Performance and Reliability of Computer System

D.S Kushwaha[1], Jyotsna Sinha[2]

[1]Institute of Engineering &Technology
Sitapur road, Lucknow, India
drkushwaha@rediffmail.com

[2]R.C Institute of Technology
Najafgarh, New Delhi, India
jyotsnasinha05@rediffmail.com

A. **Abstract:** ***The system needs to be auditable, reliable, and manageable from a security point of view and must provide records to the security control supervisor, so that system performance, security safeguards, and user activities can be monitored. The computer system reliability is the main criteria to choose any configuration to solve the problems. The Poisson distribution suffers from the same problems as the Exponential Distribution in that the design must be stable and random failures such as when assessing component failures on an established system. After the test, failure rates can be estimated statistically using well-proven techniques. The above indicates the reasons provided according to the source of the information, so it is necessary to define failure more precisely. To avoid confusion with the other types of failure, it is better to define those seen by the user as incidents or, where the whole computer stops, as system interruptions or system failures. With all these various types of failure, the term mean time between failures becomes ambiguous and needs clarifying.***

Keywords: reliability, bathtub curve, burn in, bedding in, wear out

1. Introduction

Since the invention of computers, their reliability [1] has been one of the major considerations and over the years with announcements of new ultra reliable technologies, one would have expected that a user would not need to worry about this area. In practice, many users are far from happy about the reliability of their system and feel that the computer supplier has mislead them with claims of high reliability and availability. Some reasons for the continuing unreliability are fairly obvious, such as the more complex and larger system, but one of the main reasons is that reliability is not fully understood. The first area of misunderstanding is the term 'failure'.

1.1 The Bathtub Curve

During training the engineers are taught that reliability follows the bathtub curve [3], as shown in the Fig.-1, indicating how the failure rate varies with time. The initial period of high failure rate is known as burn–in on the electronics or bedding–in on mechanical items and is due to manufacturing and assembly defects. This is followed by a constant failure rate during the useful life and finally the wear–out period when reliability rapidly deteriorates.

The bathtub curve does not completely apply to most of the computer equipments. The constant failure rate may not be achieved for some years, due to problems associated with the initial design not being completely correct or due to an unexpectedly long burn–in period. After this initial period, the reliability may be far from constant, with variation being caused by different utilization or other disturbance factors. General-purpose computer systems usually only have a required lifetime of 5 to 10 years, so wear-out period is not often a problem.

The system should be flexible and there along with convenient mechanisms and procedures for maintaining it under conditions of shifting job assignments, issuance and withdrawal of clearances, changes in need-to-know parameters, transfer of personnel from one duty assignment to another etc. The system should be responsive to changing operational conditions, particularly in time of emergency. While not an aspect of security [1], [2] control it is important that the system be responsive in that it does not deny service completely to any class of users as the total system load increases. It may prove desirable to design special emergency features into the system that can suspend or modify security controls, [4],[6] impose special restrictions, grant broad access privileges to designated individuals, and facilitate rapid change of security parameters [5].

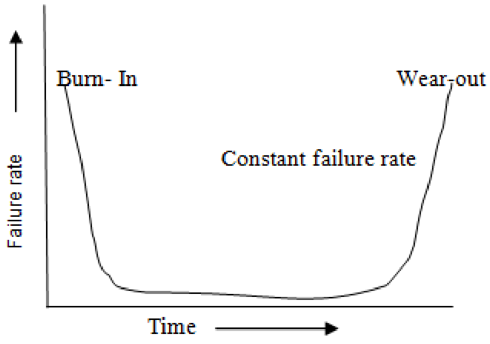


Figure 1: Bathtub curve

The system should be auditable and must be efficient to provide the records to the security control supervisor, so that system performance, security safeguards, and user activities can be monitored. This implies that both manual and automatic monitoring facilities are desirable. The system should be reliable from a security point of view. It ought to be fail-safe in the sense that if the system cannot fulfill its security controls, cannot make the proper decisions to grant access, or cannot pass its internal self-checks, it will withhold information from those users about which it is uncertain, but ideally will continue to provide service to verified users. A fallback and independent set of security safeguards must be available to function and to provide the best level of security possible under the degraded conditions if the system is to continue operation.

