# An Analysis on Host Vulnerability Evaluation of Modern Operating Systems

Afifa Sajid
Department of Computer Science
COMSATS Institute of Information Technology,
Islamabad, Pakistan

Munam Ali Shah
Department of Computer Science,
COMSATS Institute of Information Technology,
Islamabad, Pakistan

Muhammad Kamran
Department of Distance Continuing & Computer Education
University of Sindh, Hyderabad,
Pakistan

Qaisar Javaid
Department of Computer Science & Software Engineering,
International Islamic University, Islamabad, Pakistan

Sijing Zhang
Department of Computer Science & Technology
University of Bedfordshire, Luton, UK

***Abstract*—Security is a major concern in all computing environments. One way to achieve security is to deploy a secure operating system (OS). A trusted OS can actually secure all the resources and can resist the vulnerabilities and attacks effectively. In this paper, our contribution is twofold. Firstly, we critically analyze the host vulnerabilities in modern desktop OSs. We group existing approaches and provide an easy and concise view of different security models adapted by most widely used OSs. The comparison of several OSs regarding structure, architecture, mode of working, and security models also form part of the paper. Secondly, we use the current usage statistics for Windows, Linux, and MAC OSs and predict their future. Our forecast will help the designers, developers and users of the different OSs to prepare for the upcoming years accordingly.**

***Keywords*—Security; Operating system; Virtualization; kernel; Reliability; Vulnerability evaluation***

## I. Introduction

In today's world, information security is very important. It can relate to our national policy, civilization, economy, and military affairs, etc. Hence, for the information security system, operating system (OS) security is very important [1]. The OS is the core intermediary that resides on the hardware and controls the security of a computing environment. Any security vulnerability that exists in the OS causes the vulnerability at a host, therefore, the OS security is considered as the most critical part of a computing environment. Any OS that gives the reliability and supports and addresses the security issues efficiently to meet a certain set of requirements is called a secure or trusted operating system [1]. An OS manages the working and operation of all the complex applications on a computer system. It must also be capable of coping up with a largely increasing number of dangerous malicious attacks, software bugs, and hardware failures.

Through the Internet, all the web browsers accept download and run executable files. Desktop OSs and many other applications are compatible with the plug-in technology making the host environment more vulnerable to attacks. Furthermore, there is a growing trend that more users pay attention and respond to different executable codes that are present on the Internet. If these executable codes are directed from unauthorized sources or infected by viruses, then execution of such files can bring the system security at high risks [2]. Tragically, most of the desktop personal computer (PC) operating systems only give basic protection. Before running the executable files they do not authorize properly [3]; it is one of the many reasons for the accelerating spread of harmful software. Nowadays, everyone uses smartphones, desktop computers or laptops in their routine tasks. In this case, these systems are handling different environments like web browsing. At the same time, these systems are also handling data which is private or sensitive and is used for the confidential corporate matters. There exists the risk of malicious attacks on the Internet. These attacks can modify the confidential corporate data. Hence, the security of end user and/or security of the whole corporation is at risk. If this problem is neglected, it may cause severe security flaws. When a virus attacks any host or file, it propagates from one host to the other and replicates itself. To stop the propagation of virus from one host machine to the other host isolation is needed.

The existing research does not provide critical analysis and comparison of different desktop OSs and the tools that can make an OS more secure. This lag in existing literature motivated us to survey the security aspects of modern OSs. In this paper, a comparison of different OSs regarding their kernel security, network security, and system security is provided. The rest of the paper is organized as follow. Section II presents the security mechanism for different OSs. We also analyze different OSs for different parameters such as kernel-based containment and virtualization. The modules and techniques to secure an OS have been discussed in Section III. In Section IV, the security performance comparison of different OSs has been provided. The open issues are discussed in Section V where we also predict the future for different operating systems. The paper is concluded in Section VI.

## II. Operating System Security Mechanism

Security is a challenging task in computing systems. Operating system is a component that manages all the other applications, so it needs to be secured [3]. For analyzing the

characteristics of an operating system, there is no general set of metrics available that can access the possible risks present in any operating system.

### A. Security Methods

The methods used for operating system security comprises of software security and hardware security [4]. Hardware security method consists of I/O protection, running protection and storage protection while software security method relates mainly to the following features.

*a) Identification and authentication:* User identification and authentication is mandatory. Identification is that system needs the user's identity. And the process of connecting user identifier with the user is known as authentication.

*b) Access Control:* Computer system security's primary mechanism is access control. It consists of three steps. First is authorization second is access permission and the third is impose access permission.

*c) Least Privilege:* Give all users, only the kind of privileges that they need to complete the task.

*d) Trusted Channel:* Usually in computers an interaction between the Operating system and a user is through middle application layer which is not trustworthy. So the operating system needs to make sure that during communication Trojan horse cannot be able to capture the information.

*e) Virus protection:* In real world protecting your computer system from viruses is a very difficult task. In general, certain functions can be protected using security MAC (Mandatory Access Control) mechanism of operating system security.

### B. Quantifying Risks

Unfortunately it is not easy to quantify the risks related to operating system [5]. This fact is reflected by the lack of research in this area. The problem lies in recognizing such data. If we can easily identify such data only then we can take the right actions accordingly. For both personal purposes and professional purposes Linux, Windows, Mac, and Solaris are considered the best operating system and billions of users use them. Before selecting any of them, there is a need to answer the question that which of them performs the required functions in the best way. In this section, we will analyze all these operating systems and certain security modules.

### C. Windows Operating System

Windows are the most popular operating system. The desktop environment is dominated by Windows operating system. It is portable and extensible operating system, hence, it can take benefit of new hardware and new techniques. Windows allow multiprocessing (symmetric) and several operating environments [6]. It supports Non-uniform Memory Acce8uss (NUMA) computers, 32-bit and 64-bit processors. To offer basic services, Windows use the kernel objects with client server computing to give support to lots of application environments. It provided preemptive scheduling, virtual memory and integrated caching. The statistical data obtained from [7] has been used to plot the OS usage for different operating systems in 2015. It can clearly be observed that Windows is the most widely used OS.

