*Journal of Science and Science Education*
Vol. 5 No. 1 May 2021 36–43
ISSN: 2598-3830 (Online)

# A survey and study on modern OS reliability and security

**Muhammad Aqeel**

*Department of Computer Science*
*Bahria University Karachi Campus*
*Karachi, Pakistan*

## ABSTRACT

Current studies in operating systems focuses on either security or on reliability. However, current or modern OS platforms call for answers that match both styles of requirements. If we build not only best applications but also middleware and OS in a similar way, we can construct systems that not only are inherently stable however also can resist attacks from malicious programs and face up to errors. In this paper, we conduct an exploratory study on Modern OS reliability and security. We examine many real OS failure facts collected from distinct administrative and real time environments and present the results which show a number of the primary reasons of OS failures.

**Keywords**: failure types; kernel; memory management; operating system; reliability; security

**Received:** June 12, 2021; **Accepted:** July 7, 2021; **Published online:** September 22, 2021
*Corresponding author: muhammadaqeel7946@gmail.com

**Citation:** Aqeel, M. (2021). A survey and study on modern OS reliability and security. *Journal of Science and Science Education*, *5*(1), 36–43.

## 1. Introduction

Current OS have extreme security issues. Some of the reasons for this case consists of the fact that we're continuing to construct software in the way it has always been made from day one; that is, looking to do it in a short time and using poor development methodologies. We need to find a way to construct computer applications designed to be stable and reliable from the beginning. We can nevertheless use standard OS, and databases etc. We just ought to broaden all new variations of them in a systematic manner. In simple words all things i.e. user applications, OS, databases, and other system applications are all constructed the use of a similar old technique which leads to many loopholes and security issues in them. Similarly, majority of excessive-reliability programs want to run on the pinnacle of the (OS) software. Since these programs rely upon the OS facilities, if the OS layer does not assure, at least, the same stage of reliability anticipated for the user application, the whole system reliability is compromised. From the user perspective, it does not count if a failure happens at application or OS level, in both cases the person's experience is a system failure. Hence, increasing the OS reliability is likewise a major requirement toward the reliability of computing structures as an entire consisting of safety.

Modern OS face many challenges to assure high reliability. A nice way to have secure structures is to construct programs in a scientific manner where security is a crucial part of the life cycle. The same applies to reliability. If we need a system which is stable and reliable, both security and reliability need to be constructed together. If we build now not simplest applications but

additionally middleware and operating systems inside the identical manner, we are able to construct systems that not best are inherently secure but also can resist attacks from malicious programs and withstand errors. All protection and reliability constraints need to be defined inside the application level, where their semantics are understood and propagated to the lower tiers. The lower tiers offer the assure that the limitations are being found. We call this method a "high-level security structure", in which all safety constraints are defined on the conceptual or application level. The lower degrees just put in force that there aren't any techniques to skip these constraints. By mapping to a highly stable platform, e.g., one the use of skills, we're able to produce a totally stable gadget. We make a case for this approach and we speak which factors are required make it practical. Since security impacts reliability and vice versa, we need tactics to integrate these two in a coherent way.

In this paper, we have an in depth look at and conduct a study/survey to characterize special components of OS disasters.We check OS failure records of unique real world scenarios with a number of utilization profiles, which are being used in various organizations. We derive OS reliability and security functions on the basis of the results of these statistics for OS failure classes.

## 2. Background

To our know-how, no preceding study has aimed toward assessing both reliability and security together for the betterment and high overall performance of OS. However, numerous works have focused on entirely either one of these factors to asses and conclude different factors and statements.

In Microsoft (2013), authors state that OS kernel extensions (e.g., device drivers) accounted for over 70% of the Linux kernel code, at the same time as Windows 10 provided more than 35,000 special device drivers. They concluded that third-party extensions had been a main reason of Windows 10 failures, wherein drivers have been responsible for 85% of mentioned failures. Similar findings have been determined in Linux (Chou, 2001), in which device driving force errors have been appreciably extra widespread than different elements of the Linux kernel.

In Ganapathi & Patterson (2005), a number of OS disasters collected from Windows computers in an academic environment have been analyzed. Only OS disasters in kernel degree had been taken into consideration. The authors concluded that kernel failures have been much less conventional than software disasters, but with higher effect in terms of unplanned downtime.

In Ganapathi *et al*. (2006), many occasions of Windows 07 kernel crashes were analyzed. Similar to Ganapathi & Patterson (2005), simplest kernel-level failures had been analyzed. The effects corroborate the study provided in Microsoft (2013), emphasizing that device drivers contribute most for OS crashes and failures.