The system should be manageable from the point of view of security control [6]. The records, audit controls, visual displays, manual inputs, etc., used to monitor the system should be supplemented by the capability to make appropriate modifications in the operational status of the system in the event of catastrophic system failure, degradation of performance, change in workload, or conditions of crisis, etc. The system should be adaptable so that security controls can be adjusted to reflect changes in the classification and sensitivity of the files, operations, and the needs of the local installation. There should be a convenient mechanism whereby a particular user needed can embed special security controls easily in its system. Thus, the security control problem ideally must be solved with generality and economy. It would be too costly to treat each installation as an individual instance and to conceive an appropriate set of' unique safeguards [9]. The system must be dependable; it must not deny service to users. In times of crisis or urgent need, the system must be self-protecting in that it rejects efforts to capture it and thus make it unavailable to legitimate users. This point bears on the number and kinds of' internal records that the system must keep, and implies that some form of rationing algorithm must be incorporated so that a penetration would capture no more than a specified share of system capability. The system must automatically assure configuration integrity. It must self-test, violate its own safeguards deliberately, attempt illegal operations, monitor communication continuity, monitor user action, etc on a short time basis.

1.1.1 Uncertainties

The Task Force has identified several aspects of secure computer systems, which are currently impractical or impossible to assess.

1.1.2 Failure Prediction

In the present state of computer technology [7], it is impossible to completely anticipate, much less specify, all hardware failure modes, all software design errors or omissions, and, most seriously, all failure modes in which hardware malfunctions lead to software malfunctions. Existing commercial machines have only a minimum of redundancy and error-checking circuits, and thus for most military applications there may be unsatisfactory hardware facilities to assist in the control of hardware/software malfunctions. Furthermore, in the present state of knowledge, it is very difficult to predict the probability of failure of complex hardware and software configurations; thus, redundancy an important design concept.

1.1.3 Risk Level

Because failure modes and their probability of occurrence cannot be completely cataloged or stated, it is very difficult to arrive at an overall probability of accidental divulgence of classified information in a security-controlling system. Therefore, it is difficult to make a quantitative measurement of' the security risk-level of such a system, and it is also difficult to design to some a priori absolute and demonstrable security risk-level. Since the security risk probabilities of present manual systems are not well known, it is difficult to determine whether a given design for a secure computer system will do as well as or better than a corresponding manual arrangement. As described above, computer systems differ widely in the capabilities they make available to the user. In the most sophisticated and highest security risk case, a user can construct both new programs and new programming languages from his console and embed such new languages into the computer system for use. In such a computer system offering the broadest capability to the user the security problems and risks are considerably greater when users from the following two classes must be served simultaneously:

Un-cleared users over whom there is a minimum administrative control and who work with unclassified data through physically unprotected terminals connected to the computing central by unprotected communications lines.

Cleared users operating with classified information through appropriately protected terminals and communication links.

It is the opinion of the Task Force that it is unwise at the present time to attempt to accommodate both classes of users simultaneously. However, it is recognized that many installations have an operational need to serve both uncleared and cleared users.

1.1.4 Cost

Unfortunately, it is not easy at this time to estimate the cost of security controls in a computer system. Only a few computer systems are currently in operation that attempts to provide service to a broad base of users working with classified information. While such systems are serving the practical needs of their users, they are the products of research efforts, and good data reflecting the incremental cost of adding security controls to the system and operating with them are not yet available. In computer systems designed for time-sharing applications, some of the capabilities that are present in order to make a time-sharing system work at all are also applicable to the provision of security controls. In other computing systems, any facilities for security control would have to be specially installed.

