 4 models, based on all data that could be associated with a specific model designation. The extremely sparse data in the last decile is not shown in the figure. On this semi-logarithmic scale, a perfect exponential curve would follow a straight line. Sample sizes for the Sun data on which these figures are based are shown in tables 1 and 2. Many of the sample populations are too small to accurately reflect the shape of the underlying distribution, but even the curves of the larger samples are not straight. The test statistic of each of these is sufficiently large to confidently reject the hypothesis that the samples are drawn from exponential distributions.

5 Estimating mean-time-to-failure

As discussed in §3, when the time to failure is exponentially distributed, its distribution agrees in both shape and mean with that of the up-time values reported by Sun RPC. When the failure distribution is approximately exponential, the mean of the values reported by Sun RPC provides an approximation of the MTTF. Although it is unlikely that the sample was drawn from an exponential distribution, the av-

[4]January 1, 1970 in UNIX reckoning.

erages obtained are a reasonable approximation of the actual MTTF and closely match those seen in practice[5] by system administrators. If the distributions of up-times is not exponential, but instead governed by a hyperexponential distribution, then the effect will be to over-estimate the MTTF.

A summary of the results are given in tables 1 through 3. The columns give the model, the size of the sample, the sample mean and a confidence interval for that sample MTTF. Also shown are the standard deviation, quartile and median values, and minimum and maximum observed samples. The poor confidence intervals for some of the models, notably the Sun 4/40 and Sun 4/470, reflects their small sample sizes. The median values shown in the column labelled 50% generally show a relationship to the mean that is compatible with that of exponential distributions.

Table 1 summarizes the results for hosts that could be identified as a specific model of Sun 4. The Sun 4/60 data points comprise the largest sample and should produce the most accurate estimate of MTTF. The preponderance of Sun 4/60 hosts predispose this subgroup to have the highest maximum, but the Sun 4 that had been up the longest was a 4/110. While it seems unlikely that any Sun 4/110 has been continuously operating for 287 days, the possibility could not be discounted. However, there were many cases where the value reported could be proven false. In particular, as mentioned in §4, there is a problem that causes rpc.statd to sometimes report an up-time on the order of 7,000 days, while Sun Microsystems has been in business less than half that length of time.

The small sample size does not seem to affect the MTTF to the same degree for the servers such as the Sun 4/280, 4/330 and 4/390. A plausible explanation for this is that most servers are independent. When a server is disabled, often the work stations that it serves are disabled as well. In contrast, disabling one server usually does not imply the disablement of other servers at that site.

The results for specific Sun 3 systems are reported in table 2. While there were 11,683 hosts that could be identified as Sun 3 systems, only 10,497 of these could be classified by specific model. The largest samples reported are for the Sun 3/50 and Sun 3/60. Again, a suspiciously large value of over 446 days is reported for a Sun 3/60. While this value could not be proven false, the frequency of such values was low enough to have little impact on the average, though this group did have the largest standard deviation.

It is important to keep the differences in usage patterns in mind when interpreting this data. In particular, an engineer developing a new file system may reboot his work station many times during the day. By contrast, a system dedicated to a single task, such as a file server or a domain server, may remain up for many months at a time. In some cases, users may turn off their work stations when they go home for the evening or over the weekend.

Table 3 represents other architectures that responded in significant numbers to the Sun RPC request, though there were many hosts that could not be precisely classified. In general, the MTTF of these classes of hosts closely matches the values reported by the Sun systems. The lowest statistically significant MTTF was recorded for the IBM PS/2 class at 12.5 days. This may be attributable to it being primarily a single-user machine and so be more likely to be turned off during the evening or over the weekend.

While there are a vast number of VAXen, only 365 of them responded to our queries and so the results are skewed towards those that support Sun RPC. Even so, the values reported agree closely with those reported for other systems.

6 Availability

The availability of a host is an important measure, indicating the probability that a host will be accessible. Some significant differences were noted for some classes of hosts.

The initial sequence of queries using Sun RPC was used to construct a list of known hosts. Approximately two months after the initial sampling, all of the responding hosts were again polled using Sun RPC. This two-phase method guarded against incorrectly attributing the absence of a response to a failure when the host might be permanently unreachable or even non-existent.

The two phases are important, but unfortunately this method is slightly biased against hosts with poor availabilities, as such hosts were more likely to be unnoticed during the initial samplings. This bias can be minimized by extensive polling at various intervals during the first phase, to ensure that most existing hosts are marked for participation in the second phase. In this study, the first phase was limited to two weeks due to time constraints and the considerable network traffic it generated. The two-month interval between the first and second phases was chosen to ensure that there is negligible correlation between a host being up in the first phase and being up in the second phase.

