Networked Windows NT System Field Failure Data Analysis

Jun Xu, Zbigniew Kalbarczyk, Ravishankar K. Iyer
Center for Reliable and High Performance Computing
University of Illinois at Urbana-Champaign
E-mail: {junxu, kalbar, iyer}@crhc.uiuc.edu

Abstract

This paper presents a measurement-based dependability study of a Networked Windows NT system based on field data collected from NT System Logs from 503 servers running in a production environment over a four-month period. The event logs at hand contains only system reboot information. We study individual server failures and domain behavior in order to characterize failure behavior and explore error propagation between servers. The key observations from this study are: (1) system software and hardware failures are the two major contributors to the total system downtime (22% and 10%), (2) recovery from application software failures are usually quick, (3) in many cases, more than one reboots are required to recover from a failure, (4) the average availability of an individual server is over 99%,(5) there is a strong indication of error dependency or error propagation across the network, (6) most (58%) reboots are unclassified indicating the need for better logging techniques, (7) maintenance and configuration contribute to 24% of system downtime.

1 Introduction

In a network of tens or even hundreds of PC servers, errors can propagate from one machine to another, and a single error may affect multiple computation nodes. While servers are usually loosely coupled, they still share system resources and cooperate in providing required services to the user. Because of the loose dependencies between servers, it is difficult and sometimes even impossible to identify error and failure dependencies in a networked system. As a consequence, although there is a lot of work (both in academia and in industry) on designing and building networked systems, characteristics of failures occurring in such systems are not well understood.

In this paper, we study a set of field failure data. Our goals are to understand the nature of failures in networked systems and to identify performance and dependability bottlenecks in such systems. The data are collected from a networked Windows NT system running in a production environment, providing services such as mail, file system services, and resource management. Data for the study are obtained from event logs collected over a four-month period, and the analysis is based solely on machine reboots (at this time this was the only event type collected). Despite the fact that the data includes only system reboots we are able to gain useful insights from them into the nature of observed failure behavior and typical problems in networked systems. Our key findings are summarized below:

- System software and hardware failures are the two major contributors to the total system downtime (22% and 10% of the total system downtime, respectively). However, these two failure categories are responsible for only 3% (system software) and 1% (hardware) of all observed reboots.
- Planned maintenance and changes in software and hardware configuration are responsible for 31% of all reboots. Although repair times are shorter, they still contribute 24% of the total downtime.
- Application software failures are responsible for 3% of all reboot, however, they are usually repaired quickly and contribute only 1% of the total downtime.
- Reboots on single or multiple machines within a domain tend to occur in bursts. This indicates that the problems are not removed by a single reboot, and in many cases, the rebooted machine encounters problems within less than an hour.
- Although the average availability of individual servers is over 99%, there is a strong indication of error propagation across the network. There is a high probability of multiple servers failing within a short interval (e.g., five machines in one domain fail within an hour).

This paper is organized as follows. Section 2 discusses some related work. Section 3 presents the data collection method and data format. Section 4 outlines failure classification and categorization and analyzes dependability parameters for different failure categories.

Section 5 talks about failure behavior of individual servers. Section 6 presents failure behavior of a domain and explores error propagation. We conclude with Section 7.

2 Related Work

Analysis of failures in computer systems has been the focus of active research for quite some time. Lee [10] analyzed failures in Tandem's GUARDIAN operating system. In the study, processor halts were examined, and it was found that memory management software and interrupt handlers were the major causes for processor halts. Lee [11] examined software dependability of the GUARDIAN operating system. The study categorized the failures based on the underlying software problem (e.g., uninitialized pointers or incorrect update of data structures). The evaluation revealed that the systems were able to tolerate about 75% of the processor failures due to their fault tolerance techniques.

Tang [20] analyzed error logs pertaining to a multicomputer environment based on a VAX/VMS cluster. Thakur [21] presented an analysis of failures in the Tandem Nonstop-UX operating system. The study analyzed problems that resulted in panics and system crashes. Thakur [22] described a simple yet effective methodology for collecting and analyzing failures in a network of UNIX-based workstations. The majority of observed failures (68%) encountered were network related. Kalyanakrishnam [9] analyzed machine reboots collected in a network of Windows NT-based mail servers. It was observed that though the average availability of a machine is high (over 99%), a typical machine in the domain provides acceptable service only about 92% of the time, on average.