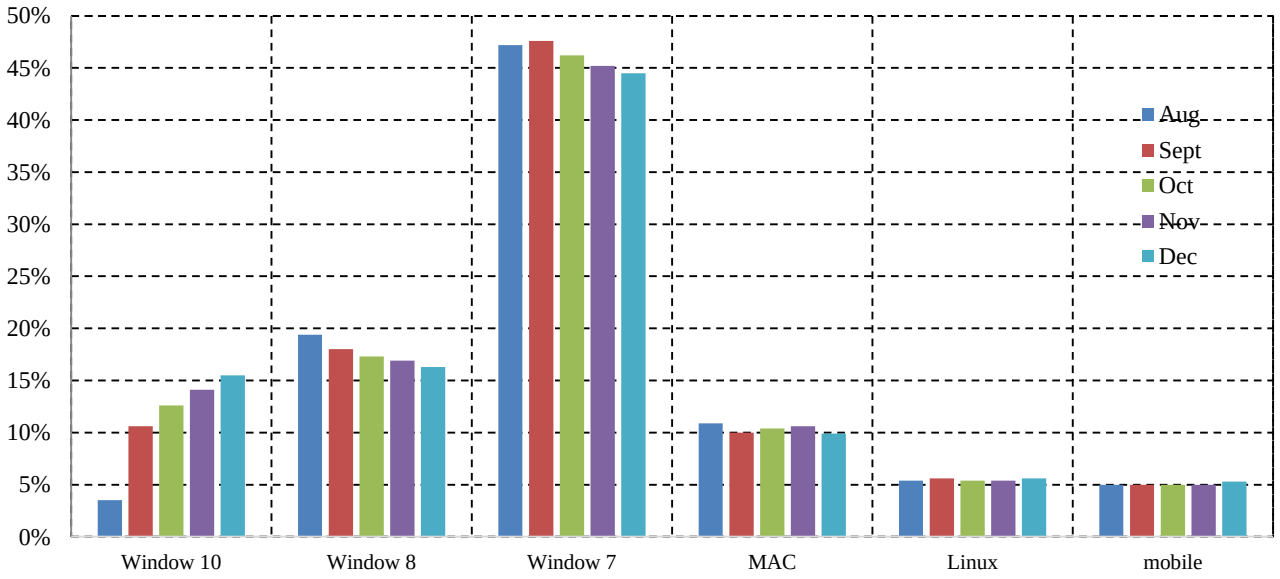


Fig. 1. Graph representing OS usage in 2015

By design, Window operating system is monolithic. In Windows operating system design, much focus is given to GUI (Graphical User Interface). In Windows operating system end points are unprotected and susceptible to malware. It is not considered efficient in managing obscure changes and software updates. In addition to updates, it gives poor end user experience. If we consider administration, actions performed by an administrator can be easily isolated from the user actions.

*a) Windows OS Kernel:* Kernel system for both Windows and Linux is very secure. Although the design decisions of Windows are different than Linux, but to exploit either of operating system by kernel is very rare. Windows kernel provides virtualization [8]. The virtualization process seperates one virtual machine from the other that's why it is considered good for security [9]. Certain security models and techniques are presented in literature for Windows operating system kernel to make it more secure.

- *Kyongi Information Security System:* This mechanism shows fast response against any attack. An effective response mechanism is built by analyzing kernel attack types [10]. It gathers information and analyzes the Kernel information and then use this information against windows kernel by using response mechanism. Kernel level programming skills are required to implement this system.
- *Symbolic Execution:* This system was proposed for testing paths to achieve high-security coverage [11]. The program is executed with symbolic inputs and do not use the concrete inputs. Whenever a branch instruction is executed it preserves the path condition. This path condition is updated with every execution of the program. Test generation is performed by solving the collected constraints with the help of constraint solver. It checks for the assertion violation or run-time errors, and it creates test inputs that trigger those errors. It checks all the supported functions. There still exists scalability issues because of the analysis of a number of large paths and constraints complexity that is produced.

*b) Windows OS System Security:* Security subsystem in Windows is made up of certain components that cannot grant access to a user without proper authentication and identification. Only security subsystem function offers the access control [12]. This can be implemented by giving different privileges and rights to the user. Another similar feature known as capabilities, it is present in Linux system [6]. There is a need to find out the possible risks associated with privileges, rights and capabilities. By doing this we can find the risks and bugs more efficiently.

*c) Windows OS Network Security:* The network security is achieved by data monitoring using socket. In this technique data is monitored using socket [13]. At any stage of the process when it realizes the need, it will monitor the net data, then apply certain development on net base data to allow secure transmission.

*d) Linux Operating System*

Linux is built on UNIX principles and is free, open source operating system. The user interface and programming interface is compatible with the regular UNIX system. A kernel of Linux operating system is fully original, but it allows many UNIX-based applications to run. Because of the performance reasons its kernel is implemented as a monolithic kernel, but at run time, drivers can be loaded and unloaded dramatically [14]. Linux design is modular enough, and Linux servers are considered best for non-local administration. It supports a multi-user environment. Using time-sharing scheduler it gives protection among several running processes. It allows multithread programming. In Linux to reduce the repetition of data shared by many processes, memory management uses copy-on-writing and page sharing. Because of the open source theory of Linux it allows more peer evaluation of the code to find bugs and to fix the code [15]. But it does not mean that Linux is secure. In past Windows had been the most attacked operating system. Because of this reason most Windows-based servers were shifted to Linux. Ultimately, the attackers started targeting Linux accordingly.

*e) Linux Kernel Security:* Kernel is the most sophisticated and complex piece of software on the system. It is the core of the system [16]. If the kernel or any of its parts is affected by virus or any malicious code then the system will be corrupted and all secret information can be stolen, and files can be deleted. The Linux kernel provides virtualization.