The servers, such as the Sun 4/280, 4/330, 4/390, 3/180 and 3/280 showed a uniformly higher availability than the work stations. This is to be expected since servers are more likely to be maintained by a staff person, and less likely to be shut down[6] when the user leaves in the evening.

[5] Several system administrators were contacted and shown the results. All agreed that the values were close to what they expected, although some thought that they were slightly too low, while others thought that they were slightly too high.

[6] It would be interesting to take geographical distribution into account since some countries, such as Austria, require unattended hosts to be

Table 1: Mean-Time-to-Failure for Specific Sun 4 Models.

Model	n	$\bar{x}$ (days)	95% Confidence		σ	min	25%	50%	75%	max
Sun 4/20	627	18.61	16.728	20.493	24.05	0.037	3.595	9.066	21.98	141.0
Sun 4/40	31	21.63	12.859	30.408	24.93	1.478	6.684	11.60	25.57	108.0
Sun 4/60	5598	17.96	17.348	18.571	23.35	0.007	3.155	9.759	22.38	238.6
Sun 4/110	489	18.18	15.891	20.462	25.78	0.004	3.028	9.028	23.52	287.0
Sun 4/260	250	16.76	14.353	19.165	19.41	0.036	3.692	9.779	22.52	140.8
Sun 4/280	347	13.90	11.932	15.858	18.66	0.016	2.356	6.712	16.79	131.0
Sun 4/330	176	13.92	11.153	16.686	18.73	0.084	2.781	6.629	16.38	108.7
Sun 4/370	99	17.36	12.700	22.010	23.63	0.011	3.088	9.152	19.27	100.7
Sun 4/390	111	12.97	10.305	15.627	14.30	0.030	3.274	7.591	20.96	98.98
Sun 4/470	38	16.59	10.299	22.887	19.80	0.113	2.778	6.364	27.38	77.03
Sun 4/490	94	15.07	10.578	19.552	22.20	0.075	3.925	9.499	16.21	170.6

Table 2: Mean-Time-to-Failure for Specific Sun 3 Models.

Model	n	$\bar{x}$ (days)	95% Confidence		σ	min	25%	50%	75%	max
Sun 3/50	4672	20.14	19.339	20.946	28.02	0.011	3.336	10.14	24.03	369.0
Sun 3/60	2679	20.08	19.036	21.127	27.61	0.003	3.106	9.822	23.67	446.4
Sun 3/75	150	16.15	12.597	19.702	22.20	0.067	2.792	9.442	17.25	119.8
Sun 3/80	804	17.10	15.449	18.757	23.93	0.036	3.344	8.697	19.93	225.7
Sun 3/110	287	17.15	14.605	19.693	21.99	0.022	3.502	8.863	22.50	127.3
Sun 3/140	108	17.30	13.238	21.369	21.56	0.065	2.903	9.637	20.49	111.8
Sun 3/160	680	17.73	15.897	19.567	24.41	0.014	3.450	9.270	20.83	240.3
Sun 3/180	237	17.84	14.851	20.836	23.50	0.103	3.578	9.663	17.78	136.4
Sun 3/260	355	16.25	14.170	18.336	20.02	0.011	2.662	9.076	21.95	129.0
Sun 3/280	453	15.85	14.196	17.499	17.94	0.010	3.315	9.597	23.09	117.0

The results for Sun 4 systems are summarized in table 4. The largest sample is for the Sun 4/60, which yields an availability of 90.6%. The inaccuracy of small samples is best illustrated by the Sun 4/470 servers. The confidence interval is so large that the estimate of the mean is effectively meaningless. When the available population is small, the only recourse is to repeatedly poll those hosts over an extended period of time until the desired degree of confidence can be obtained. Due to time constraints, this could not be undertaken in this study: to ensure independent samples, the time between polling attempts must be very large.

The results for Sun 3 systems are summarized in table 5. The largest sample is for the Sun 3/50, but the availability reported is unexpectedly low. This may be partially explained by the age of these systems, which are nearing the end of their useful life. In fact, Sun has several active programs to replace Sun 3s with Sun 4s. These older hosts may also be becoming more prone to failure, relegated to tasks that minimize the need to quickly restore them to service, or may even have been taken out of service during the months between the first and second polling phases. By examining the raw data, it was noted that several large clusters of these systems were down, as was their corresponding server.