Hsueh [5] explored errors and recovery in IBM's MVS operating system. Based on the error logs collected from MVS systems, a semi-Markov model of multiple errors (i.e. errors that manifest themselves in multiple ways) was constructed to describe the system failure behavior. Measurement-based software reliability models were also presented in [10] and [11] for the GUARDIAN system and [20] for the VAX cluster.

The impact of workload on system failures has also been extensively studied. Castillo [1] developed a software reliability prediction model that took into account the workload imposed on the software. The study examined hardware transient faults that occur in time-sharing systems and proposed a model for software reliability. Iyer [7] examined the effect of workload on the reliability of the IBM 3081 operating system. The study proposed and validated a load-hazard model in order to measure the risk of system failure with increasing workload. Mourad [16] performed a reliability study on the IBM MVS/XA operating system and found that the error distribution is heavily dependent on the type of system utilization. Meyer [13] presented an analysis of the influence of workload on the dependability of computer systems.

Maxion [14] [15] presented useful and interesting results on anomalies and their detection. Lin [12] and Tsao [23] focused on trend analysis in error logs. McConnel [17] presented results of an analysis of transient errors in computer systems. This study showed that transients follow a Weibull distribution rather than occur at a constant rate as frequently assumed. Gray [3] presented results from a census of Tandem systems. Chillarege [2] presented a study of the impact of failures on customers and the fault lifetimes. Sullivan [18], [19] examined software defects occurring in operating systems and databases (based on field data). An in-depth overview of experimental and analytical techniques for analysis of computer systems dependability can be found in [8].

3 Field Data and Analysis Parameters

This section presents data collection method and data format, and defines dependability parameters used in failure data analysis.

3.1 Data Collection Method and Format

Field	Field Description
Server	Name of the server
Outage Start	Date and time an outage started.
Boot time	Date and time the server rebooted. Reboot ends an outage, i.e., it is the Outage End time.
Shutdown Type	Clean or dirty
Outage Reason	Cause of the outage/reboot
OpAssist Time	Time an operator made an annotation
OpAssist Tier1	Operator's primary reason for the reboot
OpAssist Tier2	Operator's secondary reason for the reboot.
Operator Comm	Operator's freeform notes about the reboot.

Table 1 Failure Data Definitions

The field failure data are collected from 503 PC servers running Microsoft Windows NT 4.0, Service Pack 4 during a four-month period in a corporate production environment. The servers operate continuously (24x7) and implement a company's major internal services and applications such as file and print services, mail service, enterprise resource planning, product support service, sales automation and corporate marketing. The data in this paper are collected using the event log analyst (ELA) tool. ELA collects reliability information from NT event logs. ELA runs on a single server, the collection server, and sequentially retrieves event log information from other NT servers in the system. Although ELA

collects all events in the system, we only have access to data on system reboots. Table 1 shows the data format.

3.2 Parameter Definitions

Several dependability parameters are calculated from the field failure data.

Mean Time Between Failure (MTBF): The average time interval, in hours, between the start time (Outage Start) of two consecutive outages. The time difference between MTBF and MTTR (see below) is a mean operational time of a server.

Mean Time To Repair (MTTR): The mean time, in hours, between the Outage Start and Boot Time (Outage End). MTTR is a measure of system downtime.

Availability: Availability is defined as (MTBF – MTTR)/MTBF.

All three parameters are used to evaluate a single server as well as an entire domain in which all machines (e.g., mail server) perform the same services and are geographically closely distributed. Outage and failure are used interchangeably in this paper.

4 Failure Classification and Analysis

In this section, we discuss failure classification and distributions of time between failures.

4.1 Classifications and Observations

The field data provide information on errors and failures in the system. To assess the impact of different failures on system dependability, the recorded failures are categorized. Then dependability parameters, including MTBF and MTTR, are evaluated with respect to individual failure categories.