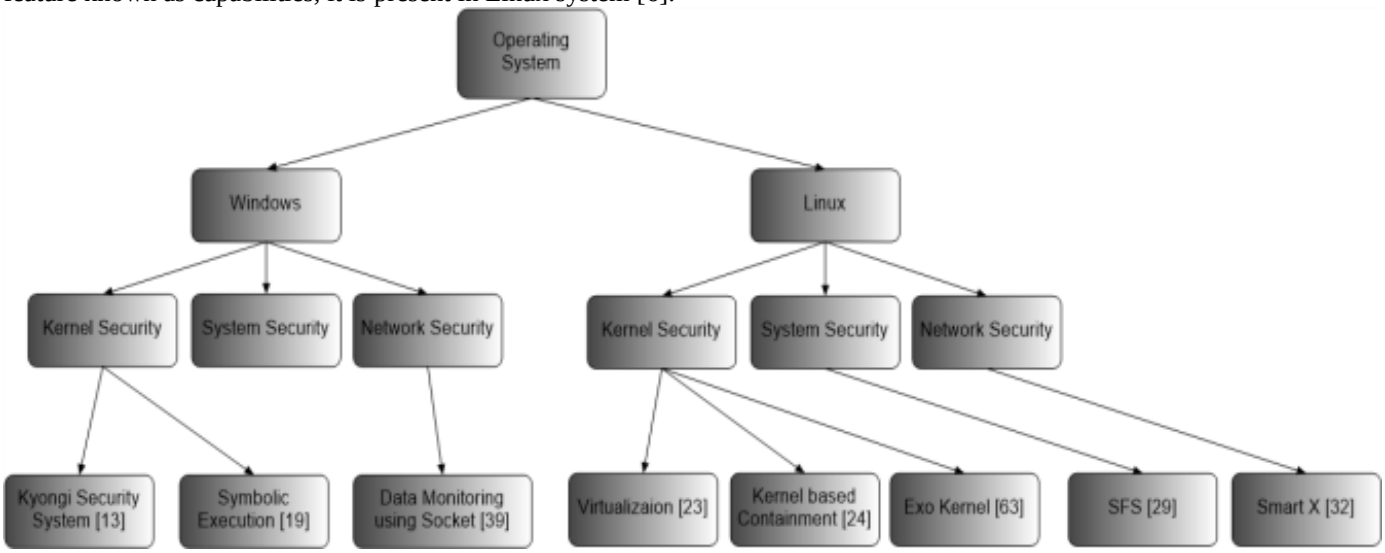


Fig. 2. Taxonomy Diagram of Linux, Windows, and their security modules

- *Virtualization:* For many server-oriented systems an approach has been taken to diminish the propagation of viruses or any other malicious attacks by separating the various services on different machines [17]. By using the technique of virtualization, we can achieve the isolation by running several virtual machines (VMs) on one physical machine, an application gets corrupted and run under a given VMs cannot be able to interrelate with different other applications in separate VMs.
- *Kernel Based Containment:* The isolation of different activities from one another. can be done by using the access rights like HRU [18]. Each user then can access only the subset of files, but in the case of corruption of any application, the attacker can get a hand on every file of the user and can also change the permission of these files [18]. Mandatory Access Control (MAC) systems such as TOMOYO Linux [19] and SELinux [20] allows limiting the damage caused by the compromised application. Due to this security system the administration can set access control policy for every user so the attacker gets fewer privileges that could be gained by the application diversion phenomenon [16]. The disadvantage in MAC solution is that it has very high cost for the administrative purposes because the policies used are very complex and large.
- *Virtualization Based Containment:* There is a virtualization technology called Xen through which different domains can be placed in various virtual machines [21]. As compared to modern kernel code, virtualization code is simpler [10], thus, it needs more security. The project proposes simplified mandatory access control kernel.
- *Exokernel:* It is an extension of the microkernel theory [22]. Abstraction of the physical devices is not provided by the base kernel, putting down that to the applications that required devices. It separates the security from abstraction, hence making parts of the operating system which are non-override-able to do nothing but to multiplex the hardware. The objective is to avoid forcing any abstraction on applications and let the abstraction use whatever abstraction best suited for their task. Exokernel technology is still much not researched comprehensively.

*f) File Data Security in Linux:* In Linux and Unix files are thought to be one of the important unit for storing information. The idea of a file is of much significance that all the devices that are input/output are believed to be files [23]. So in Linux problem of protecting data is considered to be protecting file data. In current computer environments file data security is mandatory. Utilities like "aescrypt" and "crypt" are used for securing file data. In these utilities as input password and file name is given to produce a file which will then encrypted [12]. Linux file system is a hierarchical tree that follows the Unix. To manage different file systems, Linux makes use of an abstraction layer [24]. Virtual file system, device oriented and networked these all are supported. Different modules are used to enhance the file security system some of these are

- *Secure File System (SFS):* To secure the file data, SFS uses cryptographic techniques. Encryption of SFS services is done in kernel space of Linux [23]. After integration, it provides security services as it is inherited [20]. For storing file data, SFS ask the user to make a directory named "ecrypt" like ecryptdir. Without user intervention, all of the files placed in this directory will be encrypted transparently.

*g) Linux Network Security* In Linux, certain access permissions are defined for network security. First is password security, to access the Linux system users have to go through an authentication process in this process all users needs to input the password and account then system validates the user and grant access. In Linux, each user is given different level of permissions to access any data. Some important users are given more access than the others who are given less access to the data [25]. To increase the security of the system Linux removes unnecessary services. In a network Linux limits the number of IP (Internet Protocol) addresses. A super user password is used which is generally known to the system administrator and needs to be revised periodically [26]. Also, user of Linux are given a lower level account, not the root account so if the system is compromised only the certain programs, and local files will be affected by that vulnerability.

- *Smart-X:* This software framework can be used in Windows and Linux for efficient and secure communication connecting two nodes. The working of this module involves creating a tunnel between two end points using a single context switching and a single copy of data. An algorithm is used to secure the two end points. For encryption/decryption purposes advanced standard encryption 128-bit algorithm is used [27]. This software framework remains on Network Driver Interface Specification (NDIS). Through this framework, some modifications are performed on a packet before transmission. The framework encrypts the whole packet from the start of the header to the end of the header and then generates one or more UDP packets from it. If the encrypted packet size exceeds then, it is divided into two UDP packets. These UDP packets hit the wire and send out to the destination, where they are reassembled if necessary and then decrypted.

