POSH APP

IT TECH RELOAD-PROFICIENCY OF SOFTWARE & HARDWARE ENGINEERING, HACKERS PRO, MICROSOFT OFFICE AND NETWORK MARKETING.

ICT Technician – ICTTech, Who is it for?

ICT Technicians or practitioners work in areas such as ICT hardware, software or system installation, operation, maintenance, incident/change/problem management, administration, security, fault diagnosis and fixing. They typically work in a range of jobs that involve supporting or facilitating the use of ICT equipment and applications by others and will be working in all fields of ICT, for example in data centres or on pcs or out in the field diagnosing and solving faults.

ICT Technicians can demonstrate underpinning knowledge and competence in the following: Supporting or enabling the use of ICT equipment and applications by others Selecting and using appropriate ICT resources, techniques, configurations, procedures and methods installing, operating, supporting or maintaining ICT systems Protecting ICT systems from intrusion, damage or data loss being personally responsible for ICT systems.

Why become an ICT Technician? Eg. Sinead Watts MITP ICTTech

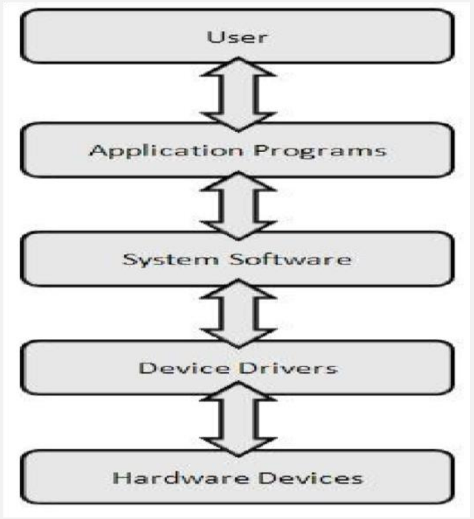
Professional registration sets an ICT Technician (ICTTech) apart from technicians who are not registered. In a fast-moving world of technologies and qualifications, it establishes proven competence and demonstrates commitment to professional standards as well as to life-long learning. Hence, whether specified in job advertisements or not, professional registration as a professional ICT Technician gives you an edge.

System Software

Computer software, or simply software, is a part of a computer system that consists of data or computer instructions, in contrast to the physical hardware from which the system is built.

Software is generally divided into two types: system software that keeps everything working, and application software that allows a user to accomplish some task **(even if that task is playing Chess Titan, solitaire or Heart)**. In this module, we will look primarily at system software. Application software and a third category, malware, will be discussed in following modules.

System software has the task of making your computer a usable system. All application programs work with the system software to accomplish their tasks. System software has three components: the operating system, system utilities **(OS helpers)**, and drivers. As can be seen at right, the OS interacts with hardware through drivers.



There is a plethora **(Large or excessive amount of something)** of computer hardware. For example, the number of disk drive models available, even from one manufacturer, can be surprisingly large. When you ask an application to "save" a file, it passes this request to the operating system. Rather than having the OS knows how to accomplish this task with every possible disk drive, it instead knows how to do it for a generic **(or virtual)** drive. The manufacturer of the disk drive provides a **(typically small)** program to implement the functionality on their particular hardware, i.e., the small program is the virtual disk drive. That program is called a driver.

An analogy that you may find useful is that the driver acts as a language interpreter. All interactions pass through the driver.

It is Key to note that, although the system you buy has many drivers pre-installed, those programs are provided by the device manufacturer. This is especially important if you are experiencing problems with a device or want to connect an unrecognized device. You will probably have to download a driver from the device manufacturer. That is, you might go to Western Digital for a disk driver, rather than Dell. **(Some OEMs provide an easy way to do this, but they are really just redistributing software provided to them.)**

Operating System

An **operating system** (**OS**) is software that manages computer hardware resources, runs other programs, and provides common services for the user and application software.

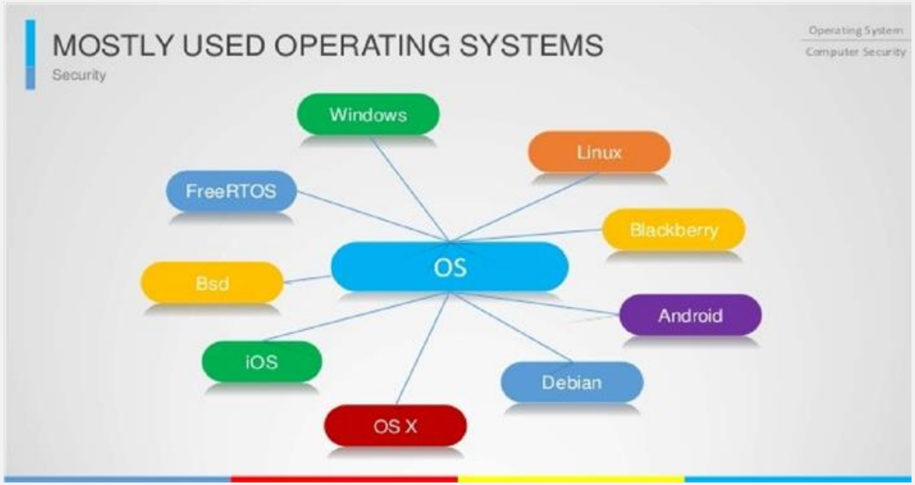
For hardware functions such as input & output and memory allocation, the operating system acts as an intermediary between application programs and the computer hardware. Operating systems are found on any device that contains a computer - from cellular phones and video game consoles to supercomputers and web servers.

Examples of popular modern operating systems are**: UNIX(Linux & Mac OS),Microsoft Windows** and various systems like **Android** and **iOS**. Early computers were built to perform a series of single tasks, like a calculator. Operating systems did not exist in their modern and more complex forms until the early 1960s. When personal (home) computers by

Companies such as **Apple Inc., Atari, IBM and Amiga** became popular in the 1980s, vendors added operating system features, such as multi-user and multi-tasking functionality, that had previously become widely used on mainframe and minicomputers **(that once bridged the gap between mainframes and PCs)**. Later, many features such as graphical user interfaces were developed specifically for personal computer operating systems.

An operating system consists of many parts. One of the most important components is the kernel, which controls low-level processes 8that the average user usually cannot see: it controls how memory is read and written, the order in which processes **(running programs)** are executed, how information is received and sent by devices like the monitor, keyboard and mouse, and decides how to interpret information received from networks. The user interface is a component that directly interacts with the computer user, allowing him or her to control and run programs. The user interface may be graphical with icons and a desktop, or textual, with a command line. Application programming interfaces provide services and code libraries that let applications developers write modular code.

Which features are considered part of the operating system is defined differently by some. For example, Microsoft Windows considers its user interface to be part of the operating system, as described above. On the other hand, UNIX and Linux have a long history of allowing the end user to choose, or even to create their own, user interface.



For our purposes, the operating system always resides in RAM while the computer is operating.

➢ Types of operating systems

▪ Single- And Multi– Tasking

A single-tasking system can only run one program at a time, while a multi-tasking operating system allows more than one program to be running in concurrency. This is achieved by time-sharing, dividing the available processor time between multiple processes that are each interrupted repeatedly in time slices by a task-scheduling subsystem of the operating system. Multi-tasking may be characterized in preemptive and co-operative types. In preemptive multitasking, the operating system slices the CPU time and dedicates a slot to each of the programs. UNIX-like operating systems, **e.g., Solaris, Linux, as well as AmigaOS support preemptive multitasking.** Cooperative multitasking is achieved by relying on each process to provide time to the other processes in a defined manner. 16-bitversions of Microsoft Windows used cooperative multi-tasking. 32-bitversions of both Windows NT and Win9x used preemptive multi-tasking.

▪ Single- And Multi-User

Single-user operating systems have no facilities to distinguish users, but may allow multiple programs to run in tandem. A multi-user operating system extends the basic concept of multi-tasking with facilities that identify processes and resources, such as **disk space**, belonging to multiple users, and the system permits multiple users to interact with the system at the same time. Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of **processor time, mass storage, printing, and other** resources to multiple users.

- **Distributed**

A distributed operating system manages a group of distinct computers and makes them appear to be a single computer. The development of networked computers that could be linked and communicate with each other gave rise to distributed computing. Distributed computations are carried out on more than one machine. When computers in a group work in cooperation, they form a distributed system.

- **Templated**

In an OS, distributed and cloud computing context, templating refers to creating a single virtual machine image as a guest operating system, then saving it as a tool for multiple running virtual machines. The technique is used both in virtualization and cloud computing management, and is common in large server warehouses.

- **Embedded**

Embedded operating systems are designed to be used in embedded computer systems**.** They are designed to operate on small machines like PDAs with less autonomy. They are able to operate with a limited number of resources. They are very compact and extremely efficient by design. Windows CE and Minx 3 are some examples of embedded operating systems.

- **Real-time**

A real - time operating system is an operating system that guarantees to process events or data by a specific moment in time. A real-time operating system may be single- or multi-tasking, but when multitasking, it uses specialized scheduling algorithms so that a deterministic nature of behavior is achieved. An event-driven system switches between tasks based on their priorities or external events while time-sharing operating systems switch tasks based on clock interrupts

Library

A library operating system is one in which the services that a typical operating system provides, such as networking, are provided in the form of libraries and composed with the application and configuration code to construct a unikernel: a specialized, single address space, machine image that can be deployed to cloud or embedded environments.

UNIX and UNIX-Like Operating Systems

UNIX was originally written in assembly language. Ken Thompson wrote B, mainly based on BCPL, based on his experience in the MULTICS project. B was replaced by C, and UNIX, rewritten in C, developed into a large, complex family of inter-related operating systems which have been influential in every modern operating system.

The **Unix-like** family is a diverse group of operating systems, with several major sub-categories including **System V,BSD, and Linux**. The name **"UNIX"** is a trademark of The Open Group which licenses it for use with any operating system that has been shown to conform to their definitions. **"UNIX-like"** is commonly used to refer to the large set of operating systems which resemble the original UNIX.

Unix-like systems run on a wide variety of computer architectures. They are used heavily for servers in business, as well as workstations in academic and engineering environments. Free UNIX variants, such as **Linux and BSD**, are popular in these areas.

Four operating systems are certified by The Open Group **(holder of the UNIX trademark)** as UNIX. HP's HP-UX and IBM's AIX are both descendants of the original System V Unix and are designed to run only on their respective vendor's hardware.

In contrast, Sun Microsystems's Solaris scan run on multiple types of hardware, including x86and Sparc servers, and PCs. Apple's macOS, a replacement for Apple's earlier **(non-Unix)** Mac OS, is a hybrid kernel-based BSD variant derived from **NeXTSTEP, Mach, and FreeBSD.** UNIX interoperability was sought by establishing the POSIX standard. The POSIX standard can be applied to any operating system, although it was originally created for various UNIX variants.

Evolution of UNIX systems.

BSD and Its Descendants

A subgroup of the UNIX family is the Distribution family, which includes **FreeBSD, NetBSD, and OpenBSD**. These operating systems are most commonly found on webservers, although they can also function as a personal computer OS. The Internet owes much of its existence to BSD, as many of the protocols now commonly used by computers to connect, send and receive data over a network were widely implemented and refined in BSD. The Web was also first demonstrated on a number of computers running an OS based on BSD called NeXTSTEP.

In 1974, Berkeley installed its first UNIX system. Over time, students and staff in the computer science department there began adding new programs to make things easier, such as text editors. When Berkeley received new VAX computers in 1978 with UNIX installed, the school's undergraduates modified UNIX even more in order to take advantage of the computer's hardware possibilities. The Defense Advanced Research Agency of the US Defense took interest, and decided to fund the project. Many schools, corporations, and government organizations took notice and started to use Berkeley's version of UNIX instead of the official one distributed by AT&T.