Other systems were also considered and the results are summarized in table 6. Significantly lower availability figures are obtained when systems other than those that responded to the initial sampling are considered. In some cases, this is due to the small number of hosts in the sample. But the primary reason for this is many hosts simply do not support the Sun RPC protocol and so will not respond to the queries. The availability estimated for the classes of hosts in table 6 are slightly lower than those reported for some of the Sun systems, though most of the confidence intervals overlap.

The VAX systems provide an interesting example.

turned off.

Table 3: Mean-Time-to-Failure for Various Systems.

Model	n	$\bar{x}$	95% Confidence		σ	min	25%	50%	75%	max
HP 9000	1000	19.03	17.224	20.828	29.08	0.002	1.566	7.309	22.53	209.4
IBM PS/2	121	12.55	8.7274	16.366	21.43	0.043	1.226	4.016	11.36	126.5
NeXT	788	15.25	13.685	16.815	22.42	0.021	2.121	7.353	18.45	151.1
Sequent	75	18.71	11.738	25.677	30.79	0.070	2.299	9.396	17.36	164.8
SGI Iris	1866	17.71	16.747	18.668	21.17	0.037	3.574	10.57	22.22	136.1
VAX	365	18.42	15.589	21.255	27.62	0.001	2.893	9.870	20.16	212.3

Table 4: Availability for Specific Sun 4 models.

Model	n	$\bar{x}$	95% Confidence		σ
Sun 4/20	609	0.8851	0.8597	0.9104	0.3192
Sun 4/40	32	0.9688	0.9075	1.0000	0.1768
Sun 4/60	5646	0.9060	0.8983	0.9136	0.2919
Sun 4/110	522	0.8697	0.8408	0.8986	0.3369
Sun 4/260	255	0.9216	0.8885	0.9546	0.2694
Sun 4/280	344	0.9477	0.9241	0.9712	0.2230
Sun 4/330	181	0.8619	0.8115	0.9123	0.3460
Sun 4/370	108	0.8796	0.8180	0.9413	0.3269
Sun 4/390	112	0.9375	0.8925	0.9825	0.2431
Sun 4/470	39	0.9231	0.8384	1.0000	0.2700
Sun 4/490	97	0.9278	0.8761	0.9796	0.2601

Table 5: Availability for Specific Sun 3 Models.

Model	n	$\bar{x}$	95% Confidence		σ
Sun 3/50	4927	0.8717	0.8624	0.8811	0.3344
Sun 3/60	2790	0.8760	0.8638	0.8882	0.3297
Sun 3/75	148	0.8581	0.8017	0.9145	0.3501
Sun 3/80	794	0.8917	0.8701	0.9133	0.3110
Sun 3/110	299	0.8662	0.8276	0.9049	0.3410
Sun 3/140	120	0.7833	0.7093	0.8574	0.4137
Sun 3/160	735	0.8735	0.8494	0.8975	0.3327
Sun 3/180	233	0.9399	0.9093	0.9705	0.2382
Sun 3/260	374	0.8717	0.8377	0.9056	0.3349
Sun 3/280	472	0.9174	0.8925	0.9422	0.2756

While there are a vast number of VAXen, only 418 VAX systems responded to one of the several initial queries. The number is small since few VAXen support Sun RPC. Of these, 364 responded to the final Sun RPC request yielding an availability of 87.09%.

Table 6: Availability of Various Systems.

Model	n	$\bar{x}$	95% Confidence		σ
HP 9000	1053	0.8329	0.8103	0.8554	0.3733
IBM PS/2	120	0.8417	0.7761	0.9073	0.3666
NeXT	832	0.8389	0.8140	0.8639	0.3678
Sequent	85	0.8471	0.7701	0.9240	0.3621
SGI Iris	1989	0.8376	0.8214	0.8538	0.3689
VAX	364	0.8709	0.8364	0.9054	0.3358

7 Estimating mean-time-to-repair

The mean-time-to-repair[7] (MTTR) of a host can be estimated using information derived in previous sections. In particular, if the MTTF and the availability are known, the MTTR can be derived using the dependencies derived for a general renewal process [14]. This relationship does not depend on exponential failure and repair distributions. If A is the steady-state probability of the host being operational,

$$\text{MTTR} = \frac{\text{MTTF}(1-A)}{A}. \qquad (1)$$

The average host availability A was measured in §6. The results summarized in tables 7 through 9 are obtained by combining this information with the estimates of MTTF obtained in §5. The accuracy of the estimated MTTR depends on the accuracy of estimates of the availability and the MTTF. If the MTTF has been over-estimated, and the availability remains the same then the estimated MTTR will be longer than the actual MTTR. Also, if the confidence interval for the availability of the MTTF is large, then the estimated MTTR can deviate significantly from the true value.