## III. Modules and Techniques for OS Security

There are some more modules and techniques available which can be used to enhance the operating system security. Some of these are:

*a) Chinese Wall Security Policy Model:* Based on the realistic commercial business model this security policy was devised by Nash and Brewer [28]. In this security policy, the company information is divided into three storage levels. Single data elements make the base level; company data elements form the next level and the level on top of all this consists of the information with regards to conflict of

interest(COI) [29]. One company can belong to only one COI. A user may visit any CD (Control Domain) within a COI without any limiting or enforcement factor [28]. But after the user makes any decision, then user cannot get access to other elements of the same COI, it is like a wall has been created around the CD.

TABLE I. MODULES AND TECHNIQUES FOR MAKING OS SECURE

| *Ref.* | *Name* | *Operating System* | *Security level* | *Features* |
|---|---|---|---|---|
| [21] | Symbolic Execution | Windows | Kernel Security | It checks all the supported functions to provide high-security coverage. |
| [20] | Kyongi Information Security System | Windows | Kernel Security | An effective response mechanism is built by analyzing kernel attack types. |
| [26] | Smart-X | Windows/ Linux/ UNIX | Network Security | Working of the module involves creating a tunnel between two endpoints using single context switching and single copy of the data. |
| [10] | Socket Data Monitoring | Windows | Network Security | Data is monitored using a socket and this technology is known as cutting age data monitoring. |
| [28] | Secure file system | Linux/ UNIX | System security/file data security | To secure the file data SFS uses cryptographic techniques. |
| [30] | Signature based code scanning | Windows/ Linux/ UNIX | N/A | scanning is performed to find any malicious code in it if identification of any malicious code is confirmed then execution of program file is denied |
| [13] | Chinese wall security policy | Windows /Linux/ UNIX | N/A | Information is divided into different levels for making it secure. |

*b) Signature Based Code Scanning*: A mostly used and well-known technique that is signature based malicious code scanning is used for the purpose of authorization and inspection of executable codes [30]. In operating system, any program file scanning is performed to find any malicious code in it. If the identification of any malicious code is confirmed then execution of program file is denied. The problem with this code scanning is, it can only defend against the attacks which are known. Solution to this is an approach known as combined integrity measurement and access control. Only authentic programs can run in this model [17]. If there is an existing code which is from an authentic source and not considered malicious. It shows malevolent manners and at run time can be trapped and manipulated.

*c) Security Enhanced Linux:* There was a project under NSA (National Security Agency) called security enhanced Linux. This project is to implement MAC on Linux for military purposes. For the architecture La Padula model [31] is followed and supports RBAC (Rule Based Access Control). In file system EA (Extended Attribute) is used to label the files. The security context is also supported for any known application. During execution of an application, security context can be switched by the application. It is a Linux modification and was initially released by NSA in January 2000 [32]. It extends the features of Linux with certain security capabilities to make it more secure. SE (Security Enhanced) Linux gives a language for all the security policies of Linux. Deletion: Delete the author and affiliation lines for the second affiliation. This specified security policy covers all the aspects of the system like file management, network communication and process control [29]. Policy enforcement uses the method in Flask Architecture. Here a server gives the decision, for policies whether user request should be granted to the operating system or not. For making a decision this security server refers to the other internal policies.

## IV. OTHER OPERATING SYSTEM

In this section, we provide the performance comparison of some other OSs. Our focus is on security of Solaris and Macintosh OS.

### A. Solaris Operating System

Solaris is a Unix-based operating system [33]. In Solaris multiple software isolated applications can run on a single system, that's how linking between servers become easy. Some abilities of Solaris 10 like Process Rights Management, Predictive Self-Healing and Dynamic Tracing helps in attaining good utilization without causing any harm to privacy or security levels [33]. Solaris operating system offers adequate CPU and memory to applications and also preserve the ability to utilize the idle resources [34]. It has an ability to recover automatically from disastrous problems that occur in the system by using both Solaris containers and self-healing functionalities.

Solaris 10 offers advanced security features [32]. Solaris containers work with process rights and user management to give secure hosting of hundreds of applications and several customers on the system. To apply secure foundation, security administrators can harden and minimize Solaris. It offers the following functionalities to the users.