Steve Jobs, upon leaving Apple Inc. in 1985, formed NeXT Inc., a company that manufactured high-end computers running on a variation of BSD called NeXTSTEP. One of these computers was used by Berners-Lee as the first web server to create the World Wide Web.

Developers like Keith Bosticen courage the project to replace any non-free code that originated with Bell Labs. Once this was done, however, AT&T sued. After two years of legal disputes, the BSD project spawned a number of free derivatives, such as **NetBSD and FreeBSD** (both in 1993), and **OpenBSD** (from NetBSD in 1995).

The first server for the Web Ran **on NeXTSTEP, based on BSD.**

- ## MacOS

MacOS **(formerly "Mac OS X" and later "OS X")** is a line of core graphical operating systems developed, marketed, and sold by Apple Inc., the latest of which is pre-loaded on all currently shipping Macintosh computers. MacOS is the successor to the original classic Mac OS, which had been Apple's primary operating system since 1984. Unlike its predecessor, macOS is a UNIX operating system built on technology that had been developed at Next through the second half of the 1980s and up until Apple purchased the company in early 1997. The operating system was first released in 1999 as **Mac OS XServer 1.0,** followed in March 2001 by a client version **(Mac OS X v10.0 "Cheetah").**

Since then, six more distinct "client" and "server" editions of macOS have been released, until the two were merged in OS X 10.7 **"Lion"**.
Prior to its merging with macOS, the server edition - macOS Server- was architecturally identical to its desktop counterpart and usually ran on Apple's line of Macintosh server hardware. macOS Server included work group management and administration software tools that provide simplified access to key network services, including a mail transfer agent, a Samba server, an LDAP server, a server, and others.

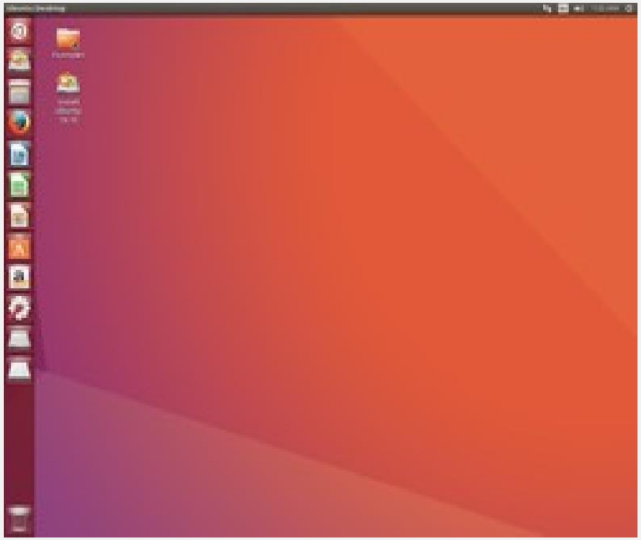
With Mac OS X v10.7 Lion, all server aspects of Mac OS X Server have been integrated into the client version and the product re-branded as **"OS X" (dropping "Mac" from the name)**. The server tools are now offered as an application.

- ## Linux

The Linux kernel originated in 1991, as a project of Linus Torvalds, while a university student in Finland. He posted information about his project on a newsgroup for computer students and programmers, and received support and assistance from volunteers who succeeded in creating a complete and functional kernel.

Linux is Unix-like, but was developed without any UNIX code, unlike BSD and its variants. Because of its open license model, the Linux kernel code is available for study and modification, which resulted in its use on a wide range of computing machinery from supercomputers to smart-watches. Although estimates suggest that Linux is used on only **1.82%** of all **"desktop"** (or laptop) PCs, it has been widely adopted for use in servers and embedded systems such as cell phones.

Linux has superseded UNIX on many platforms and is used on most supercomputers including the top 385. Many of the same computers are also on Green 500(but in different order), and Linux runs on the top 10. Linux is also commonly used on other small energy-efficient computers, such as Smartphone and smart watches. The Linux kernel is used in some popular distributions, such as **Red Hat, Debian, Ubuntu, Linux Mint and Google's Android, Chrome OS, and Chromium OS.**



Ubuntu, desktop Linux distribution

■ Microsoft Windows

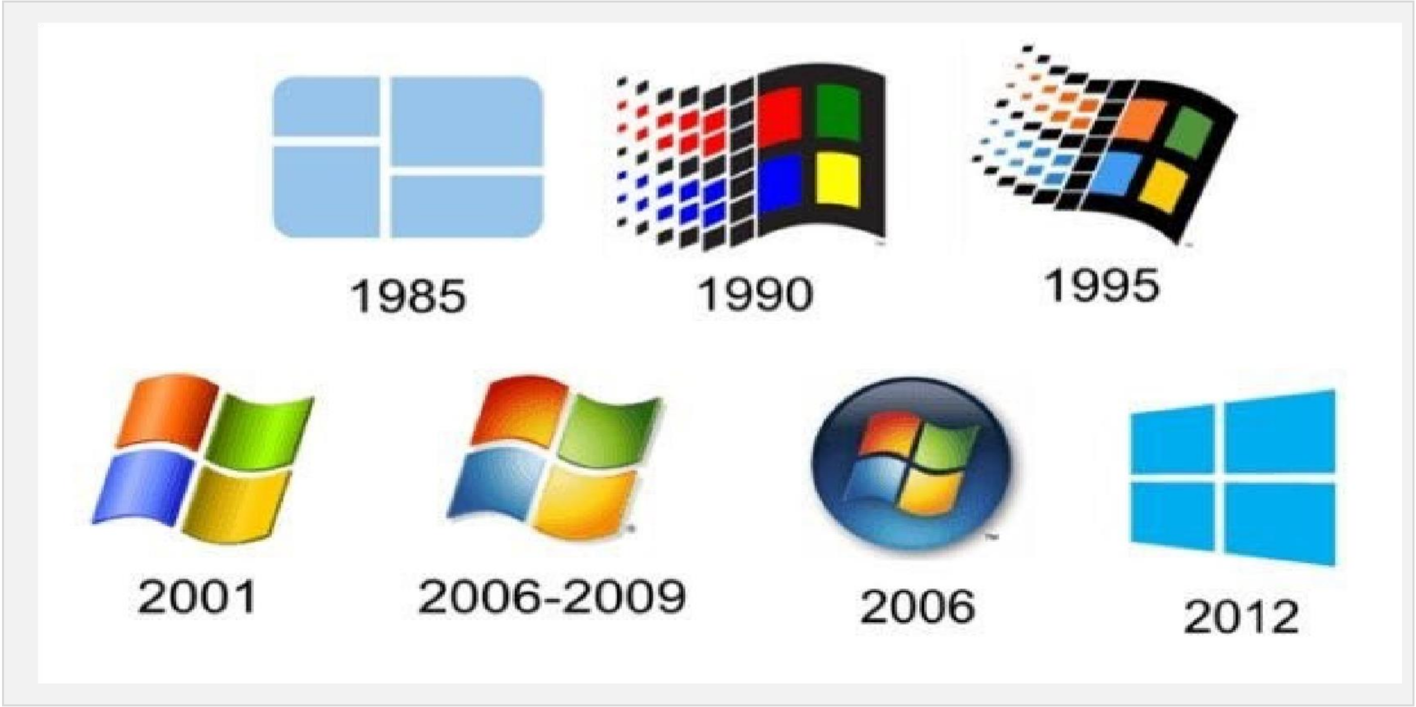
Microsoft Windows is a family of proprietary operating systems designed by Microsoft Corporation and primarily targeted to Intel architecture based computers, with an estimated 88.9 percent total usage share on Web connected computers. The latest version **is Windows 10.In 2011, Windows 7 overtook Windows XP** as most common version in use.

Microsoft Windows was first released in 1985, as an environment running on top of MS-DOS, which was the standard operating system shipped on most Intel architecture personal computers at the time. In 1995, Windows 95was released which only used MS-DOS as a bootstrap.

For backwards compatibility, Win9x could run real-mode MS-DOS and 16-bit Windows 3.xdrivers. Windows ME, released in 2000, was the last version in the Win9x family. Later versions have all been based on the Windows NT kernel. Current client versions of Windows run on IA-32,x86-64and 32-bit ARM microprocessors. In addition Itanium is still supported in older server version Windows Server 2008 R2. In the past, Windows NT supported additional architectures.

Server editions of Windows are widely used. In recent years, Microsoft has expended significant capital in an effort to promote the use of Windows as a server operating system.

However, Windows' usage on servers is not as widespread as on personal computers as Windows competes against Linux and BSD for server market share. ReactOS is a Windows-alternative operating system, which is being developed on the principles of Windows - without using any of Microsoft's code.


1985
1990
1995
2001
2006-2009
2006
2012

Other

There have been many operating systems that were significant in their day but are no longer so, such as AmigaOS; OS/2from IBM and Microsoft; classic Mac OS, the non-Unix precursor to **Apple's macOS; BeOS; XTS-300;RISC OS; MorphOS; Haiku; Bare Metal and Free Mint.** Some are still used in niche markets and continue to be developed as minority platforms for enthusiast communities and specialist applications. OpenVMS, formerly from DEC, is still under active development by Hewlett-Packard. Yet other operating systems are used almost exclusively in academia, for operating systems education or to do research on operating system concepts. A typical example of a system that fulfills both roles is MINIX, while for example Singularity is used purely for research. Other operating systems have failed to win significant market share, but have introduced innovations that have influenced mainstream operating systems, not **least Bell Labs' Plan 9**.

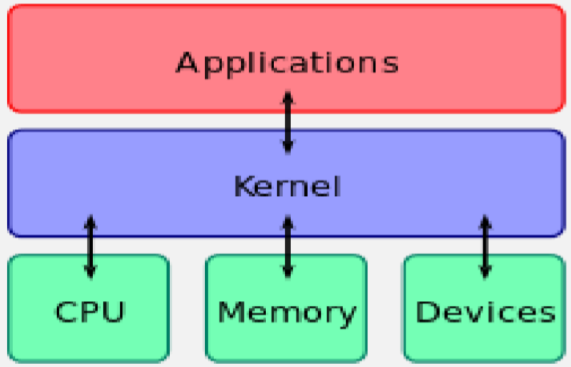
Components

The components of an operating system all exist in order to make the different parts of a computer work together. All user software needs to go through the operating system in order to use any of the hardware, whether it is as simple as a **mouse or keyboard** or as complex as an Internet component.

- ## Kernel

With the aid of the firmware and device drivers, the kernel provides the most basic level of control over all of the computer's hardware devices. It manages memory access for programs in the RAM, it determines which programs get access to which hardware resources, it sets up or resets the CPU's operating states for optimal operation at all times, and it organizes the data for long-term non-volatile storage with file systems on such media as disks, tapes, flash memory, etc.

A kernel connects the application software to the Hardware of a computer.



Program Execution

The operating system provides an interface between an application program and the computer hardware, so that an application program can interact with the hardware only by obeying rules and procedures programmed into the operating system. The operating system is also a set of services which simplify development and execution of application programs. Executing an application program involves the creation of a process by the operating system kernel which assigns memory space and other resources, establishes a priority for the process in multi-tasking systems, and loads program binary code into memory, and initiates execution of the Application program which then interacts with the user and with hardware devices.

Interrupts

Interrupts are central to operating systems, as they provide an efficient way for the operating system to interact with and react to its environment. The alternative - having the operating system **"watch"** the various sources of input for events (polling) that require action - can be found in older systems with very small stacks**(50 or 60 bytes)** but is unusual in modern systems with large stacks. Interrupt-based programming is directly supported by most modern CPUs. Interrupts provide a computer with a way of automatically saving local register contexts, and running specific code in response to events. Even very basic computers support hardware interrupts, and allow the programmer to specify code which may be run when that event takes place.