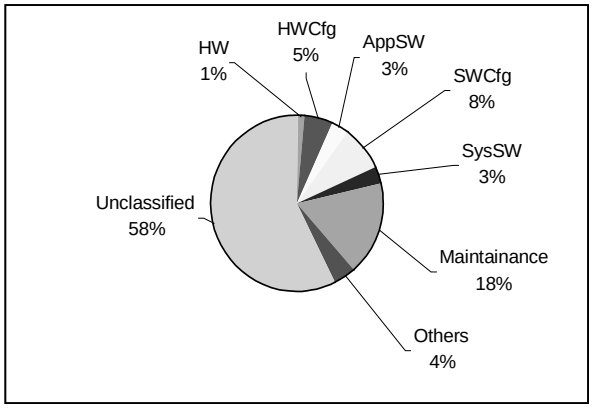

Figure 1 Distribution of Number of Outages

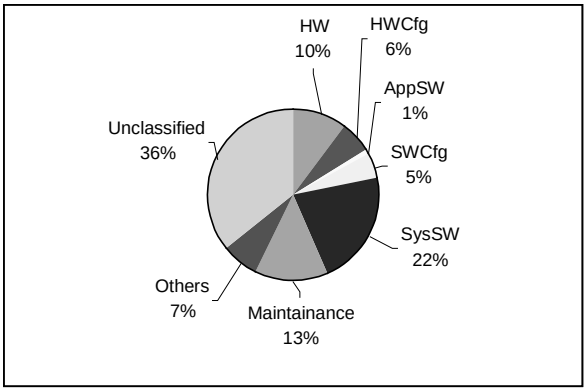

Figure 2 Downtime Distribution by Categories

There are 2,127 reboots in all, which we sorted according to Outage Reason field. This field allows us to distinguish 63 unique outage reasons. We classified these reasons into the categories as shown in Table 2. Using the above classification, MTBF and MTTR for individual failure categories are calculated and given in Table 3. In Figure 1 and Figure 2 we show the distributions (by failure categories) of outage numbers and downtime, respectively. Based on the obtained results, we came to the following observations and conclusions:

- Downtimes due to hardware and system software failures are much larger other types. Table 3 shows that MTTRs for hardware and system software failures are 10 and 9 times larger than MTTR for other failures respectively. These two categories contribute 32% of the total downtime (Figure 2), however they constitute only 4% of all failures (Figure 1).
- Planned maintenance, software and hardware configuration and install add up to 31% of all reboots. Although their MTTRs are shorter, they still contribute to 24% of total downtime. Consequently, better maintainability and configurability are essential for a networked system to achieve higher availability. Shutdown of one machine (hence the services it provides) will potentially affect many other servers in the system and cause a chained shutdown. It is highly desirable for a system to support non-shutdown maintenance, such as hot hardware and software upgrades.
- For 58% of the failures, the system log does not show any specific reasons. Better logging techniques are required to provide more accurate information. This will enable more accurate analysis of system availability and more precise identification of system vulnerabilities;
- There are 3% application software failures, but they usually recover quickly, contributing to only 1% of total downtime. MTTR is 28 and 24 times smaller than those for hardware and system software failures (see Table 3).

Catogory	Outage Reasons	Description
HW	RAM Fail, Power Supply, Disk Fail, CPU Fail, ISDN HW, Modem Fail, NIC Fail, Tape Issue	Hardware failures and problems;
HWCfg	Net HW Config, Firmware Upg, HW Config, Move Server, HW Install	Hardware Config/Install
AppSW	Invoice-SAP, Exch Err, Exch, Web Svc Fail, SQL Fail, Compass, DBCC.exe, Exch Fail, Exch Restore, Exch Store, SQL Svc Fail, SQL tools	Application Software Problems
SWCfg	Exch 5.5 SP2, SQL7 Inst, Net SW Config, SW Install, IE4 SP1 Inst, Outlook Inst, SW Config, Rename Server, VM Config, IIS Install, MDAC Inst, MDAC2.1 Inst, NT4 SP4 Inst, SQL6.5 Inst	Software Confg/Install
SysSW	NT Bugcheck, Hang, GAMMA.SYS, Out of VM, RPC, DNS Fail, WINS, RAS Fail	System Software Failure
Maintenance	SAP Weekly, Chkdsk, SMW, NT Hotfix	Planned Maintenance
Others	Customer Req, Troubleshoot, CDONTS, PDC Trust, SFM, Video, Op Error, Outage Reason, PCD Trust, Print Fail, Troubleshooting	Miscellaneous
Unclassified	Unclassified	Don't the reason

Table 2 Classification of Outage Reasons

Catogory	#Outages	MTBF	MTTR
HW	27	92.742	2.04
HWCfg	109	17.978	0.287
AppSW	69	31.524	0.072
SWCfg	178	13.61	0.151
SysSW	67	38.537	1.751
Maintenance	370	5.92	0.197
Others	88	27.854	0.456
Unclassified	1219	2.397	0.157
All	2127	1.37	0.25

Table 3 Stats for Different Failures (unit = hour)

4.2 Distribution of Time Between Failures

We also compute TBF distributions for all recorded reboots and for two selected failure categories (system software failures and hardware failures which contribute to 22% and 10% of total system downtime).