TABLE II. COMPARATIVE ANALYSIS OF DIFFERENT OSS

| Features | *Linux/Unix* | *Windows* | *Solaris* | *MAC* |
|---|---|---|---|---|
| What is it? | Open source development and free operating system [51]. | Operating system from Microsoft, not free [52]. | Unix-based operating system introduced in 1992 [32]. | Series of graphical user interface based operating system [40]. |
| Manufacturers | Developed by a society Linus Torvalds manage things [20]. | The developer of Windows is Microsoft. | Developed by Sun Microsystems. Now be in possession of Oracle Corporation [32]. | Developed by Apple Inc in 1984 [40]. |
| Cost | Can be downloaded free of charge. Some priced editions are also offered [53]. | For desktop users, it can be costly depending on the version. From $50 to $450 [13]. | Costly ranging from $500 to $720 [54]. | Mac OS is too expensive [54]. |
| Users | Anyone from home users to developers. | Anyone from home users to developers. | Not for home users, for developers. | Anyone from home users to developers. |
| GUI | Two default GUI, Gnome and KDE others are like twm, Unity, Mate, LXDE, Xfce are also offered [19]. | GUI is an important part of operating system and is not replaceable [55]. | Provides Gnome and CDE [34]. | N/A |
| File system support | NTFS, FAT, FAT32, Xfs, Btrfs, Ext2, Ext3, Ext4, Jfs, ReiserFS [15] | NTFS, FAT, exFAT, FAT32 [56]. | ZFS, UFS, HSFS, NFS, TFS, PCFS [57] | HFS or HFS+ , Macintosh file system [45]. |
| Text Mode Interface | BASH (Bourne against SHell) is default shell, can maintain several command interpreters [58]. | A command shell and each version of Windows uses single command interpreter with a command like DOS have, there is an addition of non-compulsory PowerShell [59]. | Solaris console and kernel terminal emulator [46]. | N/A |
| Security | Till date 100-200 viruses programmed not keenly spreading [60]. | There exist lots and lots of viruses; antivirus costs $20 to $450 [38]. | Fully virus protected. Assured and tested low-risk platform [60]. | Much more secure than Windows, viruses designed for Windows processors won't run on Mac [61]. |
| Threat detection and solution | In Linux case, threat detection and solution is very quick, as Linux is mostly society driven and at any time when any Linux user posts any type of threat, some developers start working on it from different parts of the world [16]. | After detecting threat, Microsoft releases a patch which fixes the problem and takes more than 2/3 month sometimes sooner. Updates and patch are weekly based [13]. | N/A | Apple provides software updates, works with the incident response community such as CERT, FIRST, and FreeBSD Security Team, to proactively identify and quickly correct operating system vulnerabilities [45]. |
| Processors | Dozens of different kinds. | Limited. | Limited. | Different kind. |
| Gaming | Very little number of games are offered natively. It can be used to play some, but frequently not all features are offered [62]. | Almost all games are compatible with Windows. Some CPU intensive and graphics intensive games are exclusive to Windows PC's [63]. | It supports few games. | Games are available for the Mac, gives environment much like Windows but not better than it [64]. |
| Version | Fedora, Red Hat, Debian, Android, Arch Linux, Ubuntu, etc. | Vista, XP, Window 7, 8, 8.1 etc. | Solaris 4.1x, 5.1, 5.2-5.11 | Mac OS x 10.1, 10.8.1, 10.8 .3 etc. |
| User experience | Although there are many GUI applications, most of the work is done through Terminal (a console window), and if a problem occurs GUI is not often usable to fix them [65]. | Everything can be controlled through GUI and incompatibility problems are exceptional [66]. | Not user-friendly not recommended for home users. | Average not more customizable and user-friendly. |
| Graphics Performance | Because hardware manufacturers, such as NVidia, regularly does not provide documentation for Linux developers, drivers can' not use full card performance [38]. | Integrated with newest DirectX versions and full graphics card support the performance is approximately as good as it can get [ 31]. | Graphics performance is poor, lack of good GUI [40]. | N/A |
| Supported Platforms | All [38] | PowerPC: versions 1.0 - NT 4.0, DEC Alpha: versions 1.0 - NT 4.0, MIPS R4000: versions 1.0 - NT 4.0, IA-32: versions 1.0 – 8, IA-64: version XP, x86-64: versions XP – 8, ARM: version RT [38] | SPARC, IA-32, PowerPC and i86PC (which includes both x86 and x86-64) platforms [67]. | Compatible with only power pc processors version 10 and version 10.3, 68k processors. |
| Preceded by | Basic terminal (CLI) [38] | MS-DOS[31]. | N/A | N/A |

| Features | *Linux/Unix* | *Windows* | *Solaris* | *MAC* |
|---|---|---|---|---|
| Terminal | Multiterminal windows | With Solaris/X86 2.4 not configured with Solaris 2.4-7 configured [36]. | N/A | N/A |

*a)* With the use of file verification and Solaris secure execution, it validates the system integrity.

*b)* Grant access only to the privileges needed processes and users, hence reduce the risk level.

*c)* For File encryption Solaris uses open standard-based cryptographic framework and hence makes the administration easier.

*d)* For network traffic security Solaris uses IP-Filter firewall.

Solaris is not for the home users; it is considered good for developers. Solaris lacks GUI (Graphical User Interface).

### *B. Macintosh Operating System*

For home users Apple Macintosh was a debatable first GUI operating system. At first, it was designed for Apple computers, then slowly it was by Microsoft Windows. Mac (Macintosh) operating system has several benefits, apart from all these benefits two main serviceable benefits are protected memory and pre-emptive multitasking [35]. The scheduler is a layer which is a point of an interaction between microprocessor and application. This concept of the scheduler was introduced by Mac. When user opens an application the scheduler takes its control and allocates certain amount of CPU (Central Processing Unit) memory to the process. When the allocated time of the process is up, then scheduler takes control back and gives the CPU memory to another application [28]. Memory activity needs to be securely controlled, so for this when an application is executed, checks are given to confirm that memory activity of application is within bound [36]. If an application tries to interfere with critical resources or tries to write in another application space, it would be infeasible. This concept is called protected memory. Because of these two features, any rogue process or application can't make the whole system hostage [37]. Scheduler supports dynamic feedback and application with time constraints. Table 1 given below summarizes different security modules/techniques and their features and the Table 2 represents the comparative analysis of different operating systems.

## V. Discussion and Open Issues

To support developers and the end users in their daily tasks, the modern OSs continue to evolve. Users do not need to have underlying architectural details of the resources. With the growing scope of resource type and growing demand for functionality, the OS has thus grown to become a flexible and large platform for which change by now is not convenient. For example, in Windows 7, to remove a dispatcher lock, one needs to write 6000 lines of code. For such complex system, execution cost is compensated by stronger and faster processor generation. With the advent of multi-core systems, the above assumption no longer holds. To provide powerful services, even though the underlying hardware will not increase exponentially anymore, the system needs to exploit the infrastructure and its specifications.

It will take effort and time into adapting OSs and programs for particular high-performance computing usage. Desktop OSs must be able to cope with any infrastructure and must demonstrate high portability. The actual objective which is behind the Microsoft Windows OS development is to hide the peculiarities of the underlying infrastructure which is complex and instead offer ease of portability of applications. The next generation operating systems must value these characteristics for usability and growth. It is obvious that Microsoft Windows has been the most popular and widely used OS until 2015. We reuse the statistics provided and Figure 1 and derive a relation to predicting the future growth of Windows OS up to the year 2020. We estimate that the Windows OS will continue to grow in its adaptation and usage by the end users. However, taking into account the worst scenario, the Microsoft Windows OS usage may decrease. This means that ideally, there will be 70% of the entire users who will be using Windows OS by the year 2020. Even, if the end users stop using the Windows OS, the total usage will still be 50% or more. The growth trends for Microsoft Windows OS are provided in Figure 3. We also predict that the Microsoft will be naming its OS as *Windows'20* in the year 2020. This would be because of the trends and the rapid growth in the concept of *Internet of Things* (IoT).