Malallah (2021) emphasized that Android, IOS, OS X, Windows 10 are more stable & reliable in term of performance, user friendliness and security unlike other OS systems. Sulaiman & Raffi (2021) compared the performance between Linux and Windows 10 by checking how many resources tasks are using before start and during experimentation.

Kalyanakrishnam *et al*. (1999) analyzed failure records from Windows 8 primarily based servers, collected over six months. The results showed that, in average, the servers' uptime turned into

283.68 hours. Similarly, Li *et al*. (2008) investigated the reliability of Windows 8 servers. They collected failure statistics from 503 servers over 4 months. They considered as OS failures all unexpected activities main to a system reboot/crash/halt. They observed the MTBFs resulting from hardware (ninety two.74 hours), applications (31.Fifty two hours), device configuration (13.61 hours), and preservation (5.92 hours). This takes a look at differs from the above stated as summarized next.

The previous researches investigated OS disasters occurring due to either reliability or security only. Differently, we take into account OS failures on both levels, given that no matter the OS additives fail due to reliability or security, in both instances the user's work experience is affected. In precise, maximum of previous works and most of these instances analyzed either one of them whereas both of reliability and security measures represent integrally the real users' notion concerning the performance and work experience of Modern OS.

## 3. Methodology

The analysis of the failure dataset of OS which has been analyzed in this research is presented in this section, with the methods adopted as well for calculating the statistics.

### *3.1 Approach used for data sampling*

On the basis of unique organizational surveys (e.g., Bott (2013), Mood *et al*. (1974), Murphy (2008)) on different working structures, we discovered that Windows 10 (Win10) is currently the maximum used desktop working device; it has extra than 45% marketplace percentage followed with the aid of others Windows own family OS (Win8.1, Win8, and Win7). Thus, we determined to awareness on OS failure styles accrued from Win10 structures. This OS gives distinct failure registries through its RAC (Reliability Analysis Component) (Microsoft, 2010), that's automatically enabled for the duration of the Win10 set up. A singular RAC's stats file consists of all disasters occasions registered because the OS installation date. We accumulate this document from different computers to conduct our look at.

We adopt two techniques for collection of our stats. Primarilyy, we ourselves make replica's of those RAC files, which require neighborhood get entry to to the surveyed computers. For every copied RAC report, we represent the surveyed systems. In the next step, we will be extracting the registries of failure which are of interest from the collection of RAC files. We notice that the RAC files keep not most effective OS disasters, however additionally other failure activities of software program running in the pc. Therefore, we filter out the RAC files content material. For initiating this filtering process, we have created 3 categories of OS disasters, mentioned in Table 1.

**Table 1**. Filtering options used.

| ID | Entity Name | OS Failure Type |
|---|---|---|
| 1000 | Application Error (firefox.exe) | Non Applicable |
| 1000 | Application Error (explorer.exe) | OS Application |
| 1002 | Application Hang (mmc.exe) | OS Application |
| 1020 | WindowsUpdate client | OS Service |
| 1037 | Windows StartupRepair | OS Kernel |

The first elaborates the failures because of malfunctioning of OS application processes, which run at consumer stage. The second class includes disasters from OS provider additives. The third class consists of failures because of OS Kernel subsystems, which are vital, typically require machine reboot, and are strongly associated with the system builtin libraries. In order to music the filtering regulations for those 3 categories, preliminarily we done a guide category of OS failure events which observe a specific sample of the gathered RAC documents. For this motive, we rely on the subsequent fields of RAC. These fields permit us to discover if the supply of the failure occasion suits to one of the 3 classes taken into consideration. The first discipline specially is a integer ID that identifies the unique windows occasion type associated with the registered failure. Table 1 indicates examples of this class.

We notice that failure activities which are related to OS Applications are clearly not enough to differentiate OS from non-OS (user) packages. Therefore, we additionally research the subject for Product Name. For example, in Table 1, the two events of failure with Event ID and Source Name identical to a thousand and “Application Error”, with admire to their Product Name are evaluated, respectively. Based on the onsite and online accrued records, we surveyed different computers resulting in a number of OS disasters. We arranged the statistics sets in six companies (A1 to A6). The disasters for the first five groups were gathered onsite. Failures in A6 have been amassed via the net shape.

