Review Article

A Comprehensive Study of Kernel (Issues and Concepts) in Different Operating Systems

.

ABSTRACT

Various operating systems (OS) with numerous functions and features have appeared over time. As a result, they know how each OS has been implemented guides users' decisions on configuring the OS on their machines. Consequently, a comparative study of different operating systems is needed to provide specifics on the same and variance in novel types of OS to address their flaws. This paper's center of attention is the visual operating system based on the OS features and their limitations and strengths by contrasting iOS, Android, Mac, Windows, and Linux operating systems. Linux, Android, and Windows 10 are more stable, more compatible, and more reliable operating systems. Linux, Android, and Windows are popular enough to become user-friendly, unlike other OSs, and make more application programs. The firewalls in Mac OS X and Windows 10 are built-in. The most popular platforms are Android and Windows, specifically the novelist versions. It is because they are low-cost, dependable, compatible, safe, and easy to use. Furthermore, modern developments in issues resulting from the advent of emerging technology and the growth of the cell phone introduced many features such as high-speed processors, massive memory, multitasking, high-resolution displays, functional telecommunication hardware, and so on.

Keywords: Operating System, Microkernel, Kernels Issues and Concept, Android OS, open-source OS.

1. INTRODUCTION

The OS is a bunch of specially developed programs running on a computer system that authorizes it to operate appropriately. The OS is designed to obey two primary purposes: (1) It manages the allotment and usage of the computer system's resources among the different tasks and users. (2) imparts an interface between the computing hardware and the developer, making it easier and simplifying it for application programs to be programmed, generated, and debugged [1].

As OS became more prominent and more complicated, interest in rational segmentation of the program grew. Comprehensive OS functions and user support will be built on top of this skeletal software base. The kernel provides all else on the machine with critical facilities and defines many of the features of higher applications. Thus, as a synonym for "kernel," we also use the word "operating system OS." [2].

In a modern general-purpose machine, the operating system kernel has the highest degree of privilege [3]. The kernel governs how scarce resources such as CPU running time and physical memory pages are used by processes on the device and arbitrates access to protected hardware, as shown in fig. 1. The kernel is the component that allows a process on the system to access files, the network, or display configuration data. The Operating System has two primary functions: it essentially needs to be used as an extension machine.

As a computer system manager, it has to handle and administer all sorts of tools reasonably. Furthermore, specific systems are responsible for protecting the computing system and offering application-specific services like networking, graphical interface, etc. [4-6]. Amongst the most challenging aspects of research are security monitoring and ensuring that no new bugs have been implemented. Until merging with the mainline branch, kernel developers try to identify as many security problems as possible. Failure to identify vulnerabilities can result in insecure kernels and systems becoming distributed. Multicore is one of the most critical trends to improve the efficiency of processors. The current leadership producers are therefore focused on becoming multicore processors (MCP) [7]. Improvement of the computer capacity multitasking is one of the main benefits of MCP. These processors provide only a few full-running cores rather than one, each with a separate front-side bus interface [8, 9].

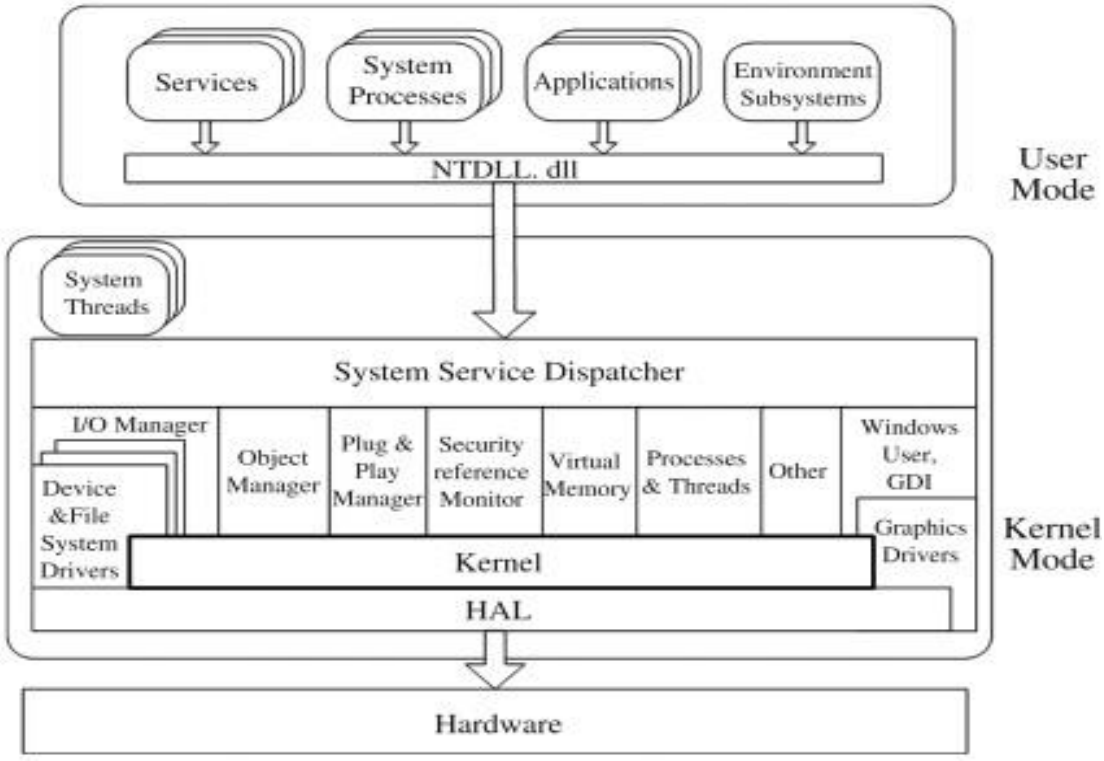

Fig. 1. The abstract view of a kernel [6].

Different kernel structure designs exist. Monolithic kernels are running entirely within one address space, cooperating with the CPU operating, primarily for speed, in the supervisor mode. As user processes do, microkernels run most of the time but not every service is used in the user area, primarily for durability and modularity. Service providers are complicated social and deliberate processes to do something. With Enterprise, we mean any business company, company, organization, and any formal or informal monarch. We mean a social body with a purposeful undertaking [10, 11].

The kernel itself offers only basic functionality in the microkernel address that enables the implementation of separate programs and servers that take former kernel roles, for example, system drivers, GUI servers, etc. The mobile devices with operating systems, which are among the most common user devices, provide various communication interfaces between the application layer software components and hardware devices [12]. Today, these devices

provide us with a significant number of services, such: voice calls, messaging, cameras, internet browsers, games, video players, and many others[13]. However, each mobile phone must include some mobile operating system to execute these services [14].

The problem statement of this review research, is presented through two distinct perspectives: detailing several concerns related to the types of kernels and merits used and evaluating how novel technologies are evaluated, and assessing underperformance. The primary purpose is to study different papers related to kernel issues on various types of OS used in different types of computers/smartphones and provide a brief review of these studies.