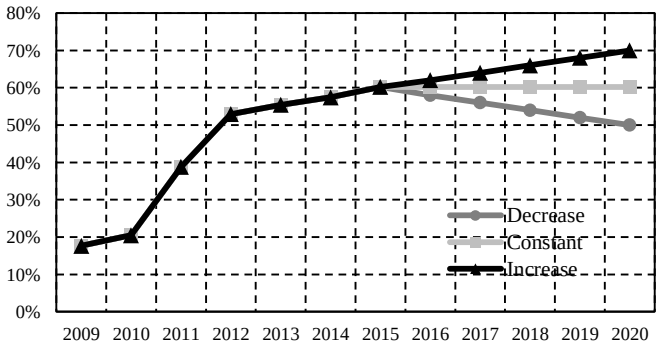


Fig. 3. Statistics representing the growth trends of Windows operating system in the future

Based on the current statistics for Linux OS, we predict that the growth in the usage of Linux OS will rise to approximately more than 7%. The future prediction of Linux usage is provided in Figure 4. One obvious reason for this inference is that Windows security model is not resilient to threats and vulnerabilities, and the end user would require some strong, secure OS, which can provide sufficient security to the confidential data. However, it is also possible that the current Linux based users migrate to Windows OS and the Linux usage may drop to 4% or lesser.

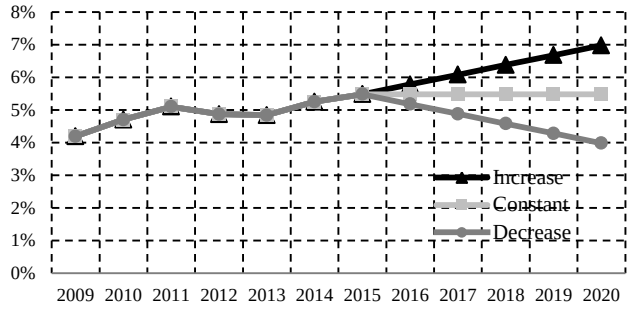


Fig. 4. Statistics representing the growth trends of Linux operating system in the future

We then predict the future usage of the Macintosh OS in the year 2020. We strongly believe that the MAC OS will continue to evolve, and the end user will adapt MAC OS more frequently in near future. However, it should be noted that comparatively, the MAC based PCs are still costly. The vendor must focus on reducing the cost of the hardware to make MAC based PCs affordable to the general public. The growth trend of MAC OS is given in Figure 5.

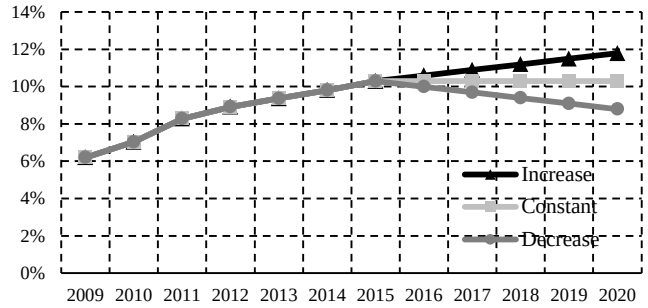


Fig. 5. Statistics representing the growth trends of Macintosh operating system in the future

## VI. Conclusion

Achieving security is a challenging task. The operating system is the one which coordinates amongst all the other systems. This is the reason; the security, is of primary and major concern. In this survey paper, we analyzed several different features such as network security, system security and kernel security etc. of multiple OSs including Windows, Linux, and MAC. Some of the salient features of different OSs have been summarized in Table 3. We believe that the choice of an OS solely depends on the user requirement. A home user with suitable performance in multi-tasking selects Windows OS. When it comes to handling a large amount of data and to secure the resources, Solaris is a better choice. For multiuser environment, Linux is best. If the graphics and GUIs are concerned, MAC OS is a leading. There are certain modules and techniques available which can be adapted to make an OS more secure. Furthermore, it is believed that absolute security is impossible to achieve, and no OS is 100% secure against all types if threats and vulnerabilities. However, the designers and developers of the OS can strive for maximizing the security in all possible aspects and must satisfy the end users' security needs.

TABLE III. Advantages And Disadvantages Of Different Operating Systems

| Operating System | Advantages | Disadvantages |
|---|---|---|
| Linux/ UNIX | • Provides multiuser environment.<br>• Free of cost.<br>• Modular design. | • Some Window programs may not run.<br>• Learning curve for new people.<br>• Not user friendly like Windows. |
| Windows | • Good GUI.<br>• Universal plug and play feature available.<br>• User friendly as compared to other OS. | • Easily compromised by virus, loopholes are there.<br>• Heavy system old hardware is not able to run it.<br>• Single user license. |
| Solaris | • No reboot needed, can run 24/7.<br>• Fully virus protected.<br>• Good backup tools. | • Not user friendly.<br>• Not good GUI.<br>• Not for home users. |
| MAC | • Much more secure than Windows.<br>• Can get bootcamp.<br>• Not popular as Windows, so not a target. | • Too expensive.<br>• Some programs that run on Window won't run on Mac.<br>• Few games available. |

## References

[1] M. Bishop, Computer security: art and science, vol. 200. Addison-Wesley, 2012.

[2] N. L. darek and O. Jae, "A system dynamics model for information security management," Information & mnagement , vol. 52, no.1, pp. 123-134, 2015.

[3] M. Maekawa, K. Shimizu, X. Jia, P. Sinha, K. S. Park, H. Ashihara, and N. Utsunomiya, Operating System. Springer Science & Business Media, 2012.

[4] J. Song, G. Hu, and Q. Xu, "Operating System Security and Host Vulnerability evaluation," in Management and Service Science, 2009. MASS'09. International Conference on, 2009, pp. 1–4.