Thus, the Task Force cannot give an accurate estimate of the cost of security. It will depend on the age of the software and hardware, but certainly security control will be cheapest if it is considered in the system architecture prior to hardware and software design [10]. In the opinion of some, the investment in the security controls will give a good return in tighter and more accurate accountability and dissemination of classified information and in improved system reliability. The cost of' security may depend on the workload of' the installation. If all classified operations can be accommodated on a single computer, and all unclassified operations on a second computer, the least expensive way to maintain the integrity of the classified information may be to retain both machines. Such a configuration will present operational inefficiency for those users who need to work with both classified and unclassified data bases, but the concept of a dual installation with one machine working in the clear and a second machine fully protected cannot be summarily rejected.

2. Objective

The computer system reliability is the main criteria to choose any configuration / a particular system to solve any problem. Different computer systems are available for research & development [12], [13] and general purposes by different manufacturers. Some specific systems are also designed for special use such as in Robotics, Artificial Intelligence, Satellite launch etc. Also different platforms are available to work on them. The broad objective of the study is to determine the accuracy of system response and failure rate and conditions on different application and platforms of several types of computer systems existing, as well as to predict future behaviors for specific use.

3. Limitations of Study

The present study has the usual limitation of time and resources to be encountered.

- It is not possible to cover all the computer systems existing / working in the world, but major types may be covered.
- The data will be based on the existing configuration of the system that may be altered / modified later on. So the results may vary at time to time.

In spite of above limitations, thoughtful attempts will be made to make the study as objective and systematic as possible.

4. Review of Literature

It is not the purpose of this synopsis to go into theoretical statistics in detail [9]; however, it is useful to examine some of the simpler formulae used in reliability predictions and to consider their limitations when applied to general-purpose computer equipment.

4.1 Exponential Distribution

It is usually assumed that complex equipment will suffer from a constant failure rate (1/mtbf or 1/m) and the probability of success (Ps), or probability of failure free operation, for a period t is:

$$Ps = e^{-a}$$

Where a is the expected failure or t/m. Conversely the probability of failure PF is:

$$P_F = 1- e^{-a}$$

An example of these probability calculations is indicated in Table 1.1 for equipment with a component mtbf of 1000 hours running for a period of T hours. So the probability of success of a short run compared with the mtbf is quite high, but at run duration equal to the mtbf, the probability of success has dropped to 0.368 and at mtbf there is not much chance that the period will be failure free.

Table 1.1: Probability of success (mtbf of 1000 hours)

Running time T hours	**Probability of success P_s**	**Probability of failure P_f**
100	0.905	0.095
500	0.607	0.393
1000	0.368	0.632
5000	0.007	0.993

This Exponential Distribution described above relies on the assumption that the failure rate is constant and failure random, so the equipment must be free from burn-in and design failure; also the distribution ignores the effects of intermittent faults. If it could be considered that figure 2A represents random component failures over time T, then Figure 2B could represent the effects of intermittent faults, which tend to be bunched immediately following a component failure. If a run was started when one could be absolutely sure that no intermittent faults were present, then the Exponential Distribution could apply, but if a run was started, such as at point x in Figure 2B then chances of success are pretty remove.

(i)

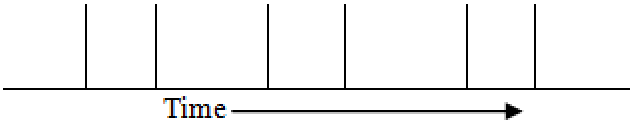


Figure 2 (a): Random Failures

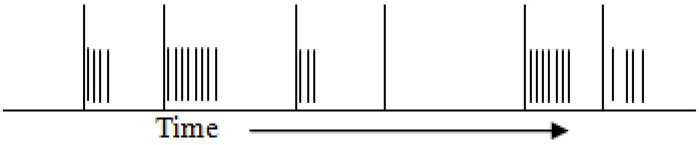


Figure 2 (b): Random failures + intermittent repeat incidents

The Exponential Distribution is used quite successfully in military or space type projects when calculating reliability of one – shot devices where the design and testing techniques may be adequate. On general-purpose computers, because of the effects of intermittent faults and other non-constant failure mechanisms, it is not recommended that this formula is used for predicting of computing payroll on a Thursday afternoon.