Figure 3, Figure 4, and Figure 5 show distribution of time between failures for the entire system (including Unclassified failures), for the entire system (excluding Unclassified failures), and for system software failures. The data labels show the numbers in each bin of a histogram.

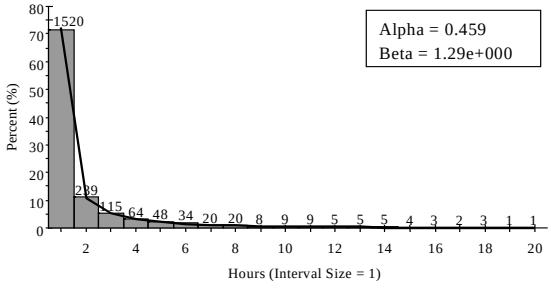

Figure 3 TBF Distribution for All Outages

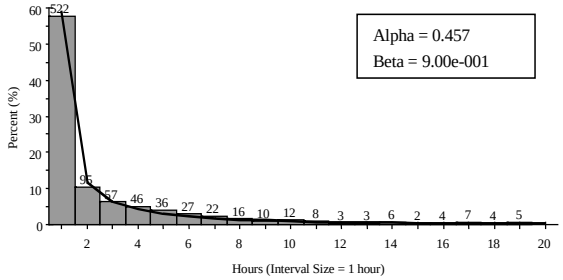

Figure 4 TBF Dist. for Classified Outages

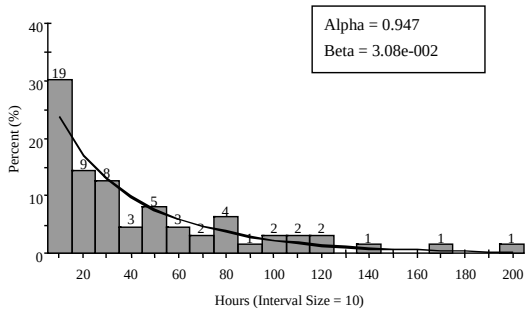

Figure 5 TBF Dist. for System Software Failures

Distributions obtained based on all outages in the system (with or without Unclassified failures) and based on the outages due to system software failures follow Weibull distribution with parameters given in the figures. We could not, however, find a good distribution fit for outages due to hardware failures. Figure 3 and Figure 4 indicate that excluding unclassified failures does not change the distribution of time between failures. From Figure 5, we observe that for more than 70% of system software failures (and for 45+% of hardware failures), the time between failures is less than two days.

5 Failure Behavior of Individual Servers

Figure 6 shows empirically obtained distribution of availability per server (only 399 servers with at least 30 days of Service Pack 4 runtime are considered). Individual servers demonstrate high availability — almost 99% of servers have availability higher than 99%. However, many servers experienced a significant number of outages during the four-month period. We picked out five servers (with the largest numbers of outages) and conducted per-server-based analysis. Table 4 shows results for these servers. An interesting observation is that the standard deviations of the TBF of selected servers are very large. This implies that the outages for a specific server are clustered closely in time.

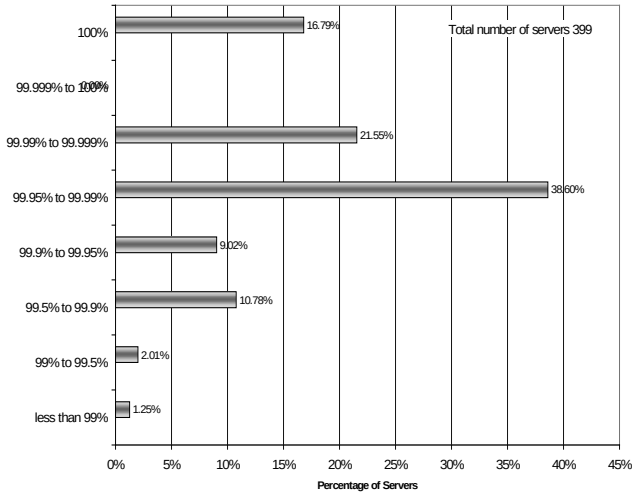

Figure 6 Server Availability Distribution

Server	CPGA	NETC	ERM11	SAPC	EASM
# of Rebs	84	51	37	28	28
Availability	0.9935	0.9986	0.9985	0.9993	0.9993
MTBF (hr)	17.15	46.98	65.31	81.94	96.74
TBF StdDev	32.13	99.64	89.35	137.04	125.28
MTTR (hr)	0.11	0.06	0.09	0.09	0.06