The rest of the review paper is organized as follows. Section 2 presents types of operating systems; in section 3, kernel issues and concept. In section 4 presents some literature reviews, and the discussion is summarized in Section 5. Finally, Section 6 outlines the conclusion.

1.1 Operating Systems Controlling of Processes Execution and Scheduling

The OS software is a device software that controls the hardware, software, and services of the computer program [15]. Timeshare system plans activities to use the system efficiently, which can also involve CPU allocation cost allocation tools, mass storage, printing, and other services [16]. The operating system serves as an interface between programs and the computing hardware to use hardware such as input, output, and memory allocation [17]. At the same time, the application code is generally run by the hardware directly and sometimes calls to or interrupted by the operating system function. Many computer-containing products – mobile phones and consoles for video games, web servers, and supercomputers – have operating systems.

1.2 Operating Systems Role on Applications and Computations

An OS is the machine software that manages computer hardware and software resources and allows various applications. These technologies can be linked with cloud computing, intelligent device applications, deployment of company systems, Web servers' performance, etc. [18, 19].

1.2.1 Cloud Computing Influence on Operating Systems

A cloud is a category of the operating system designed to work in a cloud computing network and virtualization[20]. A cloud operating system controls the service, execution, and proceedings of virtual computers, virtual servers, virtual infrastructure, hardware, and software backend [21]. Several systems are used in cloud computing technology, and most of them are implemented and used in particle physics, data retrieval, etc. However, different approaches are used to improve cloud computing performance. The word "cloud" is common in some organizations but not fully comprehensive and valuable [22].

The emphasis in the IT world has now been cloud computing. It provides individuals and organizations with robust computing services through the Internet and gives them access to a pool of standard tools, including storage servers and applications [23]. Businesses of all sizes are increasingly embracing cloud systems because they get to purchase hardware and software services at no expense but just pay for each use. This means that they are providing huge advantages, including cost savings [24]. There are various levels of cloud architecture in which each level allows extra user power. In addition, a decent operating system is essential, and conventional operating systems cannot fulfill all cloud requirements. Unique operating systems that can handle cloud requirements also need to be developed[25, 26]. Cloud computing will recently be described as the latest, commonly scalable method of providing different applications, services, and data storage [27]. Unfortunately, cloud computing does not have many systems with secure platforms with all

these benefits. Many of these missing cloud computing side tools and approaches have been used [28].

The Web is this era's most evolving forum. It is a chain of interconnected Internet hypertext records containing diverse information in text, pictures, videos, and other materials. The Internet and Web are two different words, sometimes used together incorrectly [29, 30]. Internet is an extensive network of linked computers using the TCP/IP Protocol Suite and the Web, including emails and many other applications, is currently available on the Web [31]. Fast-speed Web technologies have given access to different architectures and have helped move to web hypermedia systems [32]. The idea of cloud computing is evolving very quickly with progress in internet technology. Cloud computers allow users who are located with any device over the Internet to access their data and applications through Web browsers [33]. Cloud computing can contribute to application collaboration and to reducing platform compatibility dependence. The massive changes to cloud infrastructure and web-enabled mobile applications have impacted conventional business processes[34].

1.2.2 IoT Influences on Operating Systems

The Internet of things, which is undoubtedly the most common technology today, lies behind the future of communications [35]. IoT implementations range from widespread speech recognition to vital space programmers. Several attempts have been made to develop IoT systems since the demands of heterogeneous IoT implementations are not met either with standard Windows/Unix or with modern Real-Time operating systems [36]. The Internet of Things (IoT) enables users to connect trillions and share intelligent machines and information, tracking and monitoring resources for home automation systems, related services, healthcare, agriculture, security surveillance, energy grid, or critical infrastructure management [37]. The dynamic digitalization of physical components ready to provide value-added applications for mobile devices constantly reduces the boundaries between artificial and natural environments [38]. The IoT is operated by a specific software that feels, controls, and changes things. This life-saving invention would develop into a collection of linked artifacts that permit surgeons to perform remote procedures and persons and to assess their homes and power suppliers [39]. Lead the facilities in a reasonable manner and a sensitive national security condition. During the year 2020, the proliferation of intelligent (IoT) devices from the multiplicity of IoT use technologies was projected to rise sharply to 20.4 billion by the end of the year [40]. With the growing number of heterogeneous products connected to IoT and data generation, power and bandwidth are becoming very difficult for the IoT to allocate to tasks effectively. This view has been designed to integrate cloud computing and IoT [41].

1.2.3 Web Servers Influences on Operating Systems

An operating system, fast CPU, high memory material, specific hardware for particular purposes, running programs, and few Web pages, etc., are a general webserver [42]. These web servers are built with general use computers and use various operating systems like Unix, Linux, Windows, etc. In these implementations, the client accesses the servers via the LAN router and the Internet, and the traditional client-server Configuration is emphasized [26]. The client transmits a message to the server to connect to the Internet through the router. The Web manages the query and links eventually to the desired web server from which the requested data is transmitted to the client. An integrated web server is a program and application code microcontroller that controls and monitors processes [43]. Web apps' continuous growth has led to a growing demand for resources and information through the Internet. We should presume that users can reliably and efficiently access much of their everyday information from the Internet [44]. All these resources are website-based and

server-driven. Web servers then accept requests from the website, process, and have the answers.Further, users' dependence on the web-based public and private sectors on various electronic fields puts tremendous pressure on servers [45]. In addition, the reliability and effectiveness of web servers decide how companies and web developers draw their customers with accurate and fast responses [46]. The output of web servers, however, which are instantly influenced by the load. Therefore, it is essential to monitor the load on the web servers [47]. Therefore, researchers need to design and propose an effective device that can withstand a heavy load.

Moreover, the vast and ongoing increase in customer requirements for server resources is the leading cause of the overload. This additional burden causes servers to crash and languish precisely in providing services. Therefore, overloading server harm hurts the company's consumer appeal, reduces revenues, and loses credibility [48].

2. TYPES OF OPERATING SYSTEMS

A model of an OS is a large structure that incorporates the many services and Characteristics offered by the operating system and the tasks it performs [1]. Operating systems are mainly classified into the following categories, based on their mechanism structure:

2.1 Windows OS

This OS was introduced into the market within 1985, and as a comprehensive and robust kind of software, almost 90% market share above and over other OS. Its perceived and great effectiveness as home computers in building commercials, manufacturing plants, and its conspicuous presence. At the same time, this point is considered not to be so again due to the overwhelming interest of people in open-source operating systems [49]. The Microsoft OS as an M-S windows family was produced with its origin from the MS-DOS command line as a graphical layer over that of old MS dos, and this is maintained until date with the DOS Box command prompt that is cmd.exe [50].