4.2 Poisson distribution

On equipment which is used for long periods compared with the component mtbf, it is necessary to repair them sometimes, and the number of failure expected in a period is of more use than the probability of failure; in this case, for convenience when dealing with random events occurring in a period of time, the Poisson Distribution could be used, where the probability of no failure in time t is again e-a, and full probabilities:

Probability of 0 failures $\frac{a^{0} . e^{-a}}{0!}$

Probability of 1 failures $\frac{a. e^{-a}}{1!}$

Probability of 2 failures $\frac{a^{2} \quad e^{-a}}{2!}$

Probability of 3 failures $\frac{a^{3} \quad e^{-a}}{3!}$

Probability of n failures $\frac{a^{n} \quad e^{-a}}{n!}$

When considering the number of months or weeks with 0, 1, 2, 3 etc. failures, the above are multiplied by the total number of periods T.

Table 1.2: Number of failures per month

	Class of failures	No. of months T	No. of failures	Mean failures per month (a)	Number of failures						
					0	1	2	3	4	5	6
A	System failures recorded by user	34	46	1.352	17	5	4	3	2	2	1
B	Predictions using Poisson distribution	34	46	1.352	8.8	11.9	8.0	3.6	1.2	0.3	0.1
C	Predictions using Poisson distribution for component failures	34	23	0.676	17.3	11.7	4.0	0.9	0.1		

The Poisson distribution suffers from the same problems as the Exponential Distribution in that the design must be stable and failures random, such as when assessing component failures on an established system.

An example of the Poisson distribution is shown in Table 1.2 for a period of 34 months on an established system with an incident per fault ratio of 2.0. Table 1.2a represents the distribution of incidents seen by the user, where for 17 of the months 0 incidents were recorded and for 5 months 1 incident occurred etc.

Table 1.2b indicates predictions using the Poisson distribution with the same expected mean failure per month of 1.352: the differences between the predictions and actual results are quite clear. There being twice as many periods recorded for 0 incidents and at the other extreme the recorded figures for 4, 5 or 6 incidents in a month being in excess of the predications.

Table 1.2c represents a Poisson Prediction where an average of 0.676 component failures (1.352 incidents at 2 incidents per fault) are expected per month and comparing this with the actual recorded incidents indicates the effects of intermittent faults; during the 17 months where 0 component failures are expected 0 incidents were recorded but during the other months some of the faults gave rise to more than one incident; for example of the 11.7 months with 1 faults, 5 were rectified at the first investigation but others gave 2, 3, 4, 5 or 6 incidents.

On systems with very poor maintainability and fault tolerance facilities the differences between recorded incidents and Poisson predictions becomes even greater, such that there may be a high and equal probability of 0 and 50 incidents occurring in a period. With all the foregoing problems in defining, predicting and measuring failures it is of little wonder that some contractors are reluctant to release reliability information and, for those who do, it would be equally amazing if the system behaved exactly as predicted.

However, in order for a user to choose the best configuration from the best contractor and to be able to make contingency plans for times of trouble, it is important that he understand the sort of failure pattern expected and the effects of any maintainability, diagnosability, fault tolerance or resilience features offered.