The estimates of MTTR derived for Sun systems are are summarized in tables 7 through 8. The servers, such as the Sun 4/280, 4/390, 3/180 and 3/280 have a lower MTTR than the work stations. This is to be expected since servers

[7]For the purposes of this study, the mean-time-to-repair includes recoveries from any event that would cause the host to be rebooted.

are a critical resource and are usually maintained by support staff and are kept under a service contract. By contrast, work stations are less critical and may be serviced less frequently[8] than servers.

Table 7: Mean-Time-to-Repair for Specific Sun 4 Models.

Model	MTTF	Availability	MTTR
Sun 4/20	18.61	0.8851	2.4158
Sun 4/40	21.63	0.9688	0.6965
Sun 4/60	17.96	0.9060	1.8634
Sun 4/110	18.18	0.8697	2.7237
Sun 4/260	16.76	0.9216	1.4257
Sun 4/280	13.90	0.9477	0.7670
Sun 4/330	13.92	0.8619	2.2303
Sun 4/370	17.36	0.8796	2.3762
Sun 4/390	12.97	0.9375	0.8646
Sun 4/470	16.59	0.9231	1.3820
Sun 4/490	15.07	0.9278	1.1727

Table 8: Mean-Time-to-Repair for Specific Sun 3 Models.

Model	MTTF	Availability	MTTR
Sun 3/50	20.14	0.8717	2.9642
Sun 3/60	20.08	0.8760	2.8423
Sun 3/75	16.15	0.8581	2.6706
Sun 3/80	17.10	0.8917	2.0768
Sun 3/110	17.15	0.8662	2.6491
Sun 3/140	17.30	0.7833	4.7860
Sun 3/160	17.73	0.8735	2.5676
Sun 3/180	17.84	0.9399	1.1407
Sun 3/260	16.25	0.8717	2.3917
Sun 3/280	15.85	0.9174	1.4270

8 Summary

For this study, data were collected from as many hosts as was feasible using only data that could be obtained via the Internet with no special privileges or added monitoring facilities. The Internet was used to query domain servers to determine the name and type of over 350,000 hosts. This survey was then conducted using the more than 68,000 hosts that were judged to be types that were likely to respond to polling. These hosts were sampled several times over the course of two months to estimate the length of time that each had been up, and then a final poll was conducted determine average host availability. A surprisingly rich collection of information was gathered in this fashion, allowing estimates of availability, mean-time-to-failure (MTTF) and mean-time-to-repair (MTTR) to be derived. The measurements reported here correspond with common experience and certainly fall in the range of reasonable values.

[8]Two common approaches are weekly service by a technician and mail-in service where faulty parts are replaced through the mail.

Table 9: Mean-time-to-repair for Various Systems.

Model	MTTF	Availability	MTTR
HP 9000	19.03	0.8329	3.8179
IBM PS/2	12.55	0.8417	2.3603
NeXT	15.25	0.8389	2.9286
Sequent	18.71	0.8471	3.3771
SGI Iris	17.71	0.8376	3.4337
VAX	18.42	0.8709	2.7305

These data were gathered over a period of several months. The first phase, which lasted about seven days used Sun RPC to request statistics including the length of time the host had been up. This process was repeated approximately two weeks later. A final poll was taken several months later, and these data were used to determine availability and to augment the up-time data. This was done using time-out periods of 15, 60, and 120 seconds to determine the effect of communication delays on the results.

The domain servers were again queried to determine the type and operating system of each host. This information was useful in analyzing the system status information returned by each operational host. Some sites did not provide any host-specific information. Since such hosts cannot be classified, the information provided by these hosts was of little value. A more challenging problem stemmed from the many ways a system administrator may describe a host, making it difficult to precisely classify the host.

The effect of a bias towards hosts that have been available for longer periods must be accommodated. In the case where the total up-time X is exponentially distributed, the distribution of the sampled up-time V must match X in both shape and mean. It is reasonable to assume that the shape of X approximates that of V when the distributions are approximately exponential. This implies that in typical cases, a sampling of up-times can be treated as a sampling of times-to-failure.