2.2 Android OS

Android Inc. is the original developer of this platform. Google eventually acquired it and launched the OS as AOSP. The OHA (Open Handset Alliance) creation, a consortium responsible for distributing and creating Android, complemented this new development. The software that is now published within the Apache license is marked, including a free, open-source license. As a result of the comprehensive developer communities available that frequently update and build applications using custom versions of Java, Android releases a new version [51].

2.3 iPhone OS

An Apple Inc. develops and operates iOS which is a smartphone OS. It was built and designed for the iPhone, but it expanded to support the Apple TV and iPad [34]. Like some other OS, iOS is regularly modified from iOS version 4.0, and iOS version 5.1 is the newest [7]. At the bottom of the iPhone operating system architecture, the Main OS layer resides. An extra Pre-occupation layer, media, cocoa-touch layer, and the core services layer of the iOS architecture is included. Including the scheduler, file system, Mach kernel, memory system management, and hardware drivers, network and protection framework and inter process communication, the OS core layer includes the planner to protect system and program data [51, 52].

2.4 Macintosh OS

It is much older than Windows OS. A year ago, its Microsoft equivalent was released, and it is the first-ever popular operating system that is graphical-inclined, among other OSs. One of Mac Interface's Guidelines' key ideas is that everything should remain where it is held. Apps for the Mac operating system are not called massive monoliths. The device's graphic user interface (GUI) supports the program instructions or commands partially implemented in a hardware-conveyed ROM and partly implemented in distributed libraries, via a reasonably stable event interface, quickly communicates with Mac OS software programs [53].

2.5 Linux OS

Linux is a free OS, and that can be downloaded, updated, and even redistributed at no cost. In the domain of the operating system, Linux is relatively new. In the year 1991, it was written and also updated for current use. It is possible to compare Linux and Windows to an object whose roof and floor are either replaceable or not. However, with Linux, as a unit, both roof and floor can be transferred in whatever way one wishes. However, the roof and floor of the windows are hugely rigid that it stays secure. Developers within the open-source Linux community want to gain a significant share of end-user desktops that increase the number of intended Linux users than the Unix of old school users who are afraid to share the desire at OS market and server [54].

3. KERNEL ISSUES

In the beginning, the computer was without multitasking, and darkness was upon the face of the multiuser. The first computers were single task, single-user 'bare metal' machines. The executing processes had complete control over the memory, the hardware, and the teletype user interface. The requirement for multiuser and multitasking necessitated the invention of the (OS). An OS is a set of programs that manage a computer's resources and provide services efficiently to all application processes. At the heart of an OS is the kernel. It is the first process loaded at boot time, and it remains in continuous use for the duration of the session [55]. In its simplest form, a kernel provides the following services:

(1). A 'scheduler' to allocate CPU (Central Processor Unit) resources: (2). A 'memory manager'. (3). IPC (Inter-Process Communication) control. In addition, it is may also provide. (4). Physical device drivers (PDD) abstraction layers for peripheral devices (hardware drivers), and (5). Logical device drivers (LDD) abstraction layers for disk filing systems called a Virtual File System (VFS), Protocol Stacks such as Sockets (TCP/IP stack) [56].

All types of OS are being used for all computer machines, including laptops, desktops, supercomputers, tablets, handhelds, and even video game consoles. In today's ICT world, Apple Inc. designed various operating systems, Linux by Group, Windows by Microsoft Inc., and Android by Google Inc. and others [57].

3.1 Concepts of Kernels

There are various types of the kernel: Micro-kernel, Monolithic-kernel, and Hybrid Kernel: -

3.1.1 Microkernels

Microkernels have minimal 'built-in' functions, scheduling, memory management, and IPC. All other OS features are allocated to hardware userland drivers but controlled by the kernel. The advantage of the microkernel is that the application drivers and window managers are in the user space and can be coded and quickly substituted in languages not used by the kernel without changing the kernel. A very high IPC overhead has disadvantages. Examples of microkernels are 'Mach' and 'MINIX' [58].

3.1.2 Monolithic-kernel

It has all the microkernel functionality plus hardware drivers, virtual file system (VFS), and protocol stacks. This reduces inter-process communication requirements and enhances system security but requires careful system design and reduces flexibility available to the designer. UNIX and Linux are examples of monolithic kernels, as illustrated in fig. 2. [59].

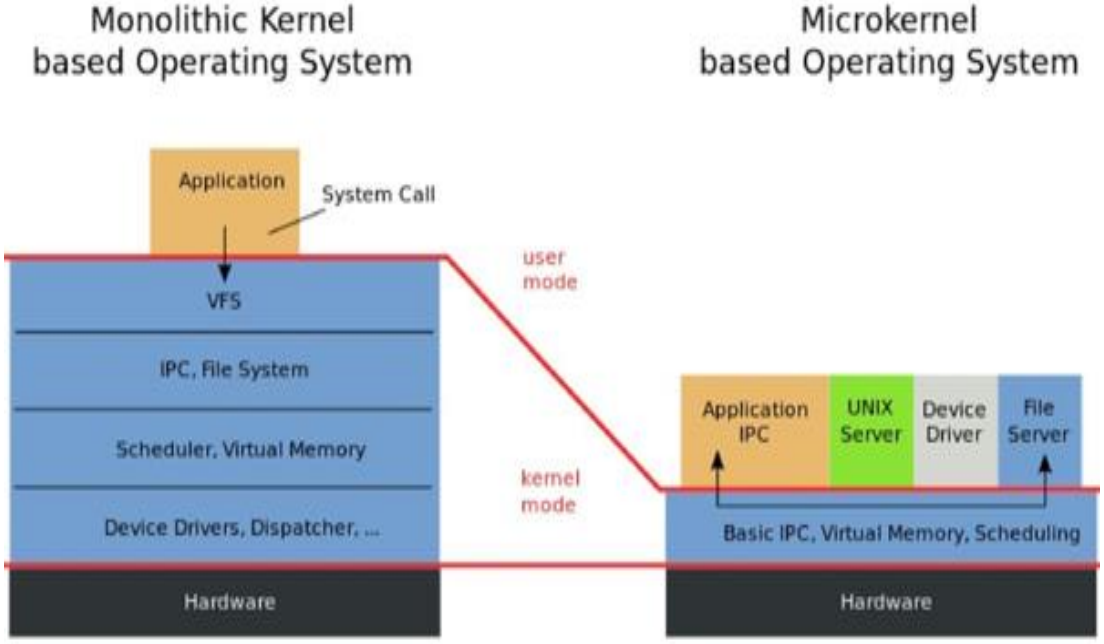

Fig. 2. The architecture of a Transform of Monolithic Kernel to Microkernel [59].

3.1.3 Hybrid-kernels

It attempts to get the best of both worlds by using a mix of micro-kernel and 2/3 monolithic kernel techniques; for example, windows have a few basic drivers built into the kernel additional drivers in userland. Unfortunately, some hybrid-kernel designs seem to incorporate the worst of both worlds. Windows and Mac claim their kernels are hybrid [60].