5. Theoretical Reliability Prediction

5.1 Processor Modules

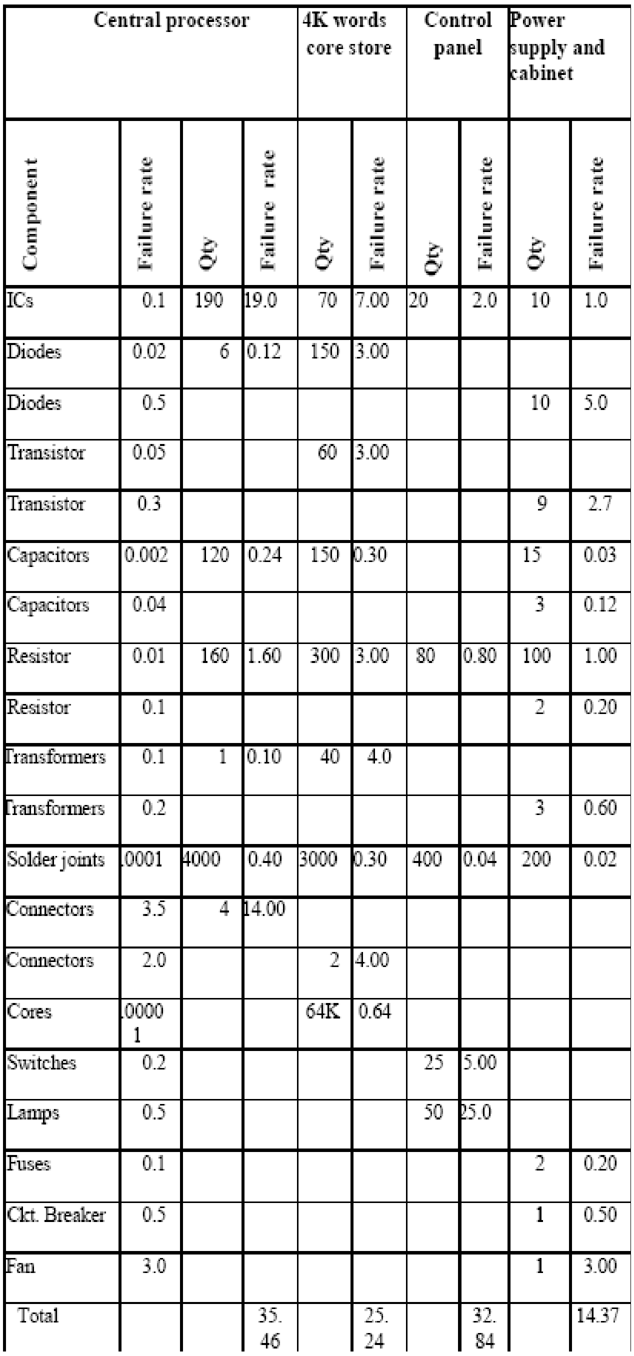
When a computer system is first designed, the manufacturer usually calculates the ultimate steady state reliability of replaceable modules, units and overall system, the calculation, being used as an aid to sorting out the initial problems or for resource allocation.

An example of failure rate calculations use the component failure rate, which usually takes into account stress factors, according to design parameters on ambient temperature, voltage and power ratings, other factors may also be taken into account, according to the environment in which the equipment will work.

More refined reliability prediction methods may be carried out on an individual component basis, rather than component on a module, and take into account complexity factors, such as the number of gates in an integrated circuit.

The predictions given in table 1.3 reflect one of the primary problems governing the use to the predictions, that is, all failures shown require engineering attention for repair but all not lead to system failure; for example, on the control panel, most of the expected failures are due to lamps, which are unlikely to cause the processor to stop.

Table 1.3: Failure rate predictions for a mini computer (Failure rates in failures per million hours)



Central processor				4K words core store		Control panel		Power supply and cabinet	
Component	Failure rate	Qty	Failure rate	Qty	Failure rate	Qty	Failure rate	Qty	Failure rate
ICs	0.1	190	19.0	70	7.00	20	2.0	10	1.0
Diodes	0.02	6	0.12	150	3.00				
Diodes	0.5							10	5.0
Transistor	0.05			60	3.00				
Transistor	0.3							9	2.7
Capacitors	0.002	120	0.24	150	0.30			15	0.03
Capacitors	0.04							3	0.12
Resistor	0.01	160	1.60	300	3.00	80	0.80	100	1.00
Resistor	0.1							2	0.20
Transformers	0.1	1	0.10	40	4.0				
Transformers	0.2							3	0.60
Solder joints	.0001	4000	0.40	3000	0.30	400	0.04	200	0.02
Connectors	3.5	4	14.00						
Connectors	2.0			2	4.00				
Cores	.0000 1			64K	0.64				
Switches	0.2					25	5.00		
Lamps	0.5					50	25.0		
Fuses	0.1							2	0.20
Ckt. Breaker	0.5							1	0.50
Fan	3.0							1	3.00
Total			35. 46		25. 24		32. 84		14.37

5.2 Overall Processor

The overall failure rate of the complete processor is calculated by adding the individual module failure rates, as shown in Table 1.4 for a processor with two sizes of main store. The standard reliability term meantime between failure (mtbf) is then determined as the reciprocal of the failure rate (x 106) giving mtbfs of 9267 and 5446 hours for the two different sizes of processor.