[5] J. K. Guo, S. Johnson, and I.-P. Park, "An operating system security method for integrity and privacy protection in consumer electronics," in Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, 2006, vol. 1, pp. 610–614.

[6] A. Silberschatz, P. B. Galvin, G. Gagne, and A. Silberschatz, Operating system concepts, vol. 9. Addison-Wesley Reading, 2008.

[7] w3school, "OS Statistics," http://www.w3schools.com/browsers/browsers_os.asp. .

[8] D. H. Lee, J. M. Kim, K.-H. Choi, and K. J. Kim, "The study of response model & mechanism against windows kernel compromises," in Convergence and Hybrid Information Technology, 2008. ICHIT'08. International Conference on, 2008, pp. 600–608.

[9] T. Ni, Z. Yin, Q. Wei, and Q. Wang, "High-Coverage Security Testing for Windows Kernel Drivers," in Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on, 2012, pp. 905–908.

[10] J. Park, J. Park, J. Lee, B. Kim, G. Lee, and B. Cho, "Windows Security Patch Auto-Management System Based on XML," in Advanced Communication Technology, The 9th International Conference on, 2007, vol. 1, pp. 407–411.

[11] C. Cadar, P. Godefroid, S. Khurshid, C. S. P\uas\uareanu, K. Sen, N. Tillmann, and W. Visser, "Symbolic execution for software testing in practice: preliminary assessment," in Proceedings of the 33rd International Conference on Software Engineering, 2011, pp. 1066–1071.

[12] E. B. Fernandez and T. Sorgente, "A pattern language for secure operating system architectures," in Proceedings of the 5th Latin American Conference on Pattern Languages of Programs, 2005, pp. 16–19.

[13] M. Howard and S. Lipner, "Inside the windows security push," IEEE Secur. Priv., vol. 1, no. 1, pp. 57–61, 2003.

[14] E. William, H. Nadkarni, A. Sadeghi, A. Reza and H. Stephan, "Programmable interface for extending security of application-based Operating System," US Patent 20,160,042,191. 2016.

[15] R. K. Pal and I. Sengupta, "Enhancing file data security in linux operating system by integrating secure file system," in Computational Intelligence in Cyber Security, 2009. CICS'09. IEEE Symposium on, 2009, pp. 45–52.

[16] G. Zhai and Y. Li, "Analysis and Study of Security Mechanisms inside Linux Kernel," in Security Technology, 2008. SECTECH'08. International Conference on, 2008, pp. 58–61.

[17] C. Border, "The development and deployment of a multi-user, remote access virtualization system for networking, security, and system administration classes," in ACM SIGCSE Bulletin, 2007, vol. 39, no. 1, pp. 576–580.

[18] T. Harada, T. Horie, and K. Tanaka, "Task oriented management obviates your onus on Linux," in Linux Conference, 2004, vol. 3.

[19] N. S. A. Peter Loscocco, "Security Vulnerabilities: An Impediment Against Further Development of Smart Grid," in Smart Grids from a Global Perspective, pp 77-93, 2016.

[20] N. Petreley, "Security report: Windows vs Linux. The Register, October 2004."

[21] A. Lackorzynski and A. Warg, "Less is More--A Secure Microkernel-Based Operating System," in SysSec Workshop (SysSec), 2011 First, 2011, pp. 103–106.

[22] R. K. Pal and I. Sengupta, "Enhancing File Data Security in Linux Operating System by Integrating Secure File System," in Computational Intelligence in Cyber Security, 2009. CICS '09. IEEE Symposium on, 2009, pp. 45–52.

[23] D. D. Clark and D. R. Wilson, "A comparison of commercial and military computer security policies," in Security and Privacy, 1987 IEEE Symposium on, 1987, p. 184.

[24] N. Provos, "Improving Host Security with System Call Policies.," in Usenix Security, 2003, vol. 3, p. 19.

[25] S. Schaefer, "Operating system abstraction and protection layer," in Proceedings of the 16<SUP>th </SUP>Symposium on Operating Systems Principles (SOPS), 'Online!, 2006, pp. 1–14.

[26] R. Luniya, A. Agarwal, M. Bhatnagar, V. Rathod, and D. Unwalla, "SmartX--Advanced Network Security for Windows Opearating System," in Intelligent Systems, Modelling and Simulation (ISMS), 2012 Third International Conference on, 2012, pp. 680–683.

[27] D. F. C. Brewer and M. J. Nash, "The chinese wall security policy," in Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on, 1989, pp. 206–214.

[28] B. Armstrong, P. England, S. A. Field, J. Garms, M. Kramer, and K. D. Ray, "Computer security management, such as in a virtual machine or hardened operating system," in Proceedings of the Computer Society Symposium on Research in Security and Privacy, IEEE Pub., 2008, pp. 2–19.

[29] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, "Remote data checking using provable data possession," ACM Trans. Inf. Syst. Secur., vol. 14, no. 1, p. 12, 2011.

[30] D. E. Bell, "Looking back at the Bell-La Padula model," in null, 2005, pp. 337–351.

[31] B. McCarty, Selinux: Nsa's open source security enhanced linux. O'Reilly Media, Inc., 2004.

[32] M. Richard, Solaris" Performance And Tools: Dtrace And Mdb Techniques For Solaris 10 And Opensolaris. Pearson Education India, 2007.

[33] K. K. Yue and D. J. Lilja, "Dynamic processor allocation with the Solaris operating system," in Parallel Processing Symposium, 1998. IPPS/SPDP 1998. Proceedings of the First Merged International... and Symposium on Parallel and Distributed Processing 1998, 1998, pp. 392–397.

[34] D. Price and A. Tucker, "Solaris Zones: Operating System Support for Consolidating Commercial Workloads.," in LISA, 2004, vol. 4, pp. 241–254.

