International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 6 | ISSN : 2456-3307

Kernel - The Heart an Operating System

Neha Bansal, Simran Saini, Kiranpreet Kaur, Charanjeet Kaur Raina
Department of Computer Science Engineering, Adesh Institute of Technology, Gharuan, Chandigarh Campus, Mohali, Punjab, India

ABSTRACT

Kernel - the core part of the OS- have drivers to talk to the Hardware, Memory manager, schedule manager, interrupt handler, CPU scheduling, etc which BIOS Doesn't has. It can also detect the hardware present on system and talk to them(hardware) through the device drivers. It is responsible for multiple applications to share the same hardware by controlling the access to CPU, Memory, and Devices by avoiding the deadlock condition.
Keywords : Kernel, functions of kernel, type of kernel, LINUX kernel, OS X kernel, Window NT kernel.

I. INTRODUCTION

When software needs attention of hardware or wants hardware to do anything, there is need of KERNEL. Kernel is the small program that is loaded into memory after POST(Power on Self Test).The original kernel code was developed by Linux Trovalds and written in C Language.

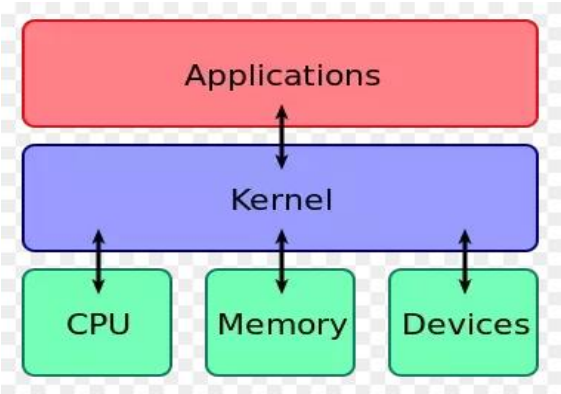
The size of kernel should be small as possible, as lie in main memory, also, provide all the essential services require by operating system and applications.But there are some limitations on the information that kernel gets about the hardware, this is why it uses the information provided by BIOS to work better. BIOS (Basic Input/output System) - a program that is reside on the ROM which is embedded on motherboard. Its function is to check if all the hardware is properly installed and working or not and then initialize the firmware of the Hardware.

II. FUNCTIONS OF KERNEL

Kernel is the interface between applications and actual data processing.Kernel have functionality to access computer resources that includes

- ✓ **CENTRAL PROCESSING UNIT** responsible for processing or executing all the programs. Kernel takes the responsibility of deciding to allocate the processor to which running program.
- ✓ **RANDOM ACCESS MEMORY** used to store both the programs as well as data. When more than one program try to access the memory, sometime more than available memory, then kernel decide how to manage all. It decide what memory a process can use, also decide what to do if not available memory.
- ✓ **INOUT/OUTPUT DEVICES**if any application needs interaction of hardware, kernel provides proper method for use that device. Kernel also provides a convenient way for Inter Process Communication (IPC).
- ✓ **MEMORY MANAGEMENT** kernel use virtual addressing. It has a full right to access system's memory and give the memory to process when required. Virtual addressing make the virtual address of any physical address using paging and segmentation
- ✓ **SYSTEM CALLS** a mechanism for requesting a service from OS. Kernel permit the access for that service.
- ✓ **SECURITY MAANGEMENT** provides security from faults and from errors in process

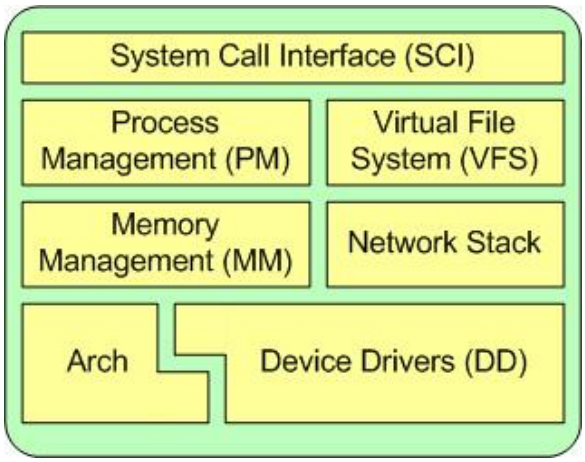
III. TYPES OF KERNEL



1. **MICROKERENLS** take of approach only managing include only CPU, memory and IPC. It can handle all in user mode which doesn't have high permission as supervision mode and hence provide **high security**. By changing the operating system, there is no need to worry as it is portable. It use small footprints for both installations and memory. Hardware may get slow ad it is in user mode and react slow. Processes have to wait in queue to get information and can't access to other process with waiting.
2. **MONOLITHIC KERNELS** opposite to micro kernels, monolithic kernels hold not only CPU, memory and IPC but also this include device drivers, system server calls, file system management. It is better at accessing hardware and performing multitasking because if any process need to get information or to access any running process, there is no need to wait in queue as there is more direct line to access. Since it works in supervisor mode(**less secure**), it may bring own your PC if one of them doesn't work properly. It use large footprints for both installations and memory.
3. **HYBRID KERNELS** has ability to pick and choose whether want to run in user mode or run in supervisor mode. It has small footprint than monolithic kernels

IV. LINUX KERNEL

Linux kernel is monolithic kernel because it carry all the basic services to kernel. It is efficient in terms of memory, CPU usage and extremely stable. It can be compiled to run on huge processors and platform with different constraints.