3.2 Kernel Issues

A comparative analysis of the following OS should be carried out: Windows, Linux, iOS, and Android. Concern Issues are architectures that support computers, supported file system, type of target computer, integrated firewall, lay-user friendly, shell terminal, security threat, type of kernel, compatibility, and stability. In addition, the benefits and drawbacks of each of the operating systems have been identified [61].

4. LITERATURE REVIEW

Different studies on OS used for desktop computer systems and handheld devices have been performed. In this section, a summary of existing works is provided.

Tian et al. [62] Addressed that the OS kernel is the operating system's central component and plays a vital role in management of OS resources. The kernel rootkit is a common way to compromise OS kernel (i.e., malicious kernel module). When rootkits are loaded into kernel space, they can perform arbitrary high-privileged malicious operations. In recent years, they concentrated on defeating kernel rootkits by offering several approaches. However, some restrictions apply to modern methods: 1) most approaches concentrate on rootkit-detection in user-mode; 2) some tools are only used to find obscure kernel modules;

To fix these issues, VKRD has been proposed, a hardware-assisted virtualization-based kernel rootkit identification method. VKRD will provide a straightforward and effective execution environment for the target kernel module to demonstrate its operating time behavior in comparison with previous methods. They used the TF-IDF approach in order to pick the important run-time functions to train their detection models. Their kernel rootkit identification solutions have been used theoretically in the cloud world by incorporating the hardware-assisted virtualization and machine learning techniques. The studies have shown that their machine can detect root kits for the windows with very high precision and low efficiency.

Kim et al. [63] proposed a Linux-kernel 4.9.108 solution beyond the platform of NVIDIA Jetson AGX Xavier. Separation of Performance was a highly desirable aspect of the runtime that undertakes the system because the Xavier series SoC is uniquely designed for embedded deep learning (based) performance-hogging applications. They suggested a memory-aware equal share scheduling algorithm that could reduce QoS applications' sensitivity to other co-operation applications' memory-related interference. Its algorithm has wisely isolated the real stall from the CPU cycles of a running task and compensates for the task that is memory dependent so that the task gets the desired CPU share until it's too long. In that it does not rely on a static allocation or division of memory hardware resources, and boosts the efficiency of QoS applications with just negligence of runtime overhead, the proposed solution was adapative, effective and efficient. In addition, it was a workaround for software only, that could easily be integrated with a few modifications to the kernel schedule. You have applied the algorithm in the Linux CFS and named the end product mCFS.

Srinuan et al. [64] COMEX, an OS kernel extension, suggested cooperative memory expansion. COMEX builds a secure memory pool across nodes in a cluster and improves the OS' memory subsystem by enabling the Process Page Table to monitor remote memory page frames with no programming effort or changes to application codes. For low laatency transmitted data with a destination kernel, the COMEX employed Remote Direct Memory Access (RDMA), which was not based on the old I/O block subsystem architecture, which is normally used by all documented remote paging. COMEX is well suited to the evolving approach of systems architecture of a disintegration of resources that divided hard walls into a new design model for different resource pools between server-centered devices. The modern architecture has allowed the scaling and scaling-out of the infrastructure and removed imbalances in data centres. They deployed and Implemented COMEX on a networked test-bed platform consisting of 32 Dell servers. Also, performance evaluation findings for ten applications within two benchmark suites. The study indicated that COMEX experiences execution at a higher speed when the ratio of the footprint size of the device execution to the local memory size increases, with speed rising to 170 when the ratio is equal to 10.

Duca et al. [65] Presented real-time open-sourcing driver implementation for Texas Instruments OMAP4 multicanal SPI peripheral on EVL, PREEMPT RT and Xenomai real-time Linux extensions. Furthermore, the device was equipped with an FPGA-based interrupt and SPI latency evaluation system which tests the drivers in real time. One million test samples were generated for each SPI real-time driver in the testing and the implementation was tested in real time. For both experiments, both latencies and the transition time of SPI are limited to available values. Which means that all four SPI implementations have real time behaviour. The results are acceptable. PREEMPT RT has the lowest power output of all four implementations, but satisfies real-time restrictions, and has considered that the non-real-time regular Linux OMAP4 mcSPI driver with stress tested latencies of several milliseconds.

By PREEMPT RT-EVL, the PREEMPT RT output has been enhanced. Then EVL came, and Xenomai achieved the best result.

Boggavarapu et al. [66] Proposed Dual-Dedup, a lightweight scheme that brought lower layer block deduplication to the knowledge of page cache management. The redundancy information sensed by the block-level deduplication layer was revealed to the cache page, thus removing cache redundancy and preventing unwanted read requests. On the Linux EXT4 they developed a device prototype. Results of experiments showed that double-degrees would increase reading efficiency dramatically. As an example, Dual-Dedup enhanced read performance 34 percent by using FIO benchmarks for a data collection of 25 percent duplicate data.

Liu et al. [67] Submitted a principle evidence for the new technique for ESD-induced soft failure identification by evaluating the operating system kernel feature trace records. The tool was used to monitor Linux function calls during regular service and after the injection of ESD tension. In order to illustrate changes caused by ESD, the records were displayed in graphical feature maps and machine call distribution for each operation. The experimental results showed that the improvements in the feature maps and the call distribution within the observed procedures showed that the soft failures. The new approach was able to detect slight system disruptions that could not be seen by regular I/O or connected device for the consumer.

Sun et al. [68] Examined the relationships in the kernel of Android OS through a dynamic network modeling of the operating system. Each node in their network was a function and connections were different call relationships between them. Three distinct relationships between topological statistics and population size have been identified with community research. They also did a percolation study and identified basic mechanisms in software networks in order to locate organizational vulnerabilities on a different scale. Its results may help to clarify the complexities of the system and to devise appropriate methods of software testing.

Iqbal et al. [69] Google has built a simple Linux kernel designed to operate on touch-screen platform such as tablets and smartphones with the Android mobile operating system. As a result of poor OS protections, numerous security attacks have made it vulnerable to restricting the access of third-party applications from sensitive infrastructure. They proposed the optimal permission elimination solution. Also demonstrated a way to avoid exploiting application authorization by holding availability on shared User of ID set applications. Proposed Security is derived through the tool as a solution and how to prevent the breach of user information to ensure Security—different types of Android protection threats and different permission types for Android applications.

Wang et al. [70] Addressing Android kernel activity extensively, a kernel based CrowdNet architecture for cloud computing platforms was first introduced. CrowdNet. CrowdNet also included an automated data provider which collects kernel footprints and a parallel malware forecast that validates malicious activity on Android. Then, using a heuristic approach to 12,750 Android apps, they measured and picked hidden centers to minimize iteration and the time complexity. Their experimental findings indicate that CrowdNet protected large-scale data validation and doubled kernel learning. Increased classification performance by detecting malicious attacks with CrowdNet, comparison with conventional neural networks and other machine learning techniques.