When an interrupt is received, the computer's hardware automatically suspends whatever program is currently running, saves its status, and runs computer code previously associated with the interrupt; this is analogous to placing a bookmark in a book in response to a phone call. In modern operating systems, interrupts are handled by the operating system's kernel. Interrupts may come from either the computer's hardware or the running program.

When a hardware device triggers an interrupt, the operating system's kernel decides how to deal with this event, generally by running some processing code. The amount of code being run depends on the priority of the interrupt **(for example: a person usually responds to a smoke detector alarm before answering the phone)**. The processing of hardware interrupts is a task that is usually delegated to software called a device driver, which may be part of the operating system's kernel, part of another program, or both. Device drivers may then relay information to a running program by various means.

A program may also trigger an interrupt to the operating system. If a program wishes to access hardware, for example, it may interrupt the operating system's kernel, which causes control to be passed back to the kernel. The kernel then processes the request. If a program wishes additional resources **(or wishes to shed resources)** such as memory, it triggers an interrupt to get the kernel's attention.

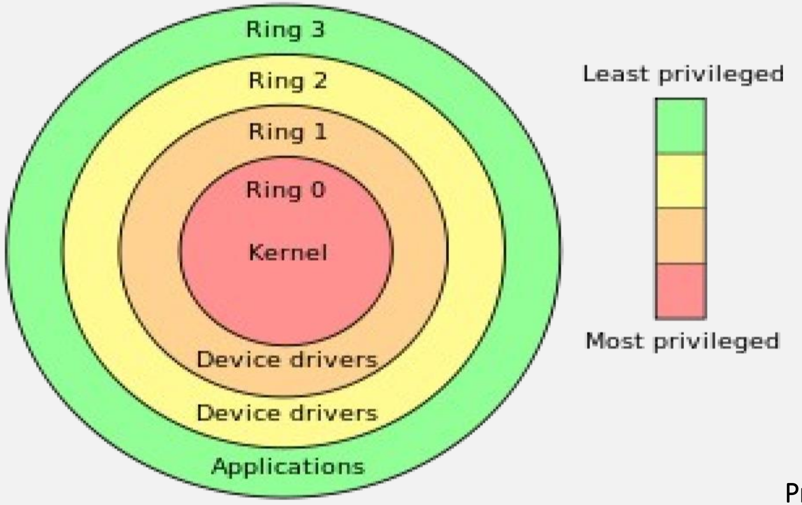
▪ Modes

Odern microprocessors **(CPU or MPU)** support multiple modes of operation. CPUs with this capability offer at least two modes: user mode and supervisor mode. In general terms, supervisor mode operation allows unrestricted access to all machine resources, including all MPU instructions. User mode operation sets limits on instruction use and typically disallows direct access to machine resources. CPUs might have other modes similar to user mode as well, such as the virtual modes in order to emulate older processor types, such as 16-bit processors on a 32-bit one, or 32-bit processors on a 64-bitone.At power-on or reset, the system begins in supervisor mode. Once an operating system kernel has been loaded and started, the boundary between user mode and supervisor mode **(also known as kernel mode)** can be established.

Managers, operate within user mode, and can only access machine resources by turning control over to the kernel, a process which causes a switch to supervisor mode. Typically, the transfer of control to the kernel is achieved by executing a software interrupt instruction, such as the Motorola 68000 `TRAP` instruction. The software interrupt causes the microprocessor to switch from user mode to supervisor mode and begin executing code that allows the kernel to take control.

In user mode, programs usually have access to a restricted set of microprocessor instructions, and generally cannot execute any instructions that could potentially cause disruption to the system's operation. In supervisor mode, instruction execution restrictions are typically removed, allowing the kernel unrestricted access to all machine resources.

The term **"user mode resource"** generally refers to one or more CPU registers, which contain information that the running program isn't allowed to alter. Attempts to alter these resources generally causes a switch to supervisor mode, where the operating system can deal with the illegal operation the program was attempting, for example, by forcibly **terminating ("killing") the program)**.



Privilege rings for the x86microprocessorArchitecture available in protected mode. Operating systems determine which processes run in each mode.

▪ **Memory Management**

Among other things, a multiprogramming operating system kernel must be responsible for managing all system memory which is currently in use by programs. This ensures that a program does not interfere with memory already in use by another program. Since programs time share, each program must have independent access to memory.

Cooperative memory management, used by many early operating systems, assumes that all programs make voluntary use of the kernel's memory manager, and do not exceed their allocated memory. This system of memory management is almost never seen any more, since programs often contain bugs which can cause them to exceed their allocated memory. If a program fails, it may cause memory used by one or more other programs to be affected or overwritten. Malicious programs or viruses may purposefully alter another program's memory, or may affect the operation of the operating system itself. With cooperative memory management, it takes only one misbehaved program to crash the system.

Memory protection enables the kernel to limit a process' access to the computer's memory. Various methods of memory protection exist, including memory segmentation and paging. All methods require some level of hardware support **(such as the 80286MMU)**, which doesn't exist in all computers.

In both segmentation and paging, certain protected mode registers specify to the CPU what memory address it should allow a running program to access. Attempts to access other addresses trigger an interrupt which cause the CPU to re-enter supervisor mode, placing the kernel in charge. This is called a segmentation violation or Seg-V for short and since it is both difficult to assign a meaningful result to such an operation, and because it is usually a sign of a misbehaving program, the kernel generally resorts to terminating the offending program, and reports the error.

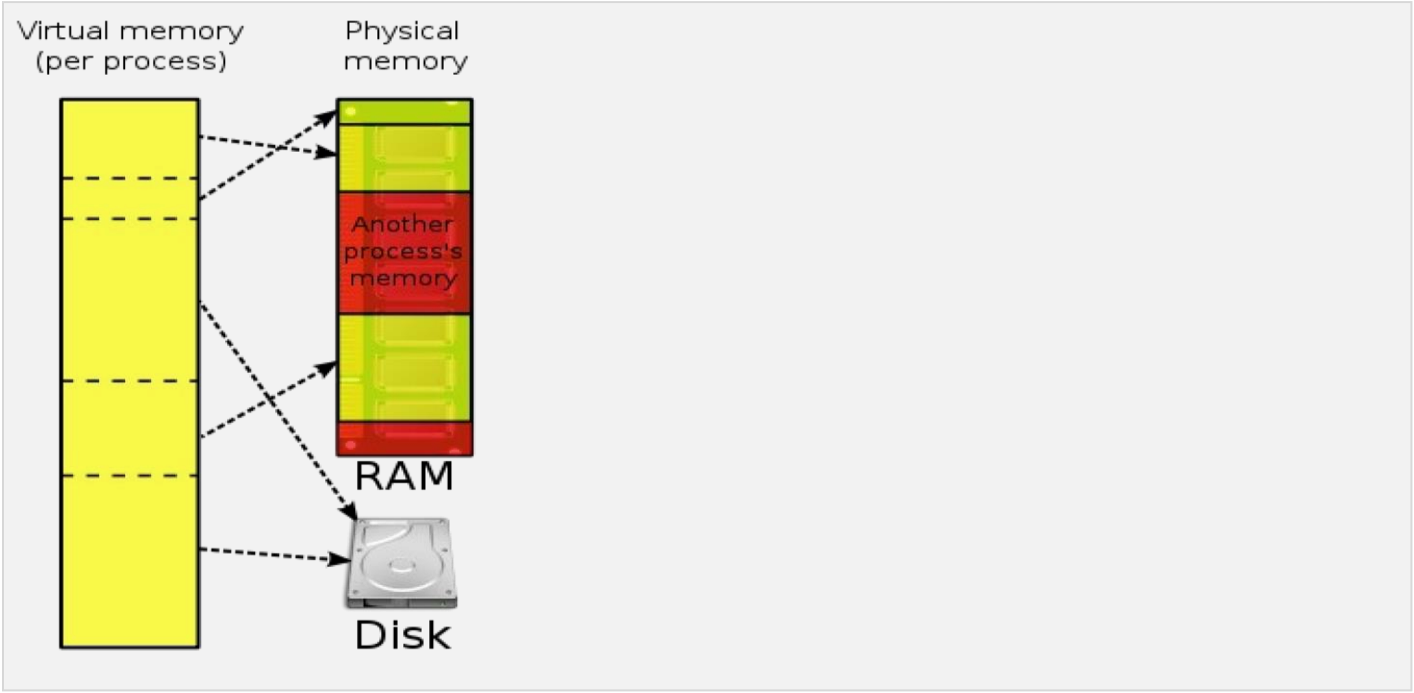
Windows versions 3.1 through ME had some level of memory protection, but programs could easily circumvent the need to use it. A general protection fault would be produced, indicating a segmentation violation had occurred; however, the system would often crash anyway.

Virtual Memory

The use of virtual memory addressing **(such as paging or segmentation)** means that the kernel can choose what memory each program may use at any given time, allowing the operating system to use the same memory locations for multiple tasks. If a program tries to access memory that isn't in its current range of accessible memory, but nonetheless has been allocated to it, the kernel is interrupted in the same way as it would if the program were to exceed its allocated memory. **(See section on memory management.)** Under UNIX this kind of interrupt is referred to as a page fault.

When the kernel detects a page fault it generally adjusts the virtual memory range of the program which triggered it, granting it access to the memory requested. This gives the kernel discretionary power over where a particular application's memory is stored, or even whether or not it has actually been allocated yet.

In modern operating systems, memory which is accessed less frequently can be temporarily stored on disk or other media to make that space available for use by other programs. This is called swapping, as an area of memory can be used by multiple programs, and what that memory area contains can be swapped or exchanged on demand.**"Virtual memory"** provides the programmer or the user with the perception that there is a much larger amount of RAM in the computer than is really there.



- ## Multitasking

Multitasking refers to the running of multiple independent computer programs on the same computer; giving the appearance that it is performing the tasks at the same time. Since most computers can do at most one or two things at one time, this is generally done via time-sharing, which means that each program uses a share of the computer's time to execute.

An operating system kernel contains a scheduling program which determines how much time each process spends executing, and in which order execution control should be passed to programs. Control is passed to a process by the kernel, which allows the program access to the CPU and memory. Later, control is returned to the kernel through some mechanism, so that another program may be allowed to use the CPU. This so-called passing of control between the kernel and applications is called a context switch.

An early model which governed the allocation of time to programs was called cooperative multitasking. In this model, when control is passed to a program by the kernel, it may execute for as long as it wants before explicitly returning control to the kernel. This means that a malicious or malfunctioning program may not only prevent any other programs from using the CPU, but it can hang the entire system if it enters a loop. Modern operating systems extend the concepts of application preemption to device drivers and kernel code, so that the operating system has preemptive control over internal run-times as well.

The philosophy governing preemptive multitasking is that of ensuring that all programs are given regular time on the CPU. This implies that all programs must be limited in how much time they are allowed to spend on the CPU without being interrupted. To accomplish this, modern operating system kernels make use of a timed interrupt. A protected mode timer is set by the kernel which triggers a return to supervisor mode after the specified time has elapsed. **(See above sections on Interrupts and Dual Mode Operation.)**
On many single user operating systems cooperative multitasking is perfectly adequate, as home computers generally run a small number of well tested programs. The AmigaOS is an exception, having preemptive multitasking from its very first version. Windows NT was the first version of Microsoft Windows which enforced preemptive multitasking, but it didn't reach the home user market until Windows XP**(since Windows NT was targeted at professionals).**

Disk Access And File Systems

Access to data stored on disks is a central feature of all operating systems. Computers store data on disks using files, which are structured in specific ways in order to allow for faster access, higher reliability, and to make better use of the drive's available space. The specific way in which files are stored on a disk is called a file system, and enables files to have names and attributes. It also allows them to be stored in a hierarchy of directories or folders arranged in a directory tree.