**Table 2.** Per investigating group profile usage.

| Group | Place/purpose of work | Machine type | Type of apps. |
|---|---|---|---|
| A1 | Academic | Desktop | Office apps., graphic editing |
| A2 | Academic | Desktop | Office apps., software & web development, multimedia, Scientific & engineering |
| A3 | Academic | Desktop/ Laptop | Office apps., graphic editing, Software & web development |
| A4 | Corporate | Desktop/ Laptop | Office apps., graphic editing, Software & web development |
| A5 | Corporate | Desktop/ Laptop | Office apps., software & web development |
| A6 | Academic, Corporate, Home | Desktop/ Laptop | Office apps., graphic editing, software & web development, multimedia, scientific & engineering, games, ERP apps., point of sale, antivirus, servers: database, web, application, email, directory, and file/printer |

Table 2 consists of six groups. Groups A1 and A2 comprise OS disasters from computers deployed in two one-of-a-kind universities, respectively. In both corporations the surveyed computer systems are used by college students to run general-purpose programs (e.g., text enhancing) and teaching laboratory training (e.g., engineering packages). A3 and A4 are also from university surroundings. However, each records units come from the identical group. A3 carries OS disasters from computer systems of a coaching laboratory with comparable usage profile present in A1 and A2. Differently, A4 businesses failures from computer systems deployed in college’s administrative places of work, now not associated with practical lab teaching programs. The OS failures which are grouped in A5

come from a corporate place of job. The first 5 corporations that are being used are based for the purpose failure statistics are of the identical environments. A6 is different form others as it is a heterogeneous institution, i.e., it is includeing OS failure records from one of a kind workplace environments, varying from corp-orates, teachers, and domestic computer systems. A unique evaluation of every institution is supplied in Section 4.

### *3.2 Analysis of data and its approach*

First of all, we performed characterization of every individual group, resulting in calculation of their details with information. For every organization, we have tested all the samples to become aware of the best model according to each computer. Assuming the fact that all the computers are of the same type or category institution (except A6) are occupying identical utilization profiles and are processing and running under the same workload, we have additionally clustered the samples of failure and have carried out the GoF (Goodness-of-Fit) checks to pick out the distribution function per group. In essential failures, the device crashes commonly and the pc is required to be rebooted. OS services / packages failures require, restarting the failed element at least minimum. In those cases, the systems studied is considered as repairable.

### *3.3 Performance comparison between Windows 10 and Linux*

All the parameters are documented for experimentation .Performance is checked for both OS based on these parameters like how much time software is taking to start in both OS , how much resources and CPU (Central Processing Unit) is acquired by software, etc. Comparison will be made for both OS types based on these parameters.

**Table 3**. Activities and parameters used to test performance in windows and linux (Sulaiman & Raffi, 2021).

| Activity | Results |
|---|---|
| Identify OS version | Windows 10 & Linux Mint |
| Identify software | Discord, Steam, Chrome, Firefox |
| Identify Parameters | Resource utilization, Time consumption |

## 4. Results

As described in previous section, first of all we have characterized failure samples of every OS group. Assessing that our samples of failures are being saved in RAC files which are being extracted from unique computer systems, it is very crucial to perceive duration of the RAC registered the failure events. Below Table depicts and shows all these period consistent with their institution. As it can be seen, all failure activities are of recent times, except for A6 which consists of registeries older than one year. This thing makes our results well applicable for use and for modeling.

Next, we compute descriptive facts for every institution to be able to have their normal failure characterization. We have a deep look at toset OS Services which have the best percentage of OS failures in all organizations; these percentages are quite near to many of the other groups. The OS Kernel disasters surprisingly don't present the minorest chances as expected. We hypothesize that their poor reliability as compared with all of the other businesses may be due to be drawbacks of the pc system management policy which are applied to the business enterprises. Figure 1 represnts

the evaluation stated in above sections. The no: of computers surveyed are visualized in the x-axis following a given usage profile in the y-axis. The bars show the proportion of each and all computers with their usage profile.

**Table 4**. Period of failure sampling of each group.

| Groups | Year of Sample |
|---|---|
| A1 | 28/02/2020 to 12/04/2020 |
| A2 | 17/06/2020 to 19/07/2020 |
| A3 | 20/10/2020 to 21/09/2020 |
| A4 | 20/04/2020 to 21/09/2020 |
| A5 | 15/02/2020 to 05/12/2020 |
| A6 | 25/10/2019 to 06/04/2019 |

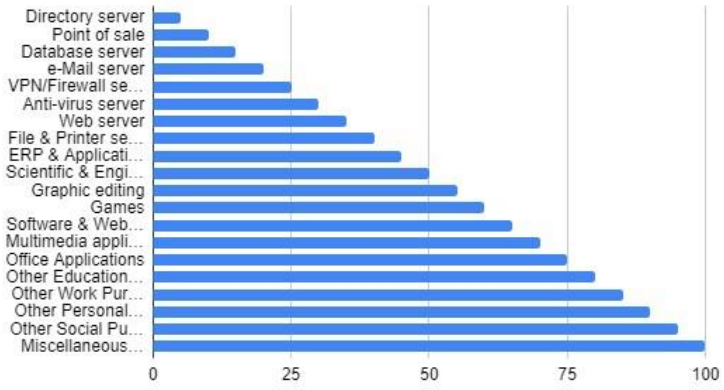


**Figure 1**. Characterization of A6 usage profile.

Let us now discuss comparison experiment results which are made to check which OS is better in terms of performance between Windows 10 and Linux.