Table 1.4 Overall failure rate and mtbf of a mini processor

	4K word system		*16K word system*	
	No. of Units	Failure rate per million hours	No. of Units	Failure rate per million Hours
CPU	1	35.46	1	35.46
Core store	1	25.24	4	100.96
Control panel	1	32.84	1	32.84
Power supply	1	14.37	1	14.37
Total		107.91		183.63
Mtbf hours		9267		5446

5.3 Peripherals and System

Failure rates of the peripheral equipment and associated controllers can be calculated in the same manner as Table 1.3 and similarly, total system reliabilities can be estimated. Table 1.5 shows the predications for two processors with various peripherals; System A represents the smallest configuration which can be used as a computer system, indicating that the overall mtbf can be expected to reduce considerably to 894 hours due to the inclusion of a typewriter; System B with additional core store, power supplies and peripherals gives a further reduction in estimated mtbf to 352 hours.

These reliability predictions represent average values and it is fairly obvious that the manner of utilization and activity on the electromechanical peripherals could have some influence; for example, on the larger system, the typewriter is likely to be only used for operator/machine communications but, on the basic configuration, it may well be used for continuous input/output, leading to a much lower reliability. Reliability of the electronics can also vary according to the manner of utilization, especially where design faults have not been identified and rectified.

Table 1.5 Reliability Predictions for complete systems

SYSTEM A		*Failures/106 hours*
Processor with 4KW store		107.91
Typewriter controller		10.20
Typewriter		1000.00
	Total	1118.11
	System mtbf	894 hours
SYSTEM B		
Processor with 16KW store		183.63
Extra 48KW store		302.88
Typewriter controller		10.20
Typewriter		1000.00
Disk controller		15.40
Disk		256.00
Magnetic tape controller		14.30
Magnetic tape unit		345.00
Paper tape reader controller		10.50
Paper tape reader		250.00
Line printer controller		13.20
Line printer		428.00
Extra power supply / cabinet		14.37
	Total	2843.48
	System mtbf	352 urs

5.4 Software

Currently, there is no equivalent standard method of predicting software reliability, even though software problems often give rise to more concern than the hardware. However, software faults are almost entirely classified as design errors and, even for hardware, it is extremely difficult to predict meaningful failure rates for the design stabilization period.

5.5 Problems in Defining Failures

The next snag is that reliability predictions14 are only as good as the basic failure rates used. For any component a multitude a failure rates can be found from various sources, for example anything from 0.01 to 0.4 per million hours for integrated circuits. The reasons for the wide variations in failure rates lie between the method of measurement and the definition of a failure.

It must be appreciated that when considering processor components with such a low failure rate, many millions of device hours must be clocked up to provide the true figure; on initial manufacture of a new component, the failure rate in the field may not be available for a long time after equipment has been designed using it. So the component supplier may provide an initial estimated based on experience of components produced by the same manufacturing techniques10.

A more usual method of providing the initial estimates of failure rates is for the component supplier to carry out tests of a large batch of components for fairly long period at temperature and other design limits. Any component failing during the test will be removed from test and analyzed to find out what a slight change to manufacturing processes will overcome the problem. After the test the failure rates can be estimated statistically using well-proven techniques.

This method of testing a batch of components and immediately removing, any which are indicated as faulty leads to the first concept of a failure; that as a faulty component can only give one failure before it is replaced. The failure rates derived from the component batch testing can be realistic but they are dependent, firstly, on whether the component suffers from some hidden failure mode which was not revealed by the particular method of testing; secondly they assume that the components will always be used within the design specification; and thirdly, that the components will be correctly screened during quality control tests to weed out any which have been incorrectly manufactured.