[35] C.-Y. Yang, Y.-K. Huang, N.-C. Perng, J.-J. Chen, Y.-H. Lee, C.-M. Hung, H.-R. Hsu, S.-W. Huang, H.-W. Tseng, A.-C. Pang, and others, "Another real-time operating system and unified MAC protocol for home controlling and monitoring," in Software Technologies for Future Embedded and Ubiquitous Systems., 2006,

[36] E. H. B. M. Gronenschild, P. Habets, H. I. L. Jacobs, R. Mengelers, N. Rozendaal, J. Van Os, and M. Marcelis, "The effects of FreeSurfer version, workstation type, and Macintosh operating system version on anatomical volume and cortical thickness measurements," PLoS One, vol. 7, no. 6, p. e38234, 2012.

[37] C. Yi, Z. Zhi-rong, and S. Chang-xiang, "Design and implementation MAC in security operating system," in TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering, 2002, vol. 1, pp. 216–219.

[38] "Linux vs Windows - Difference and Comparison | Diffen." [Online]. Available: http://www.diffen.com/difference/Linux_vs_Windows. [Accessed: 08-Dec-2014].

[39] "Solaris Operating System." [Online]. Available: http://www.slideshare.net/wangluxh09/solaris-operating-system-36793409?related=1. [Accessed: 08-Dec-2014].

[40] "Mac, Windows And Solaris." [Online]. Available: http://www.slideshare.net/shivendra007/mac-windows-and-solaris. [Accessed: 08-Dec-2014].

[41] "Oracle Solaris 10 | Operating System | Oracle." [Online]. Available: http://www.oracle.com/us/products/servers-storage/solaris/solaris10/overview/index.html. [Accessed: 08-Dec-2014].

[42] Global Pricing and Licensing." [Online]. Available: http://www.oracle.com/us/corporate/pricing/index.html. [Accessed: 08-Dec-2014].

[43] "Solaris 11 GNOME 2.30." [Online]. Available: http://toastytech.com/guis/sol11.html. [Accessed: 08-Dec-2014].

[44] "Overview of File Systems - Oracle Solaris Administration: Devices and File Systems." [Online]. Available: https://docs.oracle.com/cd/E23824_01/html/821-1459/fsoverview-51.html. [Accessed: 08-Dec-2014].

[45] OS X: Mac OS Extended format (HFS Plus) volume and file limits. Support.apple.com, 2008.

[46] "Solaris Consoles and the Kernel Terminal Emulator." [Online]. Available: https://docs.oracle.com/cd/E19253-01/816-4854/6mb1o3bg7/index.html. [Accessed: 08-Dec-2014].

[47] "UNIX System Administration: Solaris, AIX, HP-UX, Tru64, BSD.: Gaming on Solaris." [Online]. Available: http://blog.boreas.ro/2007/08/gaming-on-solaris.html. [Accessed: 08-Dec-2014].

[48] Lineage online RPG now available as Mac OS X beta. .

[49] Oracle Completes Acquisition of Sun. Yahoo, 2010.

[50] "Mac OS X versions (builds) for computers - Apple Support." [Online]. Available: http://support.apple.com/en-us/ht1159. [Accessed: 08-Dec-2014].

[51] C. Zemao, W. Xiaoping, and T. Weimin, "An Executable Code Authorization Model for Secure Operating System," in Electronic Commerce and Security, 2008 International Symposium on, 2008, pp. 292–295.

[52] J. Viega and J. Voas, "The pros and cons of Unix and Windows security policies," IT Prof., vol. 2, no. 5, pp. 40–47, 2000.

[53] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and others, "TinyOS: An operating system for sensor networks," in Ambient intelligence, Springer, 2005, pp. 115–148.

[54] O. Services, "Oracle Solaris 10," Available: http://www.oracle.com/us/products/servers-storage/solaris/solaris10/overview/index.html. [Accessed: 08-Dec-2014]. 2014.

[55] X. Song, M. Stinson, R. Lee, and P. Albee, "An approach to analyzing the Windows and Linux security models," in Computer and Information Science, 2006 and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR 2006. 5th IEEE/ACIS International Conference on, 2006, pp. 56–62.

[56] M. E. Russinovich, D. A. Solomon, and J. Allchin, Microsoft Windows Internals: Microsoft Windows Server 2003, Windows XP, and Windows 2000, vol. 4. Microsoft Press Redmond, 2005.

[57] E. Zadok, R. Iyer, N. Joukov, G. Sivathanu, and C. P. Wright, "On incremental file system development," ACM Trans. Storage, vol. 2, no. 2, pp. 161–196, 2006.

[58] J. Dike, User mode linux, vol. 2. Prentice Hall Englewood Cliffs, 2006.

[59] J. Fox, "Getting started with the R commander: a basic-statistics graphical user interface to R," J. Stat. Softw., vol. 14, no. 9, pp. 1–42, 2005.

[60] B. Leiba, "Aspects of Internet security," IEEE Internet Comput., no. 4, pp. 72–75, 2012.

[61] R. Sailer, T. Jaeger, E. Valdez, R. Caceres, R. Perez, S. Berger, J. L. Griffin, and L. Van Doorn, "Building a MAC-based security architecture for the Xen open-source hypervisor," in Computer security applications conference, 21st Annual, 2005, p. 10–pp.

[62] M. G. Martinek, M. D. Jackson, D. R. Kingham, and T. S. Wasinger, "Video gaming apparatus for wagering with universal computerized controller and I/O interface for unique architecture." Google Patents, 2005.

[63] M. P. Casey, A. G. Englman, D. P. Fiden, J. M. Hornik, and J. R. Jaffe, "Gaming machine with interactive pop-up windows providing enhanced game play schemes." Google Patents, 2009.

[64] R. Godwin-Jones, "Emerging technologies: Messaging, gaming, peer-to-peer sharing: Language learning strategies & tools for the millennial generation," Lang. Learn. Technol., vol. 9, no. 1, pp. 17–22, 2005.

[65] Z. H. S. H. W. Fengke, "Application study of development kit gtk+ technique on linux gui [j]," Comput. Appl. Softw., vol. 1, p. 50, 2009.

[66] M. Gouy, S. Guindon, and O. Gascuel, "SeaView version 4: a multiplatform graphical user interface for sequence alignment and phylogenetic tree building," Mol. Biol. Evol., vol. 27, no. 2, pp. 221–224, 2010.