The pattern of failures of many classes of hosts were clearly not exponential. This is not surprising when the data contains points that are obviously invalid, but many

classes still failed the test for exponentiality even when such noise is factored out. It is generally true that larger sample sizes provide more evidence against the exponential hypothesis. Massive amounts of data can more easily highlight minute deviations from exponential behavior. The sample comprised of all Internet responses conclusively failed the exponentiality test, pointedly emphasizing this tendency. For moderately-sized samples, it was often not possible to exhibit the deviation from exponentiality, lending credence to the common practice of assuming that MTTF is exponentially distributed.

Availability was difficult to estimate accurately using the Internet. This was due to the many possible reasons for a host not responding to a request, most of which are indistinguishable to the polling process. Among these are the host being down, the host not implementing the polling protocol, and both hard and soft network failures. This was addressed by using a two-phase approach that limited the domain to only those hosts that were known to be able to respond to Sun RPC requests. Unfortunately, this method is slightly biased against hosts with poor availabilities, as such hosts may go unnoticed during the initial samplings. This bias can be minimized by extensive polling at various intervals during the first phase, to ensure that most existing hosts are marked for participation in the second phase. The first phase was limited to two weeks due to time constraints and the considerable network traffic it generated. The two-month interval between the first and second phases was chosen to ensure that there is negligible correlation between a host being up in the first phase and being up in the second phase.

Although it is unlikely that the sample was drawn from an exponential distribution, the averages obtained are a reasonable approximation of the actual MTTF and closely match those seen in practice by system administrators. If the distributions of up-times is not exponential, but instead governed by a hyperexponential distribution, then the effect will be to over-estimate the MTTF.

The values reported for the MTTR are larger than one might expect. It is important to keep in mind that the MTTR is a derived quantity that depends heavily on the quality of estimates of both availability and the MTTF. The estimate of the MTTR will also be larger than the actual value if either the MTTF is overestimated or availability is underestimated.

Acknowledgments

The authors are especially grateful to Jim Gray and Robert Hagmann for their many excellent suggestions. Jehan-François Pâris, David Rosenthal, Richard Golding, Mary Long, and the referees contributed through their thoughtful comments. We are also grateful to Jeffrey Gould for his help in collecting the data.

References

[1] D. R. Cox and P. A. Lewis, *The Statistical Analysis of Series of Events*. London: Chapman and Hall, 1966.

[2] K. A. Doksum and B. S. Yandell, *Handbook of Statistics*, vol. 4. Elsevier, 1984.

[3] W. Feller, *Introduction to Probability Theory*, vol. II. New York: John Wiley and Sons, 1972.

[4] J. Gray, "Why do computers stop and what can be done about it?," Tech. Rep. 85.7, Tandem Computers, June 1985.

[5] J. Gray, "A census of Tandem system availability between 1985 and 1990," Tech. Rep. 90.1, Tandem Computers, Jan. 1990.

[6] D. D. E. Long, J. L. Carroll, and K. Stewart, "Estimating the reliability of regeneration-based replica control protocols," *IEEE Transactions on Computers*, vol. 38, pp. 1691–1702, Dec. 1989.

[7] P. V. Mockapetris, "Domain names — concepts and facilities," Tech. Rep. RFC-1034, USC Information Sciences Institute, Nov. 1987.

[8] S. Mourad and D. Andrews, "The reliability of the IBM/XA operating system," in *Proceedings 15th Annual International Symposium on Fault-tolerant Computing*, IEEE, June 1985.

[9] J.-F. Pâris, "Voting with witnesses: A consistency scheme for replicated files," in *Proceedings 6th International Conference on Distributed Computing Systems*, (Cambridge), pp. 606–612, IEEE, 1986.

[10] J. B. Postel, "User datagram protocol," Tech. Rep. RFC-768, USC Information Sciences Institute, Aug. 1980.

[11] J. B. Postel, "Internet control message protocol," Tech. Rep. RFC-792, USC Information Sciences Institute, Sept. 1981.

[12] J. B. Postel, "Internet protocol," Tech. Rep. RFC-791, USC Information Sciences Institute, Sept. 1981.

[13] Sun Microsystems, Incorporated, "RPC: Remote procedure call protocol specification version 2," Tech. Rep. RFC-1057, USC Information Sciences Institute, June 1988.

[14] K. S. Trivedi, *Probability & Statistics with Reliability, Queuing and Computer Science Applications*. Englewood Cliffs, New Jersey: Prentice-Hall, 1982.