For military or space type computer application with generous allowing extra special care to be taken, the lowest component failure rates may be achieved ([7], [13], [14]). On general purpose computer systems with limited budgets and necessary mass production lines, it is unlikely that the lowest failure rates will be obtained in practice and time not money will be available to bring the reliability within specification. So for these systems the manufacturers are likely to adopt more conservative failure rates.

The next method of assessing component or system failure rates, which has been adopted by a number of small computer manufacturers, is to run a number of systems for a year or again examining every failure in detail and making assumptions about changes in procedures or design to overcome the problem, and thereby producing the magic figure. Depending on assumptions made and testing techniques used, a wide variety of results can be obtained. A few years ago a mini-computer processor with about 300 integrated circuits was produced with an initial design mtbf of about 4000 hours according to calculations based on the component count and fairly low failure rates. Later, based on a life test of a number of systems, the mtbf claims were increased to 10,000 hours. Later, when feedback was obtained from systems working in the field, it was found that, from a users point of view, mtbfs varied between 100 and 2000 hours with an average of about 300 hours. At the same time feedback to the manufacturers indicated an average mtbf of more than 600 hours. The wide variations between different installations were found to be due to design deficiencies, and are considered later, but why the difference between the averages figures seen by the users and the manufacturer?

5.6 Failure Pattern

The concept of a failure mentioned was that one faulty component can only give one failure but, in practice, this is not true, as 50% or more of component faults are of an intermittent or transient nature; these faults tend not to be readily reproducible by standard test techniques and the user may not bother to call the engineer every time. This leads to a typical failure pattern as indicated in Table 1.6, where 50 faulty components can lead to 100 engineering investigations where parts are changed which are not really faulty. On top of this are 50 abortive investigation by the engineer where ‘no fault found’ is indicated by the test programs and 200 further indicated where the user does not call the engineer or prefers to restart and carry on working without incurring long period of investigation time. In this case, assuming that 50% of the faults are intermittent, the 25 solid faults will give rise to 25 system failures, which must be repaired before operation can continue. On the other hand the 25 intermittent give rise to 325 incidents or 93% of system failures are due to intermittent faults.

The above indicates the reasons for different for mtbf figures being provided according to the source of the information, so it is necessary to define failure more precisely; also each of these mtbfs has its own use.

Table 1.6: Practical reliability of a central processor

Class of failures	*No. of failures in 10,000 hours*	*Mtbf hours*	*Proper definition of mtbf*
Failures seen by the user	350	28.6	Mean time between system failures (mtbf) or mean time between system interruptions (mtbsi)
Investigations by the engineer	150	66.7	Mean time between engineering investigations or service calls
Parts replaced or repaired in situ	100	100	Mean time between repair attempts
Parts replaced and actually found to be faulty plus genuine repairs in	50	200	Mean time between component failure or component mtbf

5.7 Component failure

This first concept can be used to define failures, where one faulty component gives one failure. The figures are used for predicting inherent reliability by the design authority and when faulty components are identified, for establishing that the design reliability has been achieved; or to identify components, which are not meeting the design criteria.

5.8 Repair attempts

For replaceable items, these failures are used to determine spares holdings and resources required for the testing and repair depot.

5.9 Engineering investigations or service calls

These failure are usually the ones provided by manufacturers when indicating reliability of equipment in the field; they are the once most easily derived as the manufacturers usually have some fault reporting procedure for every investigation. The figures are used for planning maintenance manpower resources. The ratio engineering investigations divided by component failures (Investigation per fault) is a useful measure of how easy it is for the engineer to reproduce and rectify faults.

5.10 Incidents

To avoid confusion with the other types of failure, it is better to define those seen by the user as incidents or, where the whole computer stops, as system interruptions or system failures. Another useful measure is the ratio incidents per fault, this time also reflecting fault tolerance or resilience features of a system. The incidents per fault ratio for the system shown in Table 1.6 are 7.0 which are again not too good, although figures of greater than 10.0 have been recorded. At the other extreme it is possible to achieve less than 1.0 incident per fault where the fault tolerance features keep the going until scheduled maintenance periods, when the faults are investigated and cleared.