Table 4 Individual Servers' Failure Statistics

Figure 7 and Figure 8 show how failures of NETC and CPGA are distributed in time. NETC and CPGA experienced 51 and 84 reboots during the four-month period, respectively. We can see that most of the time (Y axis), the servers run stable. Once the machine encounters an error and fails, it usually experiences several failures and reboots before it returns to a stable state. We coalesced the reboots for the machine using different coalescing interval and found that periodic behavior repeats around once in 10 days (NETC) and once in 5 days (CPGA) on the average. We tried to determine the cause of such behavior; unfortunately, most of the reboot reasons are Unclassified. More detailed data are needed to conduct thorough analysis to make a conclusive statement. The similar periodic behavior is also observed in other servers.

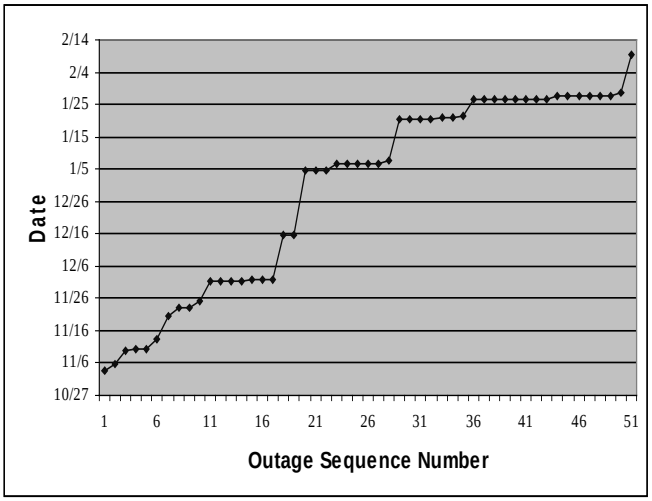

Figure 7 Server NETC's Failure Behavior

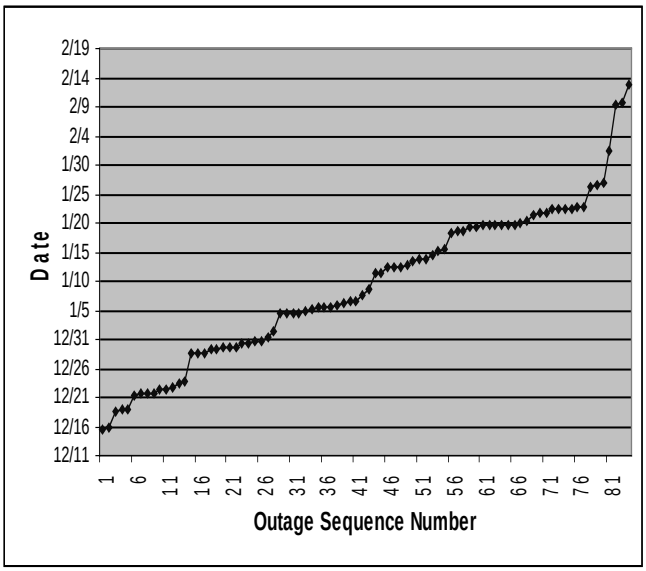

Figure 8 Server CPGA's Failure Behavior

Based on the periodic failure behavior and series of reboots to recover from a failure, two possible reasons can be identified:

- A server runs stable for certain amount of time and crashes, possibly due to cumulative problems like

memory leaks or file system errors. Such crash recurs;

- System shutdown and incomplete system cleanup because of a failure can leave the system in inconsistent state. The operating system is not able to recover (possibly repair file system inconsistencies or system configuration file damages) from such problems quickly, and it usually takes several reboots to fully recover.

6 Domain Behavior and Error Propagation Analysis

In this section, we describe results from analysis of domain failure behaviors.

6.1 Analysis Techniques

The goal of the domain behavior analysis is to explore failure patterns and to determine their indication. We define *domain* as a set of closely connected servers providing similar services (e.g., mail). Servers in a domain usually depend on one another, for instance, in electronic mail domain, one server might perform the forwarding or directory service to other servers in the domain. Another example is enterprise resource management domain, in which servers cooperate with each other to perform resource allocation. While it is difficult to identify error dependencies in a system with more than five hundred servers, we choose the domain-based analysis approach to explore error propagation in a networked environment.

