|  |  |
| --- | --- |
| What is an Operating System? | * Manages the computer’s hardware * Intermediary between the computer user and the computer hardware * Basis for application programs to interface with |
| Where are computers used? | Diagram  Description automatically generated |
| Operating system Purposes | * Mainframe -designed primarily to optimize utilization of hardware * PC – support complex games, business applications, everything in-between * Mobile – provide an environment for easy user interface |
| What does an Operating System do? | * A computer system can be divided into four components   + Hardware   + Operating system   + Application programs   + Users |
| System view | * Operating System can be viewed as a resource allocator:   + CPU time   + Memory space   + File-storage space   + I/O devices * Operating System can be viewed as a control program:   + Manage execution of user programs to prevent errors   + “” to prevent improper use   + Concerned with the operation and control of I/O devices |
| Defining an Operating System | * No clear definition * Kernel ‘program’ – YES * System ‘programs’ – YES * Application programs -NO * In 1998, the US States Department of Justice filed a lawsuit against Microsoft for what?   + Internet explorer was too integrated into the system; trying to gather data   Diagram  Description automatically generated |
| Computer Startup | * Initial program, bootstrap rgoram loads   + Stored in firmware     - ROM (read-only memory)     - EEPROM (Electrically erasable programmable read-only memory)   + Initializes all aspects of the system   + Load the operating system kernel into memory |
| Computer Startup | * System processes/daemons   + Programs loaded into memory at boot time that runs the entire time the kernel is running * Interrupt   + Signals an event that has occurred   + Hardware may trigger them at any time, usually by the way of a system bus   + Software may trigger them by using a ‘system call’ (can also be referred to as a monitor call)   + CPU transfers execution to a fixed location (starting address for a service routine for that interrupt) |
| Main Memory | * Ideal for all programs to reside in memory permanently * This is not possible   + Main memory is too small to store all needed programs   + Main memory is ‘volatile’, which means it loses its contents when power is turned off, or otherwise lost |
| Secondary Storage | * Extension of main memory * Hold large quantities of data permanently * Hard Disk Drive (HDD) common secondary-storage device   C:\Users\as668\Desktop\1_04.jpg   * Solid-state disks – faster than hard disks, nonvolatile * NVRAM – nonvolatile storage. DRAM with battery backup power |
| Computer Overview | 1 |
| Multiprocessor Systems | * System with 2 or more processors * Shared computer bus, clock, memory, peripheral devices   + Varying shared components, not always consistent * Initially only available in servers, but have made their way to desktops, laptops, and mobile devices |
| Multiprocessor Systems | * Increased throughput * Economy of scale   + Shared multi CPU access to a single storage device as opposed to multiple CPUs individually housed and multiple storage devices * Increased reliability * Asymmetric multiprocessing   + One boss, assigns work to the worker processors * Symmetric multiprocessing (SMPD) |
| Multicore Systems | Multicore – multiple computing cores on a single chip  1 |
| Multiprogramming | * Job pool   + Resides on disk   + Small number of jobs reside in memory   + CPU chooses a job and begins to work on it * What “real-world” scenarios does this mimic?   + Printers, contractors, mail delivery, prioritizing |
| Dual-Mode Operation | * Why?   + Provides system protection   + Prevent user processes from:   + Infinite loops   + “Hogging” resources   + Affecting other users   Diagram  Description automatically generated |
| Timer | * Set to interrupt the computer after a fixed period of time |

|  |
| --- |
| **Operating System Services** |
| * User Interface   + Command-line interface – text commands via keyboard or typing in commands   + Batch interface – commands/directives entered into files, and the files are executed   + Graphical user interface (GUI) - window system with pointing /touch device * Program Execution   + Load the program into memory and run the program   + Program ends normally or abnormally (indicating an error) * I/O operations * File-system manipulation   + CRUD   + Read/write/create/delete files   + Provide permissions management to allow/deny access * Communications   + Process needs to exchange information with another process   + Processes could be on the same computer, or different computers   + Implemented via shared memory     - Two processes read/write to shared section of memory   + Implemented via message passing     - Packets with predefined formats are moved between processes * Error detection   + Memory error, power failure, parity error on disk, network connection failure, printer out of paper, illegal memory location, too much CPU time used * Resource allocation   + CPU cycles, main memory, file storage * Accounting – what resources and how long are users using them for -> billing purposes, security, optimization/efficiency * System calls -> windows – windows API; Unix, Linux, Mac OSX – POSIX API; Java Virtual Machine – Java API |
| **Policy vs Mechanism**   * Mechanisms – how to do something * Policies – what will be done * Policies are flexible * Example: timer construct is a mechanism for ensuring CPU protection. Deciding how long the timer is set is a policy decision. |
| **Modules**   * Loadable kernel modules – modules that can be loaded at boot time OR during run time * Saves memory by only loading kernel modules that are needed (e.g., printing) |
| **Operating-System Debugging**   * Failure Analysis |
| **Strace vs Dtrace** |