Kouki et al. [71] The topic of mutual data monitoring under the imperfect of wireless networking has been examined in a class of multi-agent linear systems. A prediction control program for the autonomous Internet of Things was developed to drive a single-wheel robot. A new approach was developed using the STM 32-running Revolutionary Internet of Things Operating System and RFCs through the User Datagram Protocol. Due to the large number of packet errors in communication channels, for the performance assessment of the predictive control algorithm the User Datagram Protocol was used. A robust study of agent Internet of Things technologies and a network predictive packet failure management technique, restricted bandwidth and attachment connections have been undertaken. It was mostly that the management and stability of closed loop control systems could be achieved by consensus. Several experimental scenarios showed the reliability of the proposed concept solution.

Saleel et al. [72] studied the vulnerability in the Linux kernel called dirty copy on write. They indicated that the mentioned vulnerability is a serious issue, and unauthenticated users could complete controlling the devices operated by Linux OS. The authors have shown that the attacker could use the existing vulnerability and perform numerous different attacks. Also, they indicated that this issue could be addressed by using antivirus. Moreover, they revealed that all Linux distributions (Red Hat. Debian. Ubuntu. SUSE) had fixed this problem by updating the OS. Further, deep defense, up-to-date OS, and proper firewall should be used to tackle this issue on Linux OS.

Cha et al. [73] presented an analysis study related to the kernel of the Linux operating system on multicore systems that can process packets. Moreover, they identified the major performance bottlenecks in the Linux kernel network stack's packet processing direction and provided insight into various performance and scalability issues that must be addressed. The proposed study focused on the software-defined network functions running in the kernel space due to these user-space application-related overheads. They indicated that the performance of the Linux kernel could be enhanced by preventing program stack contention. In addition, the results have shown that beyond a particular stage, the output does not scale uniformly as the number of cores increases. NUMA locality, process synchronization overheads, interrupt management, buffer bloating, direct memory access overhead, as well as using CPU cycles and NIC capacity are all possible considerations.

Jung et al. [74] demonstrated how to port AODV-UU from an older kernel to a newer one. By introducing threads and direct access to the routing table. Moreover, they eliminated the need for a particular kernel by eliminating the kernel dependency. In a simple example, the authors test the operation of the implemented AODV-UU. In more details, instead of using libipq, they used a direct queue. In that architecture, the routing table was operated directly by a new AODV thread, while in previous versions, it was controlled by the kaodv module.

5. SURVEY DISCUSSION AND ANALYSIS

This review presents five operating system types (Windows, Mac, Linux, iOS, and Android) operating systems based on the operating system characteristics with their weaknesses and strengths. The research paper aims to show core concepts of the primary desktop and handheld OS and preview the comparisons between them and include and review user interface toolkits from a technological developers' standpoint. The similarity of operating systems based on issues and characteristics is illustrated in Table I.

As shown in Table 1, it concerns the issues and merits of the various operating system used in different types of computer/mobile devices. Most kernels that were widely studied in the five previous years are Linux and Android kernels, respectively. Most of the studies

performed on the central issue are the security concerns such as malware, attack intrusion detection, and prevention, etc. Moreover, the researchers used numerous tools and frameworks to carry out their studies. The target of their research was to improve the performance of the different kernels in several fields.

Table 1. Critical Analysis of Existing Studies

Ref.	Kernel Type	OS Type	Tools	Issues	Merits
[62]	Kernel Module.	Windows OS.	Kernel rootkit detection virtualization method based on machine learning and the TF-IDF technique.	Allow hackers the ability to circumvent, disable antivirus software, and monitor your keyword keys to make it easy to steal your private information from criminals.	With both extreme accuracy and moderate cost of performance, the device can detect Windows kernel rootkits.
[63]	XNU Linux kernel patch.	Embedded OS.	Five SPEC CPU2017 simulation programs were used, as well as the YOLO detection of face software.	Many H\W are based on solutions, inefficient reference management, or unsuccessful throttling execution, making it impossible to use a COTS and SoC platform is widely distributed operating systems such as Linux.	Adaptive, efficient, and responsive since it does not focus on any static partitioning or allocation of H\W memory references and enhances QoS network performance.
[64]	COMEX on Linux kernel 3.10.87.	Windows and Linux OS.	COMEX has been implemented on the Linux kernel and installed on its 32 networked servers.	To allow fine-grain kernel-level memory aggregation in computed network systems to support the design of reference segmentation and swapping of page defects through the virtual memory subsystem of the Linux OS.	Enabling software with memory footprints better than what is required for the physical memory of a node alone, COMEX uses remote node physical memory in a networked distributed system.
[65]	Linux kernel	Real-time OS open source	Real-time open-source drivers for the Texas real-time Linux versions of	The author proposed real (time) actions in the relating drivers of Linux for the Xilinx OMAP4 more than one channel SPI chip and implemented and	PREEMPT RT-EVL develops the performance of PREEMPT RT. Then arrives Xenomai and EVL obtain the correct

			EVL, PREEMPT RT, and Xenomai Instruments OMAP4 multi-channel SPI peripheral.	designed an assessment method.	results.
[66]	Linux Kernel.	Real-time OS.	Implement the Dual-De-dup module in the Virtual File System (VFS) layer.	Big data performance with eliminating unwanted disk reads in the page cache on duplicate data.	In terms of both read performance and throughput, it illustrates significant new features.
[67]	Linux Kernel function call.	GNU/Linux OS.	It is using kernel call-trace recordings using a high-speed router operating on Linux. A test port that outputs the call (trace) data has this duty.	Temporary upsets-soft faults can be caused by electrostatic discharge (ESD) into a working device. Subtle soft failures can reduce system stability and cannot be defined by the traditional operating system (OS) log-based methods.	This novel method can detect subtle device upsets that are not detectable by normal I/O or attached devices to the user.
[68]	Android OS Kernel.	Android OS.	Analyze network reliability when the networks are interrupted or attacked based on the process of the methodology developed.	It remains an open question how the feature components communicate with each other to recognize the essential dynamic behavior of the OS.	These results can help to understand the functionality of the system and design effective methods of software development.
[69]	Linux Kernel	iOS and Android mobile OS.	Security implement as a solution and how to prevent this violation of user information to optimize Security.	Issues and solutions with Android OS security.	Demonstrates to stop fraud of device permissions by holding a lock on personal user ID set software.