We processed the logging data using data coalescing techniques to identify error dependency. While most coalescing algorithms merge *errors* of the same type that occur within a certain interval ΔT into a tuple, our approach is slightly different. As our objective is to study error propagation and dependency, we coalesce *failure* entries within ΔT into a cluster, regardless of the failure type. In particular, our coalescing algorithm works as follows:

```
IF <time from previous failure> ≤ ΔT
THEN <put it into the current cluster>
ELSE <create a new cluster>
```

Note that the interval (ΔT) in the coalescing algorithm acts as a "sliding window" in a process of clustering failure events. Depending on the failure frequency, the size of a cluster (in terms of time) can be larger than ΔT as we will see in the next section. The choice of ΔT is also important. Too long or too short will cause collision or truncation [4][8]. Most published results use an interval of 5 minutes for single processor machines or tightly coupled systems. In our case, the networked system is loosely coupled and we only consider reboot data, thus we choose a value that is much larger than the usual 5 minute. The choice is difficult but after several tries, we chose a ΔT of 1 hour.

6.2 Error Dependency and Propagation

We conducted our analysis for two domains: one performing enterprise resource management (ERM Domain) and another providing mail service (MAIL Domain). Due to limited space, we only present the detailed results for ERM Domain. The important statistics of MAIL Domain are shown in Table 6.

A distinguished characteristic of the ERM Domain is that all the servers have planned weekly reboots at a weekend midnight (they are rebooted at almost the same time). We remove those weekly reboots from the log data so they do not distort our analysis. Table 5 shows the statistics we obtained for each server in the domain after the weekly reboots are removed. MTBF and MTTR columns are computed based on the data before coalescing; MTBF' is the time between beginning of two consecutive clusters, MTTR' is the duration of a cluster, MTTR'' is same to MTTR' except that it is for clusters with 2 or more failures. Table 5 also gives number of clusters (*Cluster#*), total number of reboots (*Reb#*)and reboots per clusters (*Reb/Cluster*) for each machine. Observe that although the ΔT in the coalescing algorithm is set to be 1 hour, MTTR'' can be larger than ΔT (ERM08, ERM11, ERM14). This is because ΔT is employed as a sliding window in coalescing reboots.

To get insight into error propagation we compute number of clusters, total number of reboots and number of reboots per clusters using coalesced data for entire domains. Table 6 shows results for ERM Domain, MAIL Domain, and the entire system. While the number of clusters computed for each server within ERM Domain adds up to 85 (the sum of *Cluster#* column in Table 5 or the *Clust SUM* column in Table 6) there are only 32 clusters (*Clust#* column) for the entire ERM domain as shown in Table 6. The similar situation is also true for the MAIL Domain, where the number of clusters computed for each server within the domain adds up to 100 and there are only 44 clusters for the entire domain. This is a clear indication of error propagation within a domain. If errors do not propagate, the cumulative number of clusters obtained by coalescing failure events for individual servers within the domain and the number of clusters for an entire domain (without distinguishing individual servers) should be very close (if not the same).

Server	MTBF	MTBF'	MTTR	MTTR'	MTTR"	Cluster#	Reb#	Reb/Cluster
ERM00	98.946	240.299	0.121	0.464	0.833	8	18	2.25
ERM01	336.165	796.743	0.101	0.424	0.695	4	6	1.50
ERM02	335.891	559.721	0.130	0.552	0.628	4	6	1.50
ERM03	552.261	828.392	0.050	0.460	0.664	3	4	1.33
ERM04	336.435	1007.394	0.092	0.722	0.814	3	6	2.00
ERM05	129.197	239.877	0.079	0.399	0.750	8	14	1.75
ERM06	184.891	267.645	0.167	0.458	0.822	9	11	1.22
ERM07	280.192	525.745	0.102	0.470	0.922	5	7	1.40
ERM08	156.756	587.835	0.080	0.414	1.204	6	16	2.67
ERM09	335.852	839.311	0.092	0.665	0.977	3	6	2.00
ERM10	420.083	840.165	0.080	0.366	0.828	4	5	1.25
ERM11	83.965	503.791	0.105	0.697	2.184	7	25	3.57
ERM12	335.818	559.557	0.427	0.867	1.139	4	6	1.50
ERM13	124.453	217.793	0.177	0.446	0.650	9	15	1.67
ERM14	80.013	239.782	0.108	0.824	1.288	8	22	2.75

Table 5: ERM Domain Statistics

Domain	Clust SUM	Clust#	Reb#	Reb/Clust	Mach#/Clust
ENTIRE	N/A	422	2126	5.04	3.60
ERM	85	32	167	5.22	2.56
MAIL	100	44	115	2.61	2.27

Table 6: Domain Statistics

Another indication of error propagation is the fact that the number of different servers involved in each cluster, 2.56 for ERM Domain, 2.27 for MAIL Domain, shown in column *Mach#/Clust* of Table 6. If errors are not dependent or do not propagate, the probability of two different machines fail within an hour window is very small and thus the expected number of machines failed within a single cluster shall be very close to 1, given the high availability of each individual server as shown in Figure 6.