Early operating systems generally supported a single type of disk drive and only one kind of file system. Early file systems were limited in their capacity, speed, and in the kinds of file names and directory structures they could use. These limitations often reflected limitations in the operating systems they were designed for, making it very difficult for an operating system to support more than one file system.

While many simpler operating systems support a limited range of options for accessing storage systems, operating systems like UNIX and Linux support a technology known as a virtual file system or VFS. An operating system such as UNIX supports a wide array of storage devices, regardless of their design or file systems, allowing them to be accessed through a common application programming interface (API). This makes it unnecessary for programs to have any knowledge about the device they are accessing. A VFS allows the operating system to provide programs with access to an unlimited number of devices with an infinite variety of file systems installed on them, through the use of specific device drivers and file system drivers.

A connected storage device, such as a hard drive, is accessed through a device driver. The device driver understands the specific language of the drive and is able to translate that language into a standard language used by the operating system to access all disk drives. On UNIX, this is the language of block devices.

When the kernel has an appropriate device driver in place, it can then access the contents of the disk drive in raw format, which may contain one or more file systems. A file system driver is used to translate the commands used to access each specific file system into a standard set of commands that the operating system can use to talk to all file systems. Programs can then deal with these files systems on the basis of filenames, and directories/folders, contained within a hierarchical structure. They can create, delete, open, and close files, as well as gather various information about them, including access permissions, size, and free space, and creation and modification dates. Various differences between file systems make supporting all file systems difficult. Allowed characters in file names, case sensitivity, and the presence of various kinds of file attributes makes the implementation of a single interface for every file system a daunting task. Operating systems tend to recommend using **(and so support natively)** file systems specifically designed for them; for example, NTFS in Windows and ext3and ReiserFSin Linux. However, in practice, third party drivers are usually available to give support for the most widely used file systems in most general-purpose operating systems **(for example, NTFS is available in Linux through NTFS-3g, and ext2/3 and ReiserFS are available in Windows through third-party software)**.

Support for file systems is highly varied among modern operating systems, although there are several common file systems which almost all operating systems include support and drivers for. Operating systems vary on file system support and on the disk formats they may be installed on. Under Windows, each file system is usually limited in application to certain media; for example, CDs must use ISO 9660or UDF, and as of Windows Vista, NTFS is the only file system which the operating system can be installed on. It is possible to install Linux onto many types of file systems.

Unlike other operating systems, Linux and UNIX allow any file system to be used regardless of the media it is stored in, whether it is a hard drive, a disc **(CD, DVD...)**, a USB flash drive, or even contained within a file located on another file system.

File systems allow users and programs to Organize and sort files on a computer, often through the use of directories (or "folders").

▪ Device Drivers

A device driver is a specific type of computer software developed to allow interaction with hardware devices. Typically this constitutes an interface for communicating with the device, through the specific computer bus or communications subsystem that the hardware is connected to, providing commands to and/or receiving data from the device, and on the other end, the requisite interfaces to the operating system and software applications. It is a specialized hardware-dependent computer program which is also operating system specific that enables another program, typically an operating system or applications software package or computer program running under the operating system kernel, to interact transparently with a hardware device, and usually provides the requisite interrupt handling necessary for any necessary asynchronous time-dependent hardware interfacing needs.

The key design goal of device drivers is abstraction. Every model of **hardware (even within the same class of device)** is different. Newer models also are released by manufacturers that provide more reliable or better performance and these newer models are often controlled differently. Computers and their operating systems cannot be expected to know how to control every device, both now and in the future. To solve this problem, operating systems essentially dictate how every type of device should be controlled. The function of the device driver is then to translate these operating system mandated function calls into device specific calls. In theory a new device, which is controlled in a new manner, should function correctly if a suitable driver is available. This new driver ensures that the device appears to operate as usual from the operating system's point of view.

Under versions of Windows before Vista and versions of Linux before 2.6, all driver execution was co-operative, meaning that if a driver entered an infinite loop it would freeze the system. More recent revisions of these operating systems incorporate kernel preemption, where the kernel interrupts the driver to give it tasks, and then separates itself from the process until it receives a response from the device driver, or gives it more tasks to do.

▪ Networking

Currently most operating systems support a variety of networking protocols, hardware, and applications for using them. This means that computers running dissimilar operating systems can participate in a common network for sharing resources such as computing, files, printers, and scanners using either wired or wireless connections. Networks can essentially allow a computer's operating system to access the resources of a remote computer to support the same functions as it could if those resources were connected directly to the local computer. This includes everything from simple communication, to using networked file systems or even sharing another computer's graphics or sound hardware. Some network services allow the resources of a computer to be accessed transparently, such as SSH which allows networked users direct access to a computer's command line interface.

Client/server networking allows a program on a computer, called a client, to connect via a network to another computer, called a server. Servers offer (or host) various services to other network computers and users. These services are usually provided through ports or numbered access points beyond the server's IP address. Each port number is usually associated with a maximum of one running program, which is responsible for handling requests to that port. A daemon, being a user program, can in turn access the local hardware resources of that computer by passing requests to the operating system kernel.

Many operating systems support one or more vendor-specific or open networking protocols as well, for example, SNA on IBM systems, DECnet on systems from Digital Equipment Corporation, and Microsoft-specific protocols **(SMB)** on Windows. Specific protocols for specific tasks may also be supported such as NFS for file access. Protocols like ESound, or esd can be easily extended over the network to provide sound from local applications, on a remote system's sound hardware.

- ## Security

A computer being secure depends on a number of technologies working properly. A modern operating system provides access to a number of resources, which are available to software running on the system, and to external devices like networks via the kernel.

The operating system must be capable of distinguishing between requests which should be allowed to be processed, and others which should not be processed. While some systems may simply distinguish between "privileged" and "non-privileged", systems commonly have a form of requester identity, such as a user name. To establish identity there may be a process of authentication. Often a username must be quoted, and each username may have a password. Other methods of authentication, such as magnetic cards or biometric data, might be used instead. In some cases, especially connections from the network, resources may be accessed with no authentication at all **(such as reading files over a network share).** Also covered by the concept of requester identity is authorization; the particular services and resources accessible by the requester once logged into a system are tied to either the requester's user account or to the variously configured groups of users to which the requester belongs.

In addition to the allow or disallow model of security, a system with a high level of security also offers auditing options. These would allow tracking of requests for access to resources **(such as, "who has been reading this file?").** Internal security or security from an already running program is only possible if all possibly harmful requests must be carried out through interrupts to the operating system kernel. If programs can directly access hardware and resources, they cannot be secured.

External security involves a request from outside the computer, such as a login at a connected console or some kind of network connection. External requests are often passed through device drivers to the operating system's kernel, where they can be passed onto applications, or carried out directly. Security of operating systems has long been a concern because of highly sensitive data held on computers, both of a commercial and military nature. The United States Government **Department of Defense(DoD)** created the **Trusted Computer System Evaluation Criteria(TCSEC)** which is a standard that sets basic requirements for assessing the effectiveness of security. This became of vital importance to operating system makers, because the TCSEC was used to evaluate, classify and select trusted operating systems being considered for the processing, storage and retrieval of sensitive.

- ## Classified Information

Network services include offerings such as file sharing, print services, email, web sites, and file transfer protocols **(FTP)**, most of which can have compromised security. At the front line of security are hardware devices known as firewalls or intrusion detection/prevention systems. At the operating system level, there are a number of software firewalls available, as well as intrusion detection/prevention systems. Most modern operating systems include a software firewall, which is enabled by default. A software firewall can be configured to allow or deny network traffic to or from a service or application running on the operating system. Therefore, one can install and be running an insecure service, such as Telnet or FTP, and not have to be threatened by a security breach because the firewall would deny all traffic trying to connect to the service on that port.

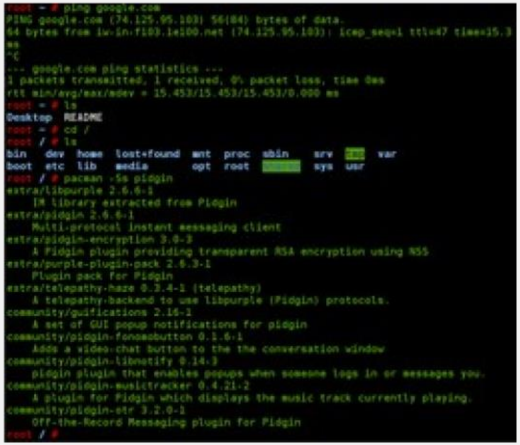
An alternative strategy, and the only sandbox strategy available in systems that do not meet the Popek and Goldberg virtualization requirements, is where the operating system is not running user programs as native code, but instead either emulates a processor or provides a host for a p-code based system such as Java.

Internal security is especially relevant for multi-user systems; it allows each user of the system to have private files that the other users cannot tamper with or read. Internal security is also vital if auditing is to be of any use, since a program can potentially bypass the operating system, inclusive of bypassing auditing.

- ### User Interface

Every computer that is to be operated by an individual requires a user interface. The user interface is usually referred to as a shell and is essential if human interaction is to be supported. The user interface views the directory structure and requests services from the operating system that will acquire data from input hardware devices, such as a keyboard, mouse or credit card reader, and requests operating system services to display prompts, status messages and such on output hardware devices, such as a video monitor or printer.

The two most common forms of a user interface have historically been the command-line interface, where computer commands are typed out line-by-line, and the graphical user interface, where a visual environment **(most commonly a WIMP)** is present.



```
root ~ # ping google.com
PING google.com (74.125.95.103) 56(84) bytes of data.
64 bytes from iw-in-f103.1e100.net (74.125.95.103): icmp_seq=1 ttl=47 time=15.3
ms
^C
--- google.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 15.453/15.453/15.453/0.000 ms
root ~ # ls
Desktop  README
root ~ # cd /
root / # ls
bin   dev  home  lost+found  mnt  proc  sbin         srv  [illegible]  var
boot  etc  lib   media       opt  root  [illegible]  sys  usr
root / # pacman -Ss pidgin
extra/libpurple 2.6.6-1
    IM library extracted from Pidgin
extra/pidgin 2.6.6-1
    Multi-protocol instant messaging client
extra/pidgin-encryption 3.0-3
    A Pidgin plugin providing transparent RSA encryption using NSS
extra/purple-plugin-pack 2.6.3-1
    Plugin pack for Pidgin
extra/telepathy-haze 0.3.4-1 (telepathy)
    A telepathy-backend to use libpurple (Pidgin) protocols.
community/guifications 2.16-1
    A set of GUI popup notifications for pidgin
community/pidgin-fonomobutton 0.1.6-1
    Adds a video-chat button to the the conversation window
community/pidgin-libnotify 0.14-3
    pidgin plugin that enables popups when someone logs in or messages you.
community/pidgin-musictracker 0.4.21-2
    A plugin for Pidgin which displays the music track currently playing.
community/pidgin-otr 3.2.0-1
    Off-the-Record Messaging plugin for Pidgin
root / #
```

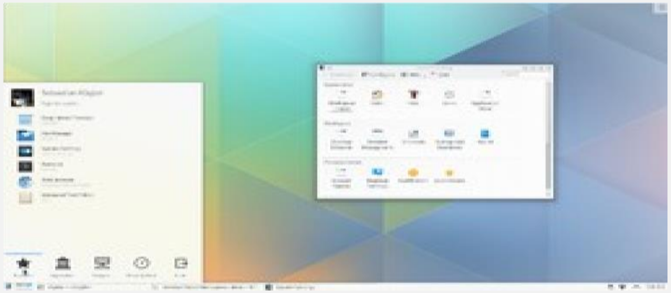
A screenshot of the Bash command line. Each Command is typed out after the 'prompt', and then its output appears below, working its way down the screen. The current command prompt is at the bottom.

