CSC 139 Chap 2

2.1 OS System Services

Userinterface - typically a gui with windows & keyboard & mice for input. Sometimes uses CLI - comand Line Interface

Program execution - must Load programs into memory & run the program, Also end program

IO Operations - The OS manages IO

File System manipulation - The OS has to allow programs to read and write Files & directories & create and delete files by name by searching, Some Os include permissions based on File ownership

Communication - An os implements shared memory where 2 or more process read & write to a shared section of memory. message passing - packets of info are moved between process

Error dectection - fix errors & also dectect them. Terminates error causing process, return error code or hault system

Resource Allocation - The Os is in charge of deciding who gets resources, cpu cycles, main memory, File storage

Logging - keep track of programs & record what computer resources they use

Protection & Security - Os has to make sure that access to system resources are controlle

Also makes user Authenticate themselves before granting access, makes sure io devices dont make invalid access atempts

2.2 User & OS interface

CLI - command Line Interface or Command interpreter user enters commands to be preformed by Os

Shells - when a system has multiple command interpreters, include Cshell, Bash, Korn

2 ways for Command interpreter to do something. 1 the command interpreter has the code to execute the command, 2 doesnt understand command, just used to find a file and executed. Example rm file.txt Looks for a file named rm then loads into memory and excutes with parameter File.txt

Gui - First one Xerox Alto computer 1973

2.3 System call - provides an interface to the services made avilable by an os asking the OS to do something

Api - specifies a set of functions avilable to an application programmer

Why use Api? will run on a system that supports said Api, also using system calls is harder

CSC 139 Chap2 Continued

Run time envirement - All the software needed to execute app written in a given programming Language. Compiler/interpreter, Libraries, Loaders

Compiled vs Interpreted - C, C++, are compiled to machine code then ran. Interpreted - an interpreter translate each line of code line by line into machine code then excutes. Python, Javascript, Ruby

System Call interface - the Link to system calls made avilable by the OS. Intercepts calls from the Api and calls the correct system call in the os

Types of System Calls

Process Control - create/end process, Load/excute. get/set process attributes wait/signal event

File management - create/delete files, open/close, read/write/reposition, get/set file attributes

Device management - request/release device, read/write/reposition, get/set device attributes, Logically attach or detach devices

Information maintenance - get/set time or date, get/set system data get/set process file or device attributes

Communications - create/delete communication connection, send/recieve messages transfer status info, attach or detach remote devices

Protection - get/set file premission

Boot Loader - Locates a peice of software then loads into memory & excutes

Process Communication - message passing model & shared-memory model share info by reading & writing data in shared memory

2.4 System Services | System Services - AKA system Utilities provide a convienient envirement for program development & execution, Some are user interfaces to system calls

registry - used to store & retrieve config info

Deamons/Services/Subsytems - Constantly running system-programs process

2.5 Linkers & Loaders | relocatable objectfile files designed to be laded into physical memory location

Linker - combines relocatable objectfiles into a single binary executable file

Loader - Loads the binary executable file into memory where it can run on a CPU core

relocation - assigns final addresses to program parts & adjusts code & data in the program

CSC 139 Chap 2 Continued

ELF - Executable & Linkable Format

entry point - the location of the first instruction when program executes

2.6 Why Apps are OS specific

An app can run on multiple Os in 3 ways

1. The app can be writen in an iterpreted language. The interpreter can be used by many Os

2. The app can be written in a language with a virtual machine that runs app
The virtual machine is part of the Run time environment

3. The app developers can use a standard language or Api where the compiler generates binaries into machine & os specific language

ABI - Application Binary interface - used to define how different components of binary code can interface for a given Os on a given architecture An ABI is the architecture equivalent to an Api

2.7 OS system Design & Implementation

Mechanism / Policy - Mechanism How something will be done, Policy what will be done

2.8 OS System Structure

monolithic - place all Kernel functionality into a sigle static binary file, runs single address space

Kernel - provides the file system, CPU scheduling & memory management... through system calls

tightly coupled - would be the monolithic, one change can effect whole system

Loosley coupled - divided into seperate, smaller components. Changes to one thing might not break everything

Layered System - Layer 0 is hardware highest layer is user interface

micro kernel - removes all nonessential components from kernel & implements them as user level programs that reside on seperate address space

modules or Loadable kernel modules - kernel has a set of core components & can link additional services over modules

Kernel enviroment - Darwin provides 2 system call interfaces, Mach systems known traps & BSD system calls. Darwin combines Mach, BSD, the IO Kit and any kernel extension into a single address space to address performance problems

29 Building & Booting an OS

Booting - Starting a computer by loading the kernel

CSC 139 2/1/21

How a system boots - bootstrap or boot loader locates the Kernel
The kernel is loaded into memory & started. The kernel initializes
hardware. The root file system is mounted

2.10 Core dump - capture of memory of process
crash - failure in kernel
Counters - Keep track of system activity, system calls counter
tracing - collects data for a specific event
BCC - BPF Compiler Collection - toolkit that provides tracing features for Linux