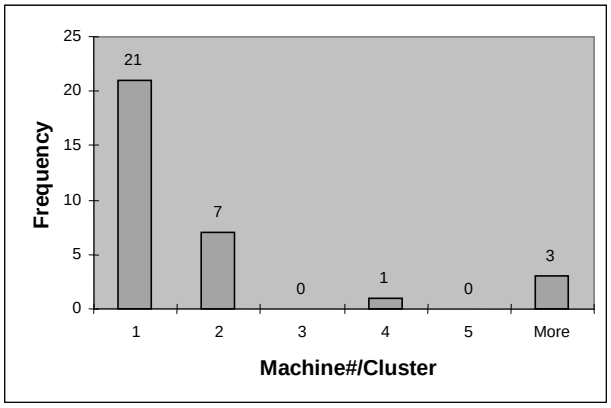

Figure 9 Number of Machines Per Cluster Histogram (ERM Domain)

From\To	0	1	2	3+
0	0.989	0.008	0.002	0.001
1	0.553	0.395	0.053	0.000
2	0.222	0.556	0.222	0.000
3+	0.250	0.000	0.000	0.750

Table 7 Transition Probability Matrix for ERM Domain

Figure 9 gives the number of machines per cluster histogram. We observed that one third of all the clusters involve more than one machine and in 3 clusters, there are more than 5 machines involved. By studying the machines involved in a cluster and the reasons for their outages, we can classify error dependency and propagation.

To obtain more accurate understanding of domain failure behavior, we modeled the domain in terms of a state transition matrix. The model was based on the reboots in the domain. The domain is either in State 0 (a fully functional state in which no servers are in outage) or in State 1 to State k (partially functional states in which 1 to k servers are in outage). Selection and assignment of states was performed as follows. The outages were split into time windows of one hour each. For each such window, the domain was assigned a state. The assignment of states is based on number of servers in outage. Table 7 shows the transition probability matrix between the different states in the ERM Domain. The source states are specified by the first column and target states by the column headings. The numbers show the transition probability from source state to target state. As the transition probability matrix shows, most of the time, the domain is in a fully functional state. However, once the domain enters a state with a server down, there is a non-negligible probability that the system will stay in this state or more servers will fail. For example, Table 7

shows that in State 1 there is 39.5% chance that the domain will stay in this state and 5.3% chance that another server will fail. This is yet another indication of possible error propagation across the network.

7 Conclusion

This paper presents a case study and preliminary failure characterization of Windows NT-based networked systems. The analysis is based on real world failure data collected by the error logging mechanism in the underlying operating system. The results show that while most servers provide high availability (> 99%), there is clear indication of error propagation across the network. Issues such as effective system maintainability, fast recovery are key in achieving higher system availability. This study also convinces us that available logs provide useful data on failure behaviors of NT-based systems. On the other hand, we have also learned that in many cases, the information contained in logs is not sufficient to make a definite interpretation. We believe, however, that errors in interpreting the log data towards the conservative side, i.e., we do not overestimate the system. This study is a starting point in more complete analysis of Windows NT based systems. We plan to continue our work based on more detailed system logs, which should allow us to address issues such as individual server's long recovery from system software and hardware failures.

References

[1] X. Castillo and D.P. Siewiorek, "A Workload Dependent Software Reliability Prediction Model," *Proc. 12th Int. Symp. Fault-Tolerant Computing*, pp.279-286, June 1982.

[2] R. Chillarege,S. Biyani, and J. Rosenthal, "Measurement Of Failure Rate in Widely Distributed Software," *Proc. 25th Int. Symp. Fault-Tolerant Computing*, July 1995.

[3] J. Gray, "A Census of Tandem System Availability between 1985 and 1990," *IEEE Trans. Reliability*, Vol. 39, No. 4, pp. 409-418, October 1990.