[70]	Android Kernel.	Android OS.	The technique of Crowd-Net, DT, SVM, LR, NB, NN, CNN.	Malicious attacks, kernel functionality, and Android intrusion prevention on Android apps.	When the size of data significantly increases and reduces the processing time caused by repeated I/O communications, the Crowd-Net technique enhances the classification efficiency.
[71]	Monolithic and Microkernel Approaches.	Embedded OS.	Context-aware framework for IoT controller design.	The consistency issues of the random or fixed data transmission NCSs are analyzed.	The structure of this scheme is willing on a solid performance, meager cost, best communication in the middle of electronic devices, and low peripheral throughput, which is not the same in prior methods.
[72]	Linux Kernel	Several Linux distributions		The COW issue existed on all Linux distributions and could be used for performing different types of attacks.	The discussed problem could be addressed by updating the OS and using the appropriate antivirus and firewall.
[73]	Linux Kernel	Linux kernel 4.4.1.	40Gbps network, multilayer virtual switch called OVS.	A variety of overheads associated with core kernel mechanisms such as task scheduling, memory management, process synchronization, interrupt handling	identify the significant performance bottlenecks in the Linux kernel network stack's packet processing direction that should be addressed
[74]	Linux Kernel	Several Linux kernels	AUDV-UU and Raspberry Pi 3	Since the latest AODV-UU source code was written for the old Linux kernel 2.6 and then extended for Linux kernel 3.8, it is incompatible with new kernel versions.	We are porting AODV-UU from an older kernel to a newer one.

6. CONCLUSION

In particular, windows and the handheld operating system appear to be the most frequently used variants of the novelist. The reason is that they are cost-effective, reliable, compatible,

secure, and user-friendly. It could have been inferred that each OS was established in a particular direction by considering security problems and their merits. Every operating system gives its suppliers specific and competitive services and features. Additionally, all open-sourced OS enjoys implementing the new techniques in software updates and applications every day by different developers from the community. This also improved their performance and security features, while the company operating system lacks design versatility. This will not stress the fact that every OS should be desired or accepted, but users' preference depends on the services needed. The majority of studies focus on improving the performance of the OS kernel in a variety of areas, including attack, malware, and intrusion prevention and detection.

COMPETING INTERESTS DISCLAIMER:

Authors have declared that no competing interests exist. The products used for this research are commonly and predominantly use products in our area of research and country. There is absolutely no conflict of interest between the authors and producers of the products because we do not intend to use these products as an avenue for any litigation but for the advancement of knowledge. Also, the research was not funded by the producing company rather it was funded by personal efforts of the authors.

REFERENCES

[1] W. Stallings and G. K. Paul, *Operating systems: internals and design principles* vol. 9: Pearson New York, 2012.

[2] D. Hwang, M. Yang, S. Jeon, Y. Lee, D. Kwon, and Y. Paek, "Riskim: Toward complete kernel protection with hardware support," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019, pp. 740-745.

[3] Z. S. Ageed, S. R. Zeebaree, M. M. Sadeeq, S. F. Kak, Z. N. Rashid, A. A. Salih*, et al.*, "A Survey of Data Mining Implementation in Smart City Applications," *Qubahan Academic Journal,* vol. 1, pp. 91-99, 2021.

[4] H. I. Dino, S. R. Zeebaree, O. M. Ahmad, H. M. Shukur, R. R. Zebari, and L. M. Haji, "Impact of Load Sharing on Performance of Distributed Systems Computations," *International Journal of Multidisciplinary Research and Publications (IJMRAP),* vol. 3, pp. 30-37, 2020.

[5] L. Haji, R. Zebari, S. Zeebaree, W. Abduallah, H. Shukur, and O. Ahmed, "GPUs Impact on Parallel Shared Memory Systems Performance," *Int. J. Psychosoc. Rehabil,* vol. 24, pp. 8030-8038.

[6] H. Jiang, W. Gao, M. Wang, S. See, Y. Yang, W. Liu*, et al.*, "Research of an architecture of operating system kernel based on modularity concept," *Mathematical and computer modelling,* vol. 51, pp. 1421-1427, 2010.

[7] B. T. Jijo, S. R. Zeebaree, R. R. Zebari, M. A. Sadeeq, A. B. Sallow, S. Mohsin*, et al.*, "A Comprehensive Survey of 5G mm-Wave Technology Design Challenges," *Asian Journal of Research in Computer Science,* pp. 1-20, 2021.

[8] H. I. Dino, S. R. Zeebaree, A. A. Salih, R. R. Zebari, Z. S. Ageed, H. M. Shukur*, et al.*, "Impact of Process Execution and Physical Memory-Spaces on OS Performance."

[9] N. J. Zaidenberg and E. Khen, "Detecting kernel vulnerabilities during the development phase," in *2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing*, 2015, pp. 224-230.

[10] A. Fernandes and J. Tribolet, "Enterprise Operating System: the enterprise (self) governing system," *Procedia Computer Science,* vol. 164, pp. 149-158, 2019.

[11] S. Zeebaree, B. Salim, R. Zebari, H. Shukur, A. Abdulraheem, A. Abdulla*, et al.*, "Enterprise Resource Planning Systems and Challenges," *Technol. Rep. Kansai Univ,* vol. 62, pp. 1885-1894, 2020.

[12] K. H. Sharif, S. R. Zeebaree, L. M. Haji, and R. R. Zebari, "Performance Measurement of Processes and Threads Controlling, Tracking and Monitoring Based on Shared-Memory Parallel Processing Approach," in *2020 3rd International Conference on Engineering Technology and its Applications (IICETA)*, 2020, pp. 62-67.

[13] A. A. Salih, S. R. Zeebaree, A. S. Abdulraheem, R. R. Zebari, M. A. Sadeeq, and O. M. Ahmed, "Evolution of Mobile Wireless Communication to 5G Revolution," *Technology Reports of Kansai University,* vol. 62, pp. 2139-2151, 2020.

[14] E. Baccelli, C. Gündoğan, O. Hahm, P. Kietzmann, M. S. Lenders, H. Petersen*, et al.*, "RIOT: An open source operating system for low-end embedded devices in the IoT," *IEEE Internet of Things Journal,* vol. 5, pp. 4428-4440, 2018.

[15] S. R. Zebari and A. S. Yowakib, "Improved Approach for Unbalanced Load-Division Operations Implementation on Hybrid Parallel Processing Systems," *Science Journal of University of Zakho,* vol. 1, pp. 832-848, 2013.

[16] Z. N. Rashid, K. H. Sharif, and S. Zeebaree, "Client/Servers Clustering Effects on CPU Execution-Time, CPU Usage and CPU Idle Depending on Activities of Parallel-Processing-Technique Operations," *Int. J. Sci. Technol. Res,* vol. 7, pp. 106-111, 2018.

[17] Z. S. Ageed, S. R. Zeebaree, M. M. Sadeeq, S. F. Kak, H. S. Yahia, M. R. Mahmood*, et al.*, "Comprehensive survey of big data mining approaches in cloud systems," *Qubahan Academic Journal,* vol. 1, pp. 29-38, 2021.

[18] A. A. Yazdeen, S. R. Zeebaree, M. M. Sadeeq, S. F. Kak, O. M. Ahmed, and R. R. Zebari, "FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review," *Qubahan Academic Journal,* vol. 1, pp. 8-16, 2021.

[19] M. Hähnel, W. Cui, and M. Peinado, "High-resolution side channels for untrusted operating systems," in *2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17)*, 2017, pp. 299-312.