1. Time consumption
   Time consumption test is ran on Windows 10 and Linux to check which OS open software files more faster. Linux has got advantage over windows 10 because of file system used by Linux in which files are placed very closed to one another.
2. Resource usage
   Before start of experiment resource usage in Windows 10 and Linux are recorded for accuracy of results in idle state. All software are started in both OS and it is seen Windows 10 is taking more resources due to large number of background process on the other side linux only have small number of background process. Background process tends to impact OS performance .

Let us now discuss the results of security measures which we found out in our analysis of data. Below mentioned is the time taken for encryption and decryption for different bytes of data that includes all types of data, i.e. images, text-files, plain text, etc. As seen in Table 5, we can clearly see that security is also a concern as it is also taking time as well the level of security is also not that good as due to which there are also many OS failures occurring.

**Table 5**. Time taken for encryption & decryption for different data bytes.

| Input (byte) | Encryption Time (ms) | Decryption Time (ms) |
|---|---|---|
| 5000 | 25 | 41 |
| 10,000 | 28 | 47 |
| 15,000 | 29 | 48 |
| 20,000 | 30 | 48 |
| 25,000 | 32 | 53 |
| 30,000 | 33 | 54 |

## 5. Conclusion

In this work, we perform a study and survey on OS reliability and security, primarily based on a number of real OS failures accumulated from distinct computer systems distributed in six exceptional place of business environments. Analyzing the data sets with the usage of exclusive statistical strategies, we have been able to perceive constant OS disasters patterns for different failure classes that were found after investigation. Overall, we can say that OS services are in reality the main cause of OS disasters in the computers surveyed in reality.

Rate of disaster in each OS Kernel & OS Applications classes are minute and quite similar. These findings are very critical, considering the fact that during company offices and the customers do not deal with OS services now immediately, which might be maintained by device admins.

Our suspect is that the difference is porportional to the uptime of the system. So, thinking the common uptime in a practical lab is assumed to be low, because of common and regular system restarts, the better uptime in organizations can promote high probabilities of failures of OS kernel. In the end, we have estimated the pleasant-in shape model of statistics for the 3 categories which are investigated for OS disasters. All these outcomes and results which are acquired are based totally on real records, offer practical modelling which may be used in special studies and researches for the purpose of simulation and modelling of computer systems.

## References

Bott, E. (2013). Latest OS share data shows Windows still dominating in PCs, ZDNet.com, Apr. 2013. Retrieved from http://www.zdnet.com/latest-os-share-data-shows-windowsstilldominating-in-pcs-7000013351/

Chou, A., Yang, J., Chelf, B., Hallem, S., & Engler, D. (2001). An empirical study of operating systems errors. *Proceedings of the ACM symposium on operating systems principles*, 73–88.

Ganapathi, A., & Patterson, D. (2005). Crash data collection: A Windows case study. *International Conference on Dependable Systems and Networks*, 280–285.

Ganapathi, A., Ganapathi, V., & Patterson, D. (2006). Windows XP kernel crash analysis. *Large Installation System Administration Conference*, 149–159.

Kalyanakrishnam, M., Kalbarczyk, Z., & Iyer, R. (1999). Failure data analysis of a LAN of Windows NT based computers. *Proceedings of 18th Reliable Distributed Systems*, 178–187.

Li, P. L., Ni, M., Xue, S., Mullally, J. P., Garzia, M., & Khambatti, M. (2008). Reliability assessment of Mass-Market Software: Insights from Windows Vista. *Proceedings of the 19th International Symposium on Software Reliability Engineering*, 265270.

Microsoft (2013). *Windows error reporting*. Retrieved from http://msdn.microsoft.com/en-us/library/bb513613(v=vs.85).aspx

Mood, A. M., Graybill, F. A., & Boes, D. C. (1974). *Introduction to the Theory of Statistics*. McGraw-Hill.

Murphy, B. (2008). *Reliability estimates for the Windows operating system*. Microsoft Research Cambridge. Retrieved from http://www.dcl.hpi.uni-potsdam.de/meetings/mshpsummit

Malallah, H. (2021). Comprehensive study of kernel (issues and concepts) in different operating systems. *Asian Journal of Computer Science and Information*, *8*(3), 16–31.

Sulaiman, N. S., & Raffi, A. S. H. A. (2021). Comparison of operating system performance between Windows 10 and Linux Mint, *International Journal of Synergy Engineering and Technology*, *2*(1), 92–102.