▪ Graphical User Interfaces

Most of the modern computer systems support graphical user interfaces (GUI), and often include them. In some computer systems, such as the original implementation of the classic Mac OS, the GUI is integrated into the kernel. While technically a graphical user interface is not an operating system service, incorporating support for one into the operating system kernel can allow the GUI to be more responsive by reducing the number of context switches required for the GUI to perform its output functions. Other operating systems are modular, separating the graphics subsystem from the kernel and the Operating System. In the 1980s UNIX, VMS and many others had operating systems that were built this way. Linux and macOS are also built this way. Modern releases of Microsoft Windows such as Windows Vista implement a graphics subsystem that is mostly in user-space; however the graphics drawing routines of versions between Windows NT 4.0and Windows Server 2003exist mostly in kernel space. Windows 9xhad very little distinction between the interface and the kernel.

Many computer operating systems allow the user to install or create any user interface they desire. The X Window System in conjunction with GNOME or KDE Plasma 5is a commonly found setup on most Unix and Unix-like**(BSD, Linux, Solaris)** systems. A number of Windows shell replacement shave been released for Microsoft Windows, which offer alternatives to the included Windows shell, but the shell Numerous Unix-based GUIs have existed over time, most derived from X11.

Competition among the various vendors of Unix **(HP, IBM, Sun)** led to much fragmentation, though an effort to standardize in the 1990s to COSE and CDE failed for various reasons, and were eventually eclipsed by the widespread adoption of GNOME and K Desktop Environment. Prior to free software-based toolkits and desktop environments, Motif was the prevalent toolkit/desktop combination **(and was the basis upon which CDE was developed)**. Graphical user interfaces evolve over time. For example, Windows has modified its user interface almost every time a new major version of Windows is released, and the Mac OS GUI changed dramatically with the introduction of Mac OS X in 1999.It cannot be separated from Windows.



A screenshot of the KDE Plasma 5graphical user Interface. Programs take the form of images on the screen, and the files, folders (directories), and applications take the form of icons and symbols. A mouse is used to navigate the computer.

▪ Real-Time Operating Systems

A real-time operating system (RTOS) is an operating system intended for applications with fixed deadlines **(real-time computing)**. Such applications include some small embedded systems, automobile engine controllers, industrial robots, spacecraft, industrial control, and some large-scale computing systems. An early example of a large-scale real-time operating system was Transaction Processing Facility developed by American Airlines and IBM for the Sabre Airline Reservations System.

Embedded systems that have fixed deadlines use a real-time operating system such as **VxWorks, PikeOS, eCos, QNX, MontaVista Linux and RTLinux**. Windows CE is a real-time operating system that shares similar APIs to desktop Windows but shares none of desktop Windows' codebase. Symbian OS also has an RTOS kernel **(EKA2)** starting with version 8.0 b. Some embedded systems use operating systems such as **Palm OS,BSD, and Linux,** although such operating systems do not support real-time computing.

➢ Operating System Development as a Hobby.

Operating system development is one of the most complicated activities in which a computing hobbyist may engage. A hobby operating system may be classified as one whose code has not been directly derived from an existing operating system, and has few users and active.

In some cases, hobby development is in support of a **"homebrew"** computing device, for example, a simple single-board computer powered by a **6502 microprocessor**. Or, development may be for architecture already in widespread use. Operating system development may come from entirely new concepts, or may commence by modeling an existing operating system. In either case, the hobbyist is his/her own developer, or may interact with a small and sometimes unstructured group of individuals who have like interests.

- Examples Of A Hobby Operating System Include Syllable.

➢ **Diversity Of Operating Systems And Portability**

Application software is generally written for use on a specific operating system, and sometimes even for specific hardware. When porting the application to run on another OS, the functionality required by that application may be implemented differently by that OS **(the names of functions, meaning of arguments, etc.)** requiring the application to be adapted, changed, or otherwise maintained.

UNIX was the first operating system not written in assembly language, making it very portable to systems different from its native PDP-11.

This cost in supporting operating systems diversity can be avoided by instead writing applications against software platforms like Java or Qt. These abstractions have already borne the cost of adaptation to specific operating systems and their libraries. Another approach is for operating system vendors to adopt standards. For example, **POSIX and OS abstraction layers** provide commonalities that reduce porting costs.

Market Share

2013 worldwide device shipments by operating system

Operating system	2012 (millions of units)	2013 (millions of units)
Android	504	878
Windows	346	328
iOS/Mac OS	214	267
BlackBerry	35	24
Others	1,117	803
Total	2,216	2,300

In 2014, Android was first **(currently not replicated by others, in a single year)** operating system ever to ship on a billion devices, becoming the most popular operating system by installed base.

Comparison of Operating Systems.

These tables provide a comparison of operating systems, of computer devices, as listing general and technical information for a number of widely used and currently available PC or handheld **(including Smartphone and tablet computer)** operating systems. The article "**Usage share of operating systems"** provides a broader, and more general, comparison of operating systems that includes servers, mainframes and supercomputers.

Because of the large number and variety of available Linux distributions, they are all grouped under a single entry; see comparison of Linux distributions for a detailed comparison. There is also a variety of BSD and DOS operating systems, covered in comparison of BSD operating systems and comparison of DOS operating systems.

For information on views of each operating system, see operating system advocacy.

• General Information

4.1 Available As Standalone Package at €29

Hardware

Propriety Software Novell Open Enterprise Server ; Was US$184 (Equivalent To $205.41 In 2016) (One-user)

2017, October 10

Bundled With Hardware

Googl Of

2017, March 27

Free

BSD

Discontinued Was Bundled with Hardware, Then sold Separately

2016, April 4

1995

Proprietary

Workstation

2015, September 13

Server, NAS, Workstation

Home-student edition

2016 , March 29

Free

IS C

Max. Three

Embedded

Per site) US$145.00

2011 (Equivalent To $154.37 In 2016)

2017 , May 3

Free

CDD L

Server, Workstation

Business Edition $290.00

Discontinued

Commercial, Free Non-commer cial use

2016 , September 23

Proprietary

Server, Workstation

1999 Commercial

- ## Embedded operating system

While the distinction is mostly import to technophiles, hand-held devices (**including early Smartphone)** use (d) a class of operating system that is called **"embedded"**. (Of embedded.) Embedded systems, on the other hand, is a term generally used to refer to (usually real-time) systems with limited functionality. Embedded systems are quite common. They are in many devices, such as **cars, microwave ovens, printers, etc**. Embedded-OS devices are becoming much more common.

- ## User Interface

From a user perspective, the user interface **(typically a GUI)** *is* the operating system. Actually this is just the way you, the user, can talk directly to the OS to accomplish system level tasks. As noted above, Microsoft considers the user interface to be a part of the OS, while some others, notably Linux, allow you to choose **(and change)** the interface. In the workplace, you typically will not have a choice because IT departments like to keep to one kind of system.

The video at right follows the progression of Microsoft desktop interfaces from DOS, a command line interface, to some guesses about Windows 7. **(It was recorded before the release).**

▪ System Utilities

Utility software is a kind of system software designed to help analyze, configure, optimize and maintain the computer. A single piece of utility software is usually called a utility or tool.

Utility software should be contrasted with application software, which allows users to do things like creating text documents, playing games, listening to music or surfing the web. Rather than providing these kinds of user-oriented or output-oriented functionality, utility software usually focuses on how the computer infrastructure **(including the computer hardware, operating system and application software and data storage)** operates. Due to this focus, some utilities are rather technical and targeted at people with an advanced level of computer knowledge. Others are commonly used by all users - a task manager is shown on the right.

Most utilities are highly specialized and designed to perform only a single task or a small range of tasks. However, there are also some utility suites that combine several features in one piece of software. Most major operating systems come with several pre-installed utilities. You may have to (find and) add utilities for your Smartphone, though. Although utilities are part of the system software, they are not part of the OS per se. They are loaded into memory as needed, either by the user or the OS.

▪ Malware

Malware, short for malicious software, consists of **programming (code, scripts, active content, and other software)** designed to damage or disable the system or data, disrupt or deny operation, gather information that leads to loss of privacy or exploitation, gain unauthorized access to system resources, and other abusive behavior. The expression is a general term used by computer professionals to mean a variety of forms of hostile, intrusive, or annoying software or program code.

Malware is not the same as defective software - software that has a legitimate purpose but contains harmful bugs or programming errors. Since the result may be the same, it should be clear that intent is considered important although this is a historically vague rule.

▪ What Is Its Purpose?

Categorizing malware by its purpose - the intent of the author - can be difficult or impossible. The following categories just give us a way to think about malware. Others might choose different categories, place a particular piece of malware in a different category, and some malware might reasonably fit into multiple categories.

- **Pranks**

Many early infectious programs, including a number of MS-DOS ***viruses***, were written as experiments or pranks. They were generally intended to be harmless or merely annoying, rather than to cause serious damage to computer systems. In some cases, the perpetrator did not realize how much harm his or her creations would do. Young programmers learning about viruses and their techniques wrote them simply for practice, or to see how far they could spread. As late as 1999, widespread viruses such as the Melissa virus appear to have been written chiefly as a prank. Many believe the infamous Morris Worm, that shut down a large part of the Internet in 1988 and started a new era in network security, was a prank or experiment gone wrong - others are not so sure it belongs in this category. In any case, none of these efforts had any **"legitimate purpose" for you, the user/manager/owner of the system "attacked".**

- **Intentionally Harmful**

Hostile intent related to vandalism can be found in programs designed to cause harm or data loss. Many DOS viruses, and the Windows Explore Zipworm, were designed to destroy files on a hard disk, or to corrupt the file system by writing invalid data to them. Network-born worms, such as the 2001 Code Redworm, fall into the same category. Sometimes a worm is designed to vandalize web pages, like the online equivalent to graffiti tagging, with the author's alias or affinity group appearing everywhere the worm goes. **(Note that the basic purpose of most web pages is to provide information, to communicate, so interfering with that goal is generally harmful.)**

- **Profit**

Since the rise of widespread broadband Internet access, some malicious software has been designed for a profit, like forced advertising, for example. **(Some forced advertising causes your browser to redirect you to an advertising page or displays pop-ups.)** Since 2003, the majority of widespread viruses and worms have been designed to take control of users' computers for black-market exploitation. Infected **"zombie computers"** are used to send email spam, to host contraband data such as child pornography, or to engage in distributed denial-of-service attacks as a form of extortion.

The latest versions seems to be claiming that you have a virus and **"offering"** you a solution or claiming that you have violated a law and need to send money **(by credit card)** to avoid prosecution**. (The latter is often related to pornography and usually occurs when you are visiting such a site, which hints at an avoidance strategy.)** In the worst case, your data is encrypted and held hostage until you pay (Ransom ware).

In each case, your best bet is to not click on anything on the web page or any pop-ups. Instead, start your task manager or system monitor and delete or kill the process. Sort the processes by name; find every instance of your browser application and "**force stop"** them all. If necessary, reboot your computer. If necessary use a "hard reboot" - hold the power button in for ten seconds. If none of those are successful, it's time to contact your IT department. The sidebar explaining process appeared in the last module.

- ## Indirect Profit

Another strictly for-profit category of malware has emerged in spyware - programs designed to monitor users' web browsing or other activity, display unsolicited advertisements, or redirect affiliate marketing revenues to the spyware creator. Spyware programs do not generally spread like viruses; they are, in general, installed by exploiting security holes or are packaged with user-installed software, such as peer-to-peer music sharing applications. Sometimes, the spyware is loaded unknowingly by the user in response to a web page pop-up. Probably the easiest step you can take to minimize attacks of all sorts is to use a **"non-privileged"** user account on a day-to-day basis.