[20] L. M. Abdulrahman, S. R. Zeebaree, S. F. Kak, M. A. Sadeeq, A.-Z. Adel, B. W. Salim*, et al.*, "A state of art for smart gateways issues and modification," *Asian Journal of Research in Computer Science,* pp. 1-13, 2021.

[21] I. M. Ibrahim, "Task scheduling algorithms in cloud computing: A review," *Turkish Journal of Computer and Mathematics Education (TURCOMAT),* vol. 12, pp. 1041-1053, 2021.

[22] O. Alzakholi, H. Shukur, R. Zebari, S. Abas, and M. Sadeeq, "Comparison among cloud technologies and cloud performance," *Journal of Applied Science and Technology Trends,* vol. 1, pp. 40-47, 2020.

[23] D. H. Maulud, S. R. Zeebaree, K. Jacksi, M. A. M. Sadeeq, and K. H. Sharif, "State of art for semantic analysis of natural language processing," *Qubahan Academic Journal,* vol. 1, pp. 21-28, 2021.

[24] D. A. Hasan, B. K. Hussan, S. R. Zeebaree, D. M. Ahmed, O. S. Kareem, and M. A. Sadeeq, "The impact of test case generation methods on the software performance: A review," *International Journal of Science and Business,* vol. 5, pp. 33-44, 2021.

[25] S. Zeebaree and K. Jacksi, "Effects of Processes Forcing on CPU and Total Execution-Time Using Multiprocessor Shared Memory System," *Int. J. Comput. Eng. Res. Trends,* vol. 2, pp. 275-279, 2015.

[26] H. Shukur, S. R. Zeebaree, A. J. Ahmed, R. R. Zebari, O. Ahmed, B. S. A. Tahir*, et al.*, "A State of Art Survey for Concurrent Computation and Clustering of Parallel Computing for Distributed Systems," *Journal of Applied Science and Technology Trends,* vol. 1, pp. 148-154, 2020.

[27] H. M. Musse and L. A. Alamro, "Cloud Computing: Architecture and Operating System," in *2016 Global Summit on Computer & Information Technology (GSCIT)*, 2016, pp. 3-8.

[28] Z. S. Ageed, R. K. Ibrahim, and M. A. Sadeeq, "Unified Ontology Implementation of Cloud Computing for Distributed Systems," *Current Journal of Applied Science and Technology,* pp. 82-97, 2020.

[29] A. M. Abdulazeez, S. R. Zeebaree, and M. A. Sadeeq, "Design and Implementation of Electronic Student Affairs System," *Academic Journal of Nawroz University,* vol. 7, pp. 66-73, 2018.

[30] S. Zeebaree, S. Ameen, and M. Sadeeq, "Social media networks security threats, risks and recommendation: A case study in the kurdistan region," *International Journal of Innovation, Creativity and Change,* vol. 13, pp. 349-365, 2020.

[31] M. A. Sulaiman, M. Sadeeq, A. S. Abdulraheem, and A. I. Abdulla, "Analyzation Study for Gamification Examination Fields," *Technol. Rep. Kansai Univ,* vol. 62, pp. 2319-2328, 2020.

[32] Z. Tahir, M. Aslam, and M. Fatima, "CLOUD COMPUTING INFLUENCE ON OPERATING SYSTEM," *Science International,* vol. 27, 2015.

[33] Z. Ageed, M. R. Mahmood, M. Sadeeq, M. B. Abdulrazzaq, and H. Dino, "Cloud computing resources impacts on heavy-load parallel processing approaches," *IOSR Journal of Computer Engineering (IOSR-JCE),* vol. 22, pp. 30-41, 2020.

[34] H. R. Abdulqadir, S. R. Zeebaree, H. M. Shukur, M. M. Sadeeq, B. W. Salim, A. A. Salih*, et al.*, "A Study of Moving from Cloud Computing to Fog Computing," *Qubahan Academic Journal,* vol. 1, pp. 60-70, 2021.

[35] M. M. Sadeeq, N. M. Abdulkareem, S. R. Zeebaree, D. M. Ahmed, A. S. Sami, and R. R. Zebari, "IoT and Cloud computing issues, challenges and opportunities: A review," *Qubahan Academic Journal,* vol. 1, pp. 1-7, 2021.

[36] P. Gaur and M. P. Tahiliani, "Operating systems for IoT devices: A critical survey," in *Proceedings of the 2015 IEEE Region 10 Symposium*, 2015, pp. 33-36.

[37] A. I. Abdulla, A. S. Abdulraheem, A. A. Salih, M. A. Sadeeq, A. J. Ahmed, B. M. Ferzor*, et al.*, "Internet of Things and Smart Home Security," *Technol. Rep. Kansai Univ,* vol. 62, pp. 2465-2476, 2020.

[38] A. S. Abdulraheem, A. A. Salih, A. I. Abdulla, M. A. Sadeeq, N. O. Salim, H. Abdullah*, et al.*, "Home automation system based on IoT," 2020.
[39] M. A. Sadeeq and S. Zeebaree, "Energy management for internet of things via distributed systems," *Journal of Applied Science and Technology Trends,* vol. 2, pp. 59-71, 2021.
[40] S. M. S. A. Abdullah, S. Y. A. Ameen, M. A. Sadeeq, and S. Zeebaree, "Multimodal emotion recognition using deep learning," *Journal of Applied Science and Technology Trends,* vol. 2, pp. 52-58, 2021.
[41] Z. S. Ageed, S. R. Zeebaree, M. A. Sadeeq, M. B. Abdulrazzaq, B. W. Salim, A. A. Salih*, et al.*, "A State of Art Survey for Intelligent Energy Monitoring Systems," *Asian Journal of Research in Computer Science,* pp. 46-61, 2021.
[42] O. H. Jader, S. Zeebaree, and R. R. Zebari, "A State Of Art Survey For Web Server Performance Measurement And Load Balancing Mechanisms," *International Journal of Scientific & Technology Research,* vol. 8, pp. 535-543, 2019.
[43] J. Riihijarvi, P. Mahonen, M. J. Saaranen, J. Roivainen, and J.-P. Soininen, "Providing network connectivity for small appliances: a functionally minimized embedded web server," *IEEE Communications Magazine,* vol. 39, pp. 74-79, 2001.
[44] Z. A. Younis, A. M. Abdulazeez, S. R. Zeebaree, R. R. Zebari, and D. Q. Zeebaree, "Mobile Ad Hoc Network in Disaster Area Network Scenario: A Review on Routing Protocols," *International Journal of Online & Biomedical Engineering,* vol. 17, 2021.
[45] R. R. Zebari, S. R. Zeebaree, and K. Jacksi, "Impact Analysis of HTTP and SYN Flood DDoS Attacks on Apache 2 and IIS 10.0 Web Servers," in *2018 International Conference on Advanced Science and Engineering (ICOASE)*, 2018, pp. 156-161.
[46] B. S. Osanaiye, A. R. Ahmad, S. A. Mostafa, M. A. Mohammed, H. Mahdin, R. Subhi*, et al.*, "Network Data Analyser and Support Vector Machine for Network Intrusion Detection of Attack Type."
[47] M. Sadeeq, A. I. Abdulla, A. S. Abdulraheem, and Z. S. Ageed, "Impact of Electronic Commerce on Enterprise Business," *Technol. Rep. Kansai Univ,* vol. 62, pp. 2365-2378, 2020.
[48] S. R. Zeebaree, K. Jacksi, and R. R. Zebari, "Impact analysis of SYN flood DDoS attack on HAProxy and NLB cluster-based web servers," *Indones. J. Electr. Eng. Comput. Sci,* vol. 19, pp. 510-517, 2020.
[49] F. Q. Kareem, S. R. Zeebaree, H. I. Dino, M. A. Sadeeq, Z. N. Rashid, D. A. Hasan*, et al.*, "A Survey of Optical Fiber Communications: Challenges and Processing Time Influences," *Asian Journal of Research in Computer Science,* pp. 48-58, 2021.
[50] W.-C. Fan, C.-S. Wong, W.-K. Lee, and S.-O. Hwang, "Comparison of Interactivity Performance of Linux CFS and Windows 10 CPU Schedulers," in *2020 International Conference on Green and Human Information Technology (ICGHIT)*, 2020, pp. 31-34.
[51] R. Győrödi, D. Zmaranda, V. G. Adrian, and C. Győrödi, "A comparative study between applications developed for Android and iOS," *International Journal of Advanced Computer Science and Applications,* vol. 8, pp. 176-182, 2017.