[4] J.P. Hansen and D.P. Siwiorek, "Models for Time Coalescence in Event Logs," *Proc. 22nd Int. Symposium on Fault-Tolerant Computing*, pp.221-227, July 1992.

[5] M.C. Hsueh, and R.K. Iyer, "A Measurement-based Model of Software Reliability in a Production Environment," *Proc. 11th Annual Int'l Computer Software & Applications Conference*, pp. 354-360, October 1987.

[6] R.K. Iyer, D.J. Rossetti and M.C. Hsueh, "Measurement and Modeling of Computer Reliability as Affected by System Activity," *ACM Trans. Computer Systems*, Vol. 4, No. 3, pp. 214-237, August 1986.

[7] R.K. Iyer and D.J. Rossetti, "Effect of System Workload on Operating System Reliability: A Study on IBM 3081," *IEEE Trans. Software Engineering*, Vol. SE-11, No. 12, pp. 1438-1448, December 1985.

[8] R. Iyer, D. Tang, "Experimental Analysis of Computer System Dependability," Chapter 5 in *Fault Tolerant Computer Design*, D.K. Pradhan, Prentice Hall, pp.282-392, 1996.

[9] M. Kalyanakrishnam, "Analysis of Failures in Windows NT Systems," *Master Thesis, Technical report CRHC 98-08*, University of Illinois at Urbana-Champaign, 1998.

[10] I. Lee and R.K. Iyer, "Analysis of Software Halts in Tandem System," *Proc. 3rd Int. Symp. Software Reliability Engineering*, pp. 227-236, October 1992.

[11] I. Lee and R.K. Iyer, "Software Dependability in the Tandem GUARDIAN Operating System," *IEEE Trans. on Software Engineering*, Vol. 21, No. 5, pp. 455-467, May 1995.

[12] T.T. Lin and D.P. Siewiorek, "Error Log Analysis: Statistical Modeling and Heuristic Trend Analysis," *IEEE Trans. Reliability*, Vol. 39, No. 4, pp. 419-432, October 1990.

[13] J.F. Meyer and L. Wei, "Analysis of Workload Influence on Dependability," *Proc. 18th Int. Symp. Fault-Tolerant Computing*, June 1988.

[14] R.A. Maxion, "Anomaly Detection for Diagnosis," *Proc. 20th Int. Symp. Fault-Tolerant Computing*, pp. 20--27, June 1990.

[15] R.A. Maxion and F.E. Feather, "A Case Study of Ethernet Anomalies in a Distributed Computing Environment," *IEEE Trans. Reliability*, Vol. 39, No. 4, pp. 433-443, October 1990.

[16] S. Mourad and D. Andrews, "On the Reliability of the IBM MVS/XA Operating System," *IEEE Trans. on Software Engineering*, October 1987.

[17] S.R. McConnel, D.P. Siewiorek, and M.M. Tsao, "The Measurement and Analysis of Transient Errors in Digital Compute Systems," *Proc. 9th Int. Symp. Fault-Tolerant Computing*, pp. 67-70, 1979.

[18] M.S. Sullivan and R. Chillarege, "Software Defects and Their Impact on System Availability — A Study of Field Failures in Operating Systems," *Proc. 21st Int. Symp. Fault-Tolerant Computing*, pp. 2-9, June 1991.

[19] M.S. Sullivan and R. Chillarege, "A Comparison of Software Defects in Database Management Systems and Operating Systems," *Proc. 22nd Int. Symp. Fault-Tolerant Computing*, pp. 475-484, July 1992.

[20] D. Tang and R.K. Iyer, "Analysis of the VAX/VMS Error Logs in Multicomputer Environments — A Case Study of Software Dependability," *Proc. Third Int. Symp. Software Reliability Engineering*, Research Triangle Park, North Carolina, pp. 216-226, October 1992.

[21] A.Thakur, R.K.Iyer, L. Young, I. Lee, "Analysis of Failures in the Tandem NonStop-UX Operating System," *Proc. Int'l Symp. Software Reliability Engineering*, pp. 40-49, 1995.

[22] A.Thakur and R.K.Iyer, "Analyze-NOW — An environment for Collection and Analysis of Failures in a Network of Workstations," *IEEE Trans. Reliability*, Vol. R-46, No. 4, pp. 561-570, 1996.

[23] M.M. Tsao and D.P. Siewiorek, "Trend Analysis on System Error files," *Proc. 13th Int. Symp. Fault-Tolerant Computing*, pp. 116-119, June 1983.