6. Conclusion

The number of incidents per fault is also dependent on the nature of the problems and user attitudes. Some users opt to attempt to restart for almost every incident but others will insist that the engineer investigates every time, reducing the overall number of incidents but increasing investigation times. With all these various types of failure, the term mean time between failure or mtbf becomes ambiguous and needs clarifying, otherwise, as indicated in Table 1.6, the mtbf of a system could be correctly given as 200 hours or 28.6 hours, where the former really indicates mean time between component failure.

References

[1] James P. Anderson, Computer Security Technology Planning Study, ESD-TR-73-51, ESD/AFSC, Hanscom AFB, Bedford, MA 01731 (Oct. 1972) [NTIS AD-758 206]

[2] David E. Bell and Leonard La Padula, Secure Computer System: Unified Exposition and Multics Interpretation, ESD-TR-75-306, ESD/AFSC, Hanscom AFB, Bedford, MA 01731 (1975) [DTIC AD-A023588]

[3] Grace H. Nibaldi, Proposed Technical Evaluation Criteria for Trusted Computer Systems, M79-225, The Mitre Corporation, Bedford, MA 01730 (Oct. 1979)

[4] Roger R. Schell, Peter J. Downey, and Gerald J. Popek, Preliminary Notes on the Design of Secure Military Computer Systems, MCI-73-1, ESD/AFSC, Hanscom AFB, Bedford, MA 01731 (Jan. 1973)

[5] W. L. Schiller, The Design and Specification of a Security Kernel for the PDP-11/45, MTR-2934, The MITRE Corporation, Bedford, MA 01730 (Mar. 1975)

[6] Willis Ware, Security Controls for Computer Systems (U): Report of Defense Science Board Task Force on Computer Security; Rand Report R609-1, The RAND Corporation, Santa Monica, CA (Feb. 1970)

[7] Aeronautical Radio Inc. (ARINC) Reliability Engineering. Prentice Hall, Englewood Cliffs, NJ, 1964.

[8] K.K. Agarwal, K.B. Mishra, and J.S. Gupta. Reliability evaluation, a comparative study of different techniques. Microelectronics and Reliability, 14(1: 49-56), 1975.

[9] Roy Longbottom, Computer System Reliability-Large Scientific Systems Branch, Central Computer Agency: 1-18, 1980.

[10] Peter G. Neumann, L. Robinson, Karl N. Levitt, R. S. Boyer, and A. R. Saxena, A Provably Secure Operating System, M79-225, Stanford Research Institute, Menlo Park, CA 94025 (June 1975)

[11] Philip Myers, Subversion: The Neglected Aspect of Computer Security, Master Thesis. Naval Postgraduate School, Monterey, CA 93940 (June 1980)

[12] Theodore A. Linden, Operating System Structures to Support Security and Reliable Software, NBS Technical Note 919, Institute for Computer Sciences and Technology, National Bureau of Standards, US Department of Commerce, Washington DC 20234 (Aug. 1976)

[13] Ford Aerospace, Secure Minicomputer Operating System (KSOS): Executive Summary Phase I: Design, Western Development Labratories Division, Palo Alto, CA 94303 (April 1978)

[14] Department of Defense, Trusted Computer System Evaluation Criteria (Orange Book), DoD 5200.28-STD (1983, 1985)

Author Profile

Dr. D S Kushwaha received the degrees, M Sc (Physics) from CSJM University and worked for his PhD in LCD at IIT Kanpur. Dr. D S Kushwaha received the degrees M Sc (Physics) from CSJM University and worked for his Ph D in LCD at IIT Kanpur. He completed PhD (IT &System Engg) from Lorenz University U.S.A. and Doctor of Computer Science (D CSc) in Information Technology from Cambell State University U.S.A. He worked as a Professor of Computer Science & Information System at Ministry of Higher Education, Kingdom of Saudi Arabia (K.S.A). Presently he is working as System Manager & Senior Facility at Institute of Engineering & Technology, Lucknow (India).

Jyotsna Sinha received her Master’s Degree in Computer Science in 2002. She is currently pursuing her Research work in Liquid Crystal Display Technology and also serving as Director in RC Institute of Technology, New Delhi.