[52] A. B. Sallow, M. Sadeeq, R. R. Zebari, M. B. Abdulrazzaq, M. R. Mahmood, H. M. Shukur*, et al.*, "An Investigation for Mobile Malware Behavioral and Detection Techniques Based on Android Platform," *IOSR Journal of Computer Engineering (IOSR-JCE),* vol. 22, pp. 14-20.

[53] B. Charles, N. C. Rowe, and M. R. McCarrin, "Memory forensics and the macintosh os x operating system," in *Digital Forensics and Cyber Crime: 9th International Conference, ICDF2C 2017, Prague, Czech Republic, October 9-11, 2017, Proceedings*, 2018, p. 175.

[54] O. C. Novac, M. Novac, C. Gordan, T. Berczes, and G. Bujdosó, "Comparative study of Google Android, Apple iOS and Microsoft Windows phone mobile operating systems," in *2017 14th International Conference on Engineering of Modern Electric Systems (EMES)*, 2017, pp. 154-159.

[55] N. Harki, A. Ahmed, and L. Haji, "CPU Scheduling Techniques: A Review on Novel Approaches Strategy and Performance Assessment," *Journal of Applied Science and Technology Trends,* vol. 1, pp. 48-55, 2020.

[56] K. Bala, S. Sharma, and G. Kaur, "A study on smartphone based operating system," *International Journal of Computer Applications,* vol. 121, 2015.

[57] M. T. Ahvanooey, Q. Li, M. Rabbani, and A. R. Rajput, "A survey on smartphones security: software vulnerabilities, malware, and attacks," *arXiv preprint arXiv:2001.09406,* 2020.

[58] S. Dong, Y. Xiong, W. Huang, and L. Ma, "KIMS: Kernel Integrity Measuring System based on TrustZone," in *2020 6th International Conference on Big Data Computing and Communications (BIGCOM)*, 2020, pp. 103-107.

[59] T. F. Bissyandé, "Contributions for improving debugging of kernel-level services in a monolithic operating system," Université Sciences et Technologies-Bordeaux I, 2013.

[60] J. Ding, X. Zhu, J. Guo, X. Li, and R. Yan, "End-to-End Automated Verification for OS Kernels," in *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, 2018, pp. 139-148.

[61] H. Nair and R. Sridaran, "An Innovative Model (HS) to Enhance the Security in Windows Operating System-A Case Study," in *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2019, pp. 1207-1211.

[62] D. Tian, R. Ma, X. Jia, and C. Hu, "A Kernel Rootkit Detection Approach Based on Virtualization and Machine Learning," *IEEE Access,* vol. 7, pp. 91657-91666, 2019.

[63] J. Kim, P. Shin, M. Kim, and S. Hong, "Memory-Aware Fair-Share Scheduling for Improved Performance Isolation in the Linux Kernel," *IEEE Access,* vol. 8, pp. 98874-98886, 2020.

[64] P. Srinuan, X. Yuan, and N.-F. Tzeng, "Cooperative memory expansion via OS kernel support for networked computing systems," *IEEE Transactions on Parallel and Distributed Systems,* vol. 31, pp. 2650-2667, 2020.

[65] L.-C. Duca, A. Duca, and A.-S. Lup, "Real-time Linux drivers and latency evaluation system for TI OMAP4 mcSPI peripheral," in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, 2020, pp. 1-4.

[66] R. K. Boggavarapu and S. Jiang, "Deduplication-aware I/O Buffer Management in the Linux Kernel for Improved I/O Performance and Memory

Utilization," in *2020 12th International Conference on Knowledge and Smart Technology (KST)*, pp. 70-74.

[67] X. Liu, G. Maghlakelidze, J. Zhou, O. H. Izadi, L. Shen, M. Pommerenke*, et al.*, "Detection of ESD-induced soft failures by analyzing linux kernel function calls," *IEEE Transactions on Device and Materials Reliability,* vol. 20, pp. 128-135, 2020.

[68] P. Sun, S. Yang, Z. Lai, D. Li, and A. Yao, "Function-call network reliability of kernel in android operating system," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019, pp. 1-5.

[69] S. Iqbal, A. Yasin, and T. Naqash, "Android (Nougats) security issues and solutions," in *2018 IEEE International Conference on Applied System Invention (ICASI)*, 2018, pp. 1152-1155.

[70] X. Wang, C. Li, and D. Song, "CrowdNet: Identifying Large-Scale Malicious Attacks Over Android Kernel Structures," *IEEE Access,* vol. 8, pp. 15823-15837, 2020.

[71] R. Kouki, A. Boe, T. Vantroys, and F. Bouani, "Autonomous Internet of Things predictive control application based on wireless networked multi-agent topology and embedded operating system," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering,* vol. 234, pp. 577-595, 2020.

[72] A. Saleel, M. Nazeer, and B. D. Beheshti, "Linux kernel OS local root exploit," in *2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2017, pp. 1-5.

[73] S.-J. Cha, S. H. Jeon, Y. J. Jeong, J. M. Kim, and S. Jung, "Analysis of Linux Kernel Packet Processing on Manycore Systems," in *TENCON 2018-2018 IEEE Region 10 Conference*, 2018, pp. 2276-2280.

[74] S. Jung, B.-S. Kim, K.-I. Kim, B. Roh, and J.-H. Ham, "Implementation of AODV-UU on Linux 4.15 Kernel," in *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems Workshops (MASSW)*, 2019, pp. 160-161.