- ## Hacktivism

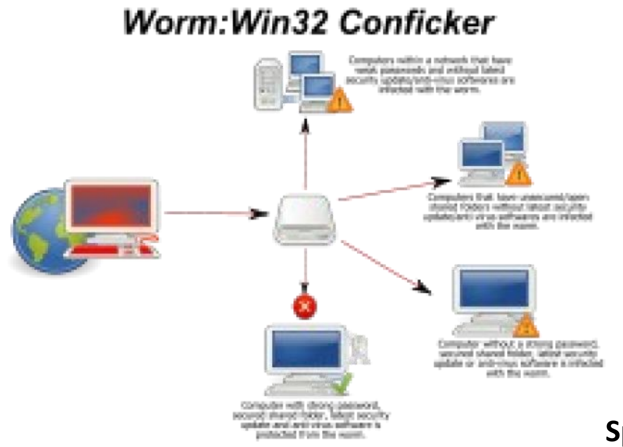
Hacktivism is the use of **(illegal)** hacking techniques for an activist cause. Hacktivism could be further defined as "the non-violent use of illegal or legally ambiguous digital tools in pursuit of political ends". These tools include web site defacements, redirects, denial-of-service attacks, information theft, virtual sit-ins, and virtual sabotage. Like any form of activism, doing something for **(what you consider to be)** a good cause does not make it legal or right.

▪ Viruses & Worms

The best-known types of **malware, viruses and worms**, are known for the manner in which they spread, rather than any other particular behavior. The term computer virus is used for a program that has infected some executable software and, when run, causes the virus to spread to other executable.

Viruses may also perform other actions, like creating a backdoor for later use, damaging files, or even damaging equipment. On the other hand, a worm is a program that actively transmits itself over a network to infect other computers. Worms may also take malicious actions.

These definitions lead to the observation that a virus requires user intervention to spread, whereas a worm spreads itself automatically. Using this distinction, infections transmitted by email or Microsoft Word documents, which rely on the recipient opening a file or email to infect the system, would be classified as viruses rather than worms.


Worm:Win32 Conficker
Computers within a network that have weak passwords and without latest security update/anti-virus softwares are infected with the worm.
Computers that have unsecured/open shared folders without latest security update/anti-virus softwares are infected with the worm.
Computer without a strong password, secured shared folder, latest security update or anti-virus software is infected with the worm.
Computer with strong password, secured shared folder, latest security update and anti-virus software is protected from the worm.

Spread of Conficker worm

- ## Trojans

In broad terms, a Trojan horse is any program that invites the user to run it, concealing a harmful or malicious payload. The payload may take effect immediately and can lead to many undesirable effects, such as deleting the user's files or further installing malicious or undesirable software. Trojan horses known as droppers are used to start off a worm outbreak, by **"injecting"** the worm into users' local networks.

One of the most common ways that spyware is distributed is as a Trojan horse, bundled with a piece of desirable software that the user downloads from the Internet. When the user installs the software, the spyware is installed alongside. Spyware authors who attempt to act in a legal fashion may include an end-user license agreement that states the behavior of the spyware in loose terms, which the users are unlikely to read or understand.

- **Rootkits**

Originally, a **rootkit** was a set of tools installed by a human attacker on a UNIX system, allowing the attacker to gain administrator **(root)** access. Today, the term rootkit is used more generally for concealment routines in a malicious program. Once a malicious program is installed on a system, it is essential that it stays concealed, to avoid detection and disinfection. The same is true when a human attacker breaks into a computer directly. Techniques known as rootkits allow this concealment, by modifying the host's operating system so that the malware is hidden from the user. Rootkits can prevent a malicious process from being visible in the system's list of processes, or keep its files from being read.

In an attempt to keep the user from stopping a malicious process, another is sometimes installed to monitor it. When the process is stopped (**killed**), another is immediately created. Modern malware starts a number of processes that monitor and restore one another as needed. In the event that a user running Microsoft Windows is infected with such malware **(if they wish to manually stop it),** they could use Task Manager's 'processes' tab to find the main process (the one that spawned the **"resurrector process (es)"),** and use the **'end process tree'** function, which would kill not only the main process, but the **"resurrector(s)"** as well, since they were started by the main process. Some malware programs use other techniques, such as naming the infected file similar to a legitimate or trustworthy file **(expl0rer.exe VS explorer.exe)** to avoid detection in the process list.

- ## Backdoors

A **backdoor** is a method of bypassing normal authentication procedures. Once a system has been compromised **(by one of the above methods, or in some other way),** one or more backdoors may be installed in order to allow easier access in the future. Backdoors may also be installed prior to malicious software, to allow attackers entry.

- ## Spyware

Spyware is a type of malicious software that can be installed on computers, and which collects small pieces of information about users without their knowledge. The presence of spyware is typically hidden from the user, and can be difficult to detect. Typically, spyware is secretly installed on the user's personal computer. While the term spyware suggests software that secretly monitors the user's computing, the functions of spyware extend well beyond simple monitoring. Spyware programs can collect various types of personal information, such as Internet surfing habits and sites that have been visited, but can also interfere with user control of the computer in other ways, such as installing additional software and redirecting Web browser activity.

Spyware is known to change computer settings, resulting in slow connection speeds, different home pages, and/or loss of Internet connection or functionality of other programs. In an attempt to increase the understanding of spyware, a more formal classification of its included software types is provided by the term privacy-invasive software.

Classification of code as spyware **(or sometimes browser cookies as "tracking" cookies)** can be controversial. Often the software is installed by the user knowing that some amount of monitoring will take place. Users generally agree to this activity to get free software and it is often associated with music and video sharing. Some such software allows the user to turn off the monitoring, assuming they are aware of it and can find instructions for disabling it. Anti-spyware is usually part of anti-virus programs; scan using at least two different AV packages. Spybot Search and Destroy is a good freeware program for looking for spyware.

▪ Loggers

Keystroke logging **(often called key logging)** is the action of tracking **(or logging)** the keys struck on a keyboard, typically in a covert manner so that the person using the keyboard is unaware that their actions are being monitored. There are numerous key logging methods, ranging from hardware and software-based approaches to electromagnetic and acoustic analysis.

Key logging is often used by law enforcement, parents, and jealous or suspicious spouses **(lovers).** The most common use, however, is in the workplace, where your employer is monitoring your use of the computer. Unfortunately, all of these activities are legal.

Hardware Key logger.

- **Adware**

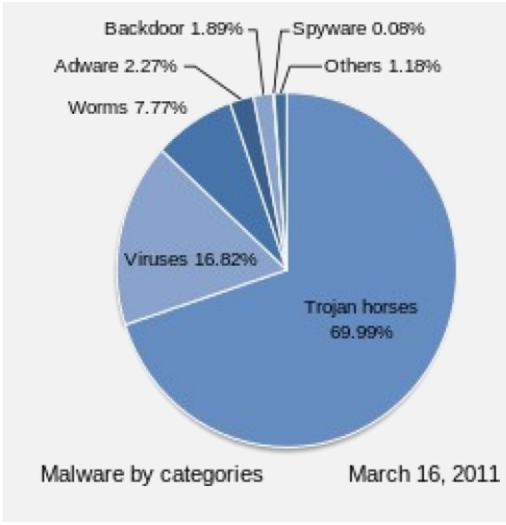
Adware, or advertising-supported software, is any software package which automatically plays, displays, or downloads advertisements to a computer. These advertisements can be in the form of a pop-up. The object of the Adware is to generate revenue for its author. Adware, by itself, is harmless; however, some adware may come with integrated spyware such as key loggers and other privacy-invasive software.

Advertising functions are integrated into or bundled with the software, which is often designed to note what Internet sites the user visits and to present advertising pertinent to the types of goods or services featured there - making it related to spyware. Adware is usually seen by the developer as a way to recover development costs, and in some cases it may allow the software to be provided to the user free of charge or at a reduced price. The income derived from presenting advertisements to the user may allow or motivate the developer to continue to develop, maintain and upgrade the software product. Conversely, the advertisements may be seen by the user as interruptions or annoyances or as distractions from the task at hand.

Some adware is also shareware, and so the word may be used as a term of distinction to differentiate between types of shareware software. What differentiates adware from other shareware is that it is primarily advertising-supported, like many free Smartphone apps. Users may also be given the option to pay for **a "registered" or "licensed"** copy to do away with the advertisements. Radio offers both a free version **(with ads)** and a paid subscription **(without ads).**

There is a group of software **(Alexa toolbar, Google toolbar, Eclipse data usage collector, etc.)** that sends data to a central server about which pages have been visited or which features of the software have been used. However they differ from "classic" malware because these tools only send data with the user's approval. The user may opt in to share the data in exchange for additional features and services or **(in case of Eclipse)** as a form of voluntary support for the project. Some security tools report such loggers as malware while others do not. The status of this group is questionable.

Some tools like PDF Creator are more on the boundary than others because opting out has been made more complex than it could be **(during the installation, the user needs to uncheck two check boxes rather than one).** However, PDF Creator is only sometimes mentioned as malware and is still subject of discussions.



Malware static's on 2011-03-16 (Panda Security)

- ## Security & Administration

It seems appropriate after discussing malware to talk about protecting your system and keeping it running well. In the workplace, the IT department will handle this - of course, in a small business, you may be the IT department. For hand-held devices, you generally don't have access to all the tools discussed here. However, much of this applies as much to Smartphone as to your desktop PC or laptop. For these devices, you are the system administrator.

What if you have a Mac? It may surprise those in the Windows world to hear it, but there is disagreement about whether it is worth it to worry about malware in the Mac world. In fact, most malware has been generated for the far more popular Windows world. And, the lack of control over applications for Windows has encouraged this **(and the many more applications than are available for Macs)**. Some also think there are inherently fewer bugs **(vulnerabilities)** to exploit in the Mac. However, currently there is a great deal of news being generated by malware written for the Mac. **(I believe Firewalls, anti-virus and anti-spyware should be used on all computer systems, including your Smartphone.)** As malware and general PC administration is the focus here, we will revisit some of these issues later for more specific topics, like networks and the Internet.

▪ Security Tools

Most PC systems include basic **security tools**. These can be replaced or supplemented, but should be used immediately. Connecting a PC to the Internet will lead to attacks very quickly, often within seconds. Using the tools below and following the suggestions will give you a reasonable chance of avoiding, catching or blocking, and removing most of the malware discussed in the last module.

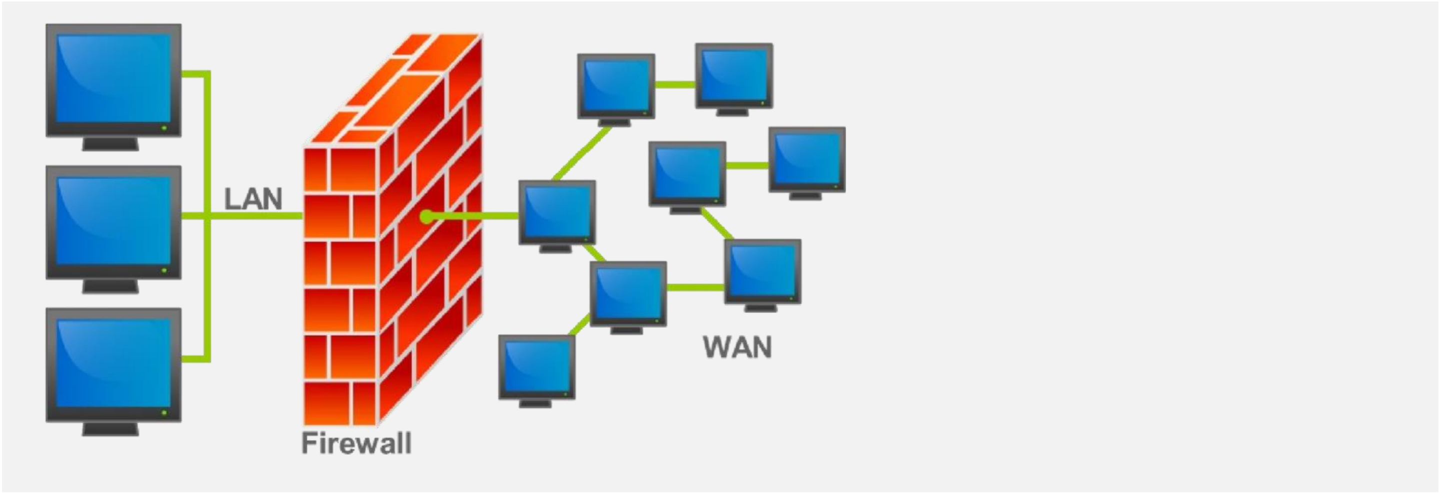
Obviously avoidance is best. Re-read the quick tips in the sidebar above.

- **Firewall**

A firewall is a device or set of devices designed to permit or deny network transmissions based upon a set of rules and is frequently used to protect networks from unauthorized access while permitting legitimate communications to pass. **Note that illegitimate outgoing communications can also be blocked. Many personal computer operating systems include software-based firewalls to protect against threats from the public Internet. Many routers that pass data between networks contain a firewall also.** (You probably have a wireless router in your home.)

Without proper configuration, a firewall can often become worthless. Standard security practices dictate a **"default-deny"** firewall ruleset, in which the only network connections which are allowed are the ones that have been explicitly allowed.

Unfortunately, such a configuration requires detailed understanding of the network applications and endpoints required for the organization's day-to-day operation. Many businesses lack such understanding, and therefore implement a **"default-allow"** ruleset, in which all traffic is allowed unless it has been specifically blocked. This configuration makes inadvertent network connections and system compromise much more likely.



▪ Antivirus

Antivirus (AV) or anti-virus software is used to prevent, detect, and remove malware, including but not limited to computer viruses**,** **computer worms, Trojan horses, spyware and adware.** Most AV programs provide *on-access* scans of any program that you run or download for install. Some extend this to your email as well.

A variety of strategies are typically employed. ***Signature****-based detection* involves searching for known patterns of data within executable code. However, it is possible for a computer to be infected with new malware for which no signature is yet known. To counter such so-called zero-day threats, heuristics can be used. One type of ***heuristic approach***, generic signatures, can identify new viruses or variants of existing viruses by looking for known malicious code, or slight variations of such code, in files. Some antivirus software can also predict what a file will do by running it in a sandbox and analyzing what it does to see if it performs any malicious actions.

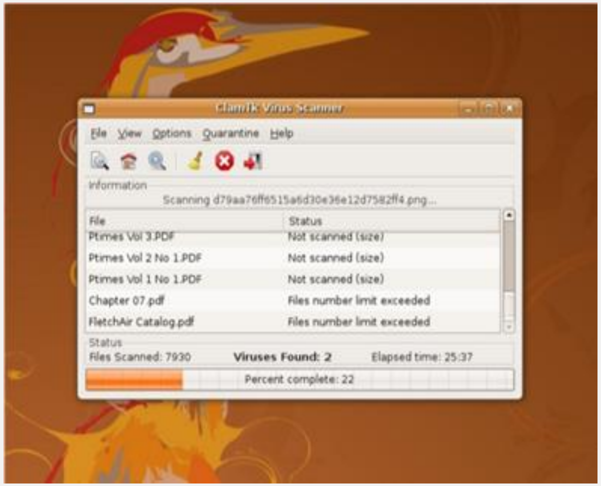
No matter how useful antivirus software can be, it can sometimes have drawbacks. Antivirus software can impair a computer's performance. **(This is why some Smartphone and Mac users choose not to run AV software.)** If the antivirus software employs heuristic detection, success depends on achieving the right balance between false positives and false negatives. **(A *false positive* means that useful, safe software will be flagged as dangerous.)** Finally, antivirus software generally runs at the highly trusted kernel level of the operating system, creating a potential avenue of attack.

- ## Anti-Spyware

Although most antivirus programs claim to detect and remove spyware and adware, they are sometimes not as good at this as software intended specifically for this task. This is at least partly due to the controversy over what exactly qualifies as spyware and adware **(discussed in the last module).** Many experts recommend that you use multiple software defenses, although the second or third are usually employed as manually run scans **(not on-access).**

▪ Specific Tools

There are a number of AV packages available **(Top ten review)** and they are sold **(or distributed for free)** in various suites, that is with differing components, for example, **Norton Antivirus with Antispyware, Norton Internet Security, and Norton 360 All-In-One Security, all by Symantec.** Comments here are general. You should refer to the software web site **(and Google it)** for more information.



Scanning d79aa76ff6515a6d30e36e12d7582ff4.png...

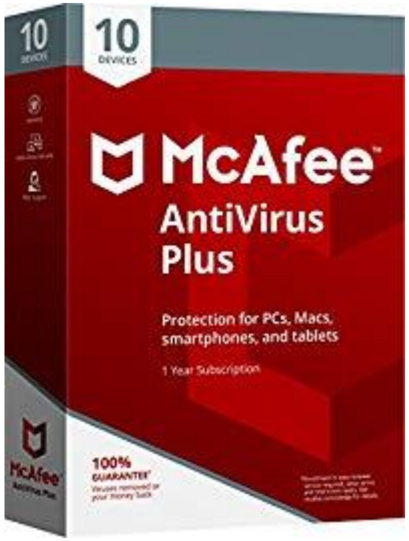
File	Status
Ptimes Vol 3.PDF	Not scanned (size)
Ptimes Vol 2 No 1.PDF	Not scanned (size)
Ptimes Vol 1 No 1.PDF	Not scanned (size)
Chapter 07.pdf	Files number limit exceeded
FletchAir Catalog.pdf	Files number limit exceeded

Files Scanned: 7930 **Viruses Found: 2** Elapsed time: 25:37

Percent complete: 22

An example of free antivirus software: ClamTk3.08

Norton and McAfee are well-known suites of protection tools. They have the advantage of a large company's quick reaction to new threats but the disadvantage of using a lot of system resources itself. MacAfee Enterprise Edition is for businesses.



At SRU, any student or employee can get a copy of this version for their laptop or home computer for free. **Webroot, Kaspersky, and Bit defender** are all rated well in the review linked above. Avg is a well-respected AV package. A free version is available for 30 days. Malware bytes provides a free version, although the free version does not provide on-access scans. At SRU, I have used this in conjunction with McAfee Enterprise successfully.

All these are types of an Antivirus.

▪ Protecting the system from ourselves

As Louis Binstock said, **"we are our own worst enemy"**. Not only do we delete the file we need, fail to keep a copy before editing, ignore updates, spill coffee on the system, or drop a laptop, we are also responsible for installing most of the programs that harm our computer systems. Sometimes this is explicitly downloading a program containing adware or spyware, other times it is because we are using an account that has administrator privileges. The former may be a choice, the latter certainly is, and a poor one at that.

In the early days of personal computing, a PC didn't have accounts. There was a single user that both used and administered the machine. This has evolved to systems that allow the creation of user accounts with various privileges, that are protected from one another, and that have a separate administrative account. All of these steps match what was done on multi-user operating systems used in IT departments **(and as servers)** many years before. Unfortunately, most PC users simply use the administrator account. Create and use a **"normal"** account. Only use administrator privileges when needed.

▪ Backup And Restore

It is a fair assumption that everyone has lost a file they needed. Often it is not lost to malware, but either hardware failure or our own failure to backup files. There are tools that will assist you in making backups, or you can manually copy your data to another drive. The advantage to manual backup is that it is fast. The disadvantage is that it is not scheduled. It is not automatic. Whatever the cause of the loss may be, it represents time and effort. Make a backup plan and stick to it.

There are online services for backing up your files. These services run a program that is scheduled to automatically backup your files **(while you sleep)**. They have an additional advantage of offering off-site backup. A second hard drive in your computer is also vulnerable to malware and an external drive, even when not connected, is susceptible to theft or fire. The biggest disadvantage of such a service is that you are trusting a third party with your data**: work, pictures, saved email, etc.** Make sure they encrypt your data to stop casual perusal by employees and theft by others. But, remember that they may have the encryption key. Restore points are a type of backup for the system itself. Create a restore point before installing software, partitioning a drive, etc.

▪ Authentication

Authentication refers to the process of making sure a user has access to the resource, usually the entire system; that is, logging on. If you are using accounts, as recommended above, you can also protect your files from other users. They can be given permission to change **(delete)**, read, or denied access. When you attempt to access a resource you do not have permission for, or install a program, you are generally prompted to give permission **(and authenticated).** You may have to log in as an administrator to accomplish some tasks - then switch back to your user account.

Standard authentication is accomplished with a password. Password that is hard to guess: 8-12 (or more) characters and including non-alphabetical characters that you can remember. Random would be great, if you have that kind of memory. Otherwise consider modifying a phrase that is meaningful to you **(ignore any spaces)**. For example, **your favorite dog is named Princess (and she has paws):**

- PrinceSaysPause - WEAK, uses caps, but uses all dictionary words
- PrintSezPauzz - better, but all letters
- Pr1n5essP0z - good, three digits (the 1 is read as an I, 5 is an S, and 0 an O)
- Pr1nse55-P0z - very good, 4 digits and hyphen
- Pr1nse55-P0z@@ - excellent, 14 characters long with digits, caps and "special" chars (Are those last two her paws or eyes?)

Don't forget that Princess "barqs", "jumpz", and "cud.del5" also. It's kind of endless, which is the point. If you choose to substitute digits or punctuation for letters like above (@b0v3), make sure you also "mis-spell" the word **(phonetically).** You can also mix and match the above with simple substitution ciphers (A becomes B, B becomes C, etc.), so "Princess paws" becomes **"pr1nse55-qbxt".**

You also need to **manage your passwords**. Change them regularly; once a semester is good rule for college students. Chances are someone else knows one of them. When your relationship ends, change your passwords - it's like changing the locks, only easier. **(Do you really want them reading your email?)**

Don't forget that many passwords are saved on PCs, laptops and Smartphone, including banking or online trading passwords. Secure the device first. Your Smartphone almost certainly has an authentication mechanism. Use it! It also keeps rude friends and significant others from perusing your texts, emails, and photos. Change all passwords if a device is lost or stolen.

Shoulder surfing refers to obtaining someone's password **(or PIN)** through direct observation. You should not feel uncomfortable asking someone to **"excuse you"** while you type a password - and wait for them to comply.

- ### Encryption

Assume your PC **(and online backup service)** will eventually get hacked. Why not save your files in a way that prevents others from accessing them. Encryption packages are available **(some are free)** that will create a "container" file that appears like another disk drive to your system. Anything saved on that drive is actually saved in the encrypted container**. (You do have to start this up before using it and make sure you close it down when done.)** This is a must if you are using business files at home, working with healthcare information, or student grades. You can purchase flash drives that will protect your data also. They have the encryption software on them. **(Kingston has been a leader.)**

Note that if you encrypt a file or create an **"encrypted volume"** - a big file that acts like a separate encrypted disk - you can back that up on an encrypted cloud service. It will be encrypted twice and be that much harder to hack.

▪ Speeding Up Your PC

There are a number of things you can do to speed up your PC, but the most effective is likely to be removing spyware and adware that you installed yourself. In addition, you will find that there are a number of programs that are running all the time, so they can **"start"** quickly when you select them. **Scan for viruses, spyware and adware first.** Remove whatever you can - some of the spyware and adware might be there because you want the program it came with. Most of the rest of this is for Windows, but I put in some links for OS X.

Uninstall programs you no longer use with the uninstall utility **(Windows-Control_Panel:_Add_or_remove_programs)**. For the Mac, some files can simply be dragged to the Trash, but "bundles" are more complicated: search for Uninstalling Applications in Mac OS Xon the web.

There are some system level things you can do, as well. **Empty the trash bin** to get rid of files you no longer need and run the disk clean up utility**(Windows_Start:All_Programs:Accessories:System_Tools)** to **get rid of temporary internet files** and other junk. Simply creating space on the hard disk will make it easier to place needed files in one contiguous location on the disk to speed up access. (No speed boost if you have an SSD drive, though.)

After cleaning up old files create a restore point and then **defragment the hard disk** (System Tools). This will optimize the location of files, again speeding up access. The more space you have available, the better this will work, so clean up first. (Not necessary with SSD drives and Apple says don't bother defragging in OS X - in support article 25668.)

Finally, if you access the Windows Systems Properties **(Control_Panel:_System)**, under the Advanced tab, you will find System Performance. Select Performance Settings. Generally, Windows ships with "**adjust for best appearance"** selected. You can turn off features you don't care about, or all, by selecting **"adjust for best performance”.** The most common, and easiest, hardware solution is to add RAM.

▪ Setting Up A New PC

If you've purchased a new system and set it up physically, there are still some things you should do before you start browsing the web. Primarily, you need to update all the software on your **"new"** system, as it may actually be months old, and install antivirus software **(if it didn't come pre-installed).**

This is also a great time to write down all the processes that show up in the Task Manager for later reference. You can do this on an old system after you get rid of unwanted programs also. This is great for later comparisons to find malware. You probably also have an old system to get rid of. Recycle! And make sure you ***shred*** **(wipe)** the disk to protect your privacy.

- **Over Heating**

You can help prevent over heating by keeping the inside of the (desktop) PC clean & organized.

- **Device Driver**

In computing, a device driver is a computer program that operates or controls a particular type of device that is attached to a computer. A driver provides a software interface to hardware devices, enabling operating systems and other computer programs to access hardware functions without needing to know precise details about the hardware being used.

A driver communicates with the device through the computer bus or communications subsystem to which the hardware connects. When a calling program invokes a routine in the driver, the driver issues commands to the device. Once the device sends data back to the driver, the driver may invoke routines in the original calling program. Drivers are hardware dependent and operating-system-specific. They usually provide the interrupt handling required for any necessary asynchronous time-dependent hardware interface.

- **Purpose**

The main purpose of device drivers is to provide abstraction by acting as a translator between a hardware device and the applications or operating systems that use it. Programmers can write higher-level application code independently of whatever specific hardware the end-user is using.

For example, a high-level application for interacting with a serial port may simply have two functions for **"send data"** and **"receive data".** At a lower level, a device driver implementing these functions would communicate to the particular serial port controller installed on a user's computer. The commands needed to control a **16550 UART** are much different from the commands needed to control an FTDI serial port converter, but each hardware-specific device driver abstracts these details into the same **(or similar)** software interface.

- **Development**

Writing a device driver requires an in-depth understanding of how the hardware and the software works for a given platform function. Because drivers require low-level access to hardware functions in order to operate, drivers typically operate in a highly privileged environment and can cause system operational issues if something goes wrong. In contrast, most user-level software on modern operating systems can be stopped without greatly affecting the rest of the system. Even drivers executing in user mode can crash a system if the device is erroneously programmed. These factors make it more difficult and dangerous to diagnose problems.

The task of writing drivers thus usually falls to software engineers or computer engineers who work for hardware-development companies. This is because they have better information than most outsiders about the design of their hardware. Moreover, it was traditionally considered in the hardware manufacturer's interest to guarantee that their clients can use their hardware in an optimum way. Typically, the **Logical Device Driver (LDD)** is written by the operating system vendor, while the **Physical Device Driver (PDD)** is implemented by the device vendor. But in recent years non-vendors have written numerous device drivers, mainly for use with free and open source operating systems. In such cases, it is important that the hardware manufacturer provides information on how the device communicates. Although this information can instead be learned by reverse engineering, this is much more difficult with hardware than it is with software.

Microsoft has attempted to reduce system instability due to poorly written device drivers by creating a new framework for driver development, called **Windows Driver Foundation (WDF)**. This includes **User-Mode Driver Framework (UMDF)** that encourages development of certain types of drivers—primarily those that implement a message-based protocol for communicating with their devices—as user-mode drivers. If such drivers malfunction, they do not cause system instability. The **Kernel-Mode Driver Framework (KMDF)** model continues to allow development of kernel-mode device drivers, but attempts to provide standard implementations of functions that are known to cause problems, including cancellation of I/O operations, power management, and plug and play device support.

Apple has an open-source framework for developing drivers on macOS called the I/O Kit.

In Linux environments, programmers can build device drivers as parts of the kernel, separately as loadable modules, or as user-mode drivers **(for certain types of devices where kernel interfaces exist, such as for USB devices). Makedev includes a list of the devices in Linux: ttyS (terminal), lp (parallel port), HD (disk), loop, sound (these include mixer, sequencer, dsp, and audio).**

The Microsoft Windows .sys files and Linux .ko modules contain loadable device drivers. The advantage of loadable device drivers is that they can be loaded only when necessary and then unloaded, thus saving kernel memory.

Kernel Mode Vs. User Mode

Device drivers, particularly on modern Microsoft Windows platforms, can run in kernel-mode **(Ring 0 on x86 CPUs) or in user-mode (Ring 3 on x86 CPUs)**. The primary benefit of running a driver in user mode is improved stability, since a poorly written user mode device driver cannot crash the system by overwriting kernel memory.

On the other hand, user/kernel-mode transitions usually impose a considerable performance overhead, thereby prohibiting user-mode drivers for low latency and high throughput requirements.

Kernel space can be accessed by user module only through the use of system calls. End user programs like the UNIX shell or other GUI-based applications are part of the user space. These applications interact with hardware through kernel supported functions.

Applications

Because of the diversity of modern hardware and operating systems, drivers operate in many different environments. Drivers may interface with:

- Printers
- Video adapters
- Network cards
- Sound cards
- Local buses of various sorts—in particular, for bus mastering on modern systems
- Low-bandwidth I/O buses of various sorts (for pointing devices such as mice, keyboards, USB, etc.)
- Computer storage devices such as hard disk, CD-ROM, and floppy disk buses (ATA, SATA, SCSI)
- Implementing support for different file systems
- Image scanners
- Digital cameras

Common levels of abstraction for device drivers include:

- **For hardware:**

- Interfacing directly
- Writing to or reading from a device control register
- Using some higher-level interface (e.g. Video BIOS)
- Using another lower-level device driver (e.g. file system drivers using disk drivers)
- Simulating work with hardware, while doing something entirely different

- **For software:**

- Allowing the operating system direct access to hardware resources
- Implementing only primitives
- Implementing an interface for non-driver software (e.g. TWAIN)
- Implementing a language, sometimes quite high-level (e.g. PostScript)
- So choosing and installing the correct device drivers for given hardware is often a key component of computer system configuration.

- **Virtual Device Drivers**

Virtual device drivers represent a particular variant of device drivers. They are used to emulate a hardware device, particularly in virtualization environments, for example when a DOS program is run on a Microsoft Windows computer or when a guest operating system is run on, for example, **a Xen host**. Instead of enabling the guest operating system to dialog with hardware, virtual device drivers take the opposite role and emulates a piece of hardware, so that the guest operating system and its drivers running inside a virtual machine can have the illusion of accessing real hardware. Attempts by the guest operating system to access the hardware are routed to the virtual device driver in the host operating system as e.g., function calls. The virtual device driver can also send simulated processor-level events like interrupts into the virtual machine.

Virtual devices may also operate in a non-virtualized environment. For example, a virtual network adapter is used with a virtual private network, while a virtual disk device is used with **iSCSI**. A good example for virtual device drivers can be Daemon Tools. There are several variants of virtual device drivers, such as **VxDs, VLMs, and VDDs.**

Open Drivers

- Printers: CUPS
- RAIDs: CCISS (Compaq Command Interface for SCSI-3 Support)
- Scanners: SANE
- Video: Vidix, Direct Rendering Infrastructure

- Solaris descriptions of commonly used device drivers:
- fas: Fast/wide SCSI controller
- hme: Fast (10/100 Mbit/s) Ethernet
- isp: Differential SCSI controllers and the SunSwift card
- glm: (Gigabaud Link Module) UltraSCSI controllers
- scsi: Small Computer Serial Interface (SCSI) devices
- sf: soc+ or social Fiber Channel Arbitrated Loop (FCAL)
- soc: SPARC Storage Array (SSA) controllers and the control device
- Social: Serial optical controllers for FCAL (soc+)

APIs

- Windows Display Driver Model (WDDM) – the graphic display driver architecture for Windows Vista, Windows 7, Windows 8, and Windows 10.
- Unified Audio Model (UAM)
- Windows Driver Foundation (WDF)
- Windows Driver Model (WDM)
- Network Driver Interface Specification (NDIS) – a standard network card driver API
- Advanced Linux Sound Architecture (ALSA) – as of 2009 the standard Linux sound-driver interface
- Scanner Access Now Easy (SANE) – a public-domain interface to raster-image scanner-hardware
- I/O Kit – an open-source framework from Apple for developing macOS device drivers
- Installable File System (IFS) – a filesystem API for IBM OS/2 and Microsoft Windows NT

- Open Data-Link Interface (ODI) – a network card API similar to NDIS
- Uniform Driver Interface (UDI) – a cross-platform driver interface project
- Dynax Driver Framework (dxd) – C++ open source cross-platform driver framework for KMDF and IOKit

Identifiers

A device on the PCI bus or USB is identified by two IDs which consist of 4 hexadecimal numbers each. The vendor ID identifies the vendor of the device. The device ID identifies a specific device from that manufacturer/vendor. A PCI device has often an ID pair for the main chip of the device, and also a subsystem ID pair which identifies the vendor, which may be different from the chip manufacturer.

See also

- Class driver
- Controller (computing)
- Device driver synthesis and verification
- Driver wrapper
- Free software
- Firmware
- Interrupt

- Loadable kernel module
- Makedev
- Open-source hardware
- Printer driver
- Replicant (operating system)
- Udev

Microsoft Office

Definition - What does Microsoft Office mean?

Microsoft Office is a suite of desktop productivity applications that is designed specifically to be used for office or business use. It is a proprietary product of Microsoft Corporation and was first released in 1990. Microsoft Office is available in 35 different languages and is supported by Windows, Mac and most Linux variants. It mainly consists of Word, Excel, PowerPoint, Access, OneNote, Outlook and Publisher applications.

Techopedia explains Microsoft Office Microsoft Office was primarily created to automate the manual office work with a collection of purpose-built applications.

Each of the applications in Microsoft Office serves as specific knowledge or office domain such as:

- Microsoft Word: Helps users in creating text documents.
- Microsoft Excel: Creates simple to complex data/numerical spreadsheets.
- Microsoft PowerPoint: Stand-alone application for creating professional multimedia presentations.

- Microsoft Access: Database management application.
- Microsoft Publisher: Introductory application for creating and publishing marketing materials.
- Microsoft OneNote: Alternate to a paper notebook, it enables a user to neatly organize their notes. Besides desktop applications, Microsoft Office is available to use online or from cloud under a lighter (Office Web Apps) and full (Office 365) version. As of 2013, Microsoft Office 2013 is the latest version, available in 4 different variants including Office Home Student 2013, Office Home Business 2013 and Office Professional 2 and the online/cloud Office 365 Home Premium.