Architecture of LINUX KERNAL

System Call Interface(SCI) perform the function call from user space to kernel space.SCI implementation is found in ./linux/kernel

Process Management (PM) is executing the processes. Kernel provides API(Application program Interface) through SCI to create new process. We can find process management sources in ./linux/kernel

Memory management(MM) if memory required by multiple process at same time is more than available memory then it use the concept of swapping or provide the virtual addresses. Memory management sources are in ./linux/mm.

Virtual file system(VFS) provides switching layer between SCI and the file system support by kernel. Anything in Linuxis considered as file, we can find the file system sources in ./linux/fs.

Network Stack process incoming packet i.e. asynchronous event. It collects, identifies and dispatches before a process take care of it and thus diliver the packets to program and network stack. All the routing of addresses are done in kernel. Network sources in kernel are found in ./linux/net.

Device Driver make the hardware device usable. The code written for particular device to make is responsive is called device driver. It is found in ./linux/drivers.

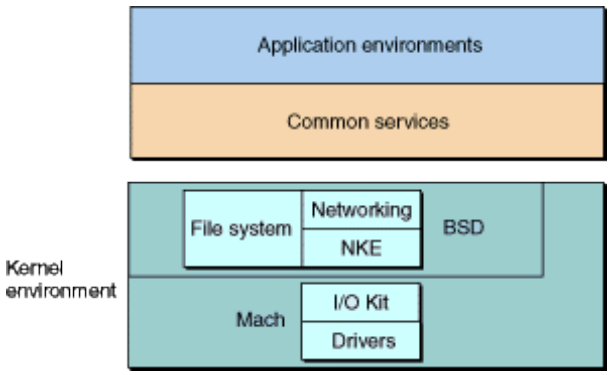
V. OS X KERNEL

OS X is the version 10 of Apple Macintosh Operating system describe as its First Complete version of OS. Enhance features of OS X includes:

1. Improved Reliability and Performance
2. enhanced networking features
3. an object-based system programming interface
4. increased support for industry standards

OS X kernel includes:

1. Mach microkernel(I/O kit, Derivers)
2. BSD(File System, Networking)



Architecture of OS X Kernel

Mach Microkernel: It manages processor resources like CPU usage, memory management, handle scheduling, memory protection, messaging centered infrastructure to rest of OS layers.

ALLOCATING MEMORY IN KERNEL:

<libkern/OSMalloc.h> define the header file for allocating memory from i/o kit(a framework for creating device drivers in OS X).

OSMalloc defines allocating a block of memory.

OSMallock_noblock defines allocating a block of memory, but immediately returns NULL if request would block.

OSMallock_tagalloc creates a unique tag for memory allocation.

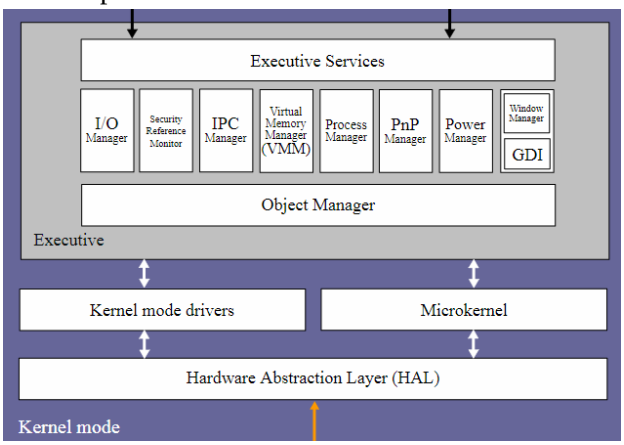
BSD Berkeley Software Distribution layer above Mach layer provides OS API's and services. It is based on FreeBSD(an advanced computer operating system used to power modern servers, desktops, and embedded platforms). It supports:

File system default file system HFS+. Advance feature is Virtual File System of layered architecture.

Networking takes the advantage of BSD's advance networking capabilities to support NAT and firewalls. It supports multihoming (a mechanism used to configure one computer with more than one network interface and multiple IP addresses).

VI. WINDOW NT KERNEL

It has full access to hardware and system resources of computer. It runs the code in protected memory area. Kernel mode stops the user mode to access some critical area of operating system to access. User mode processes have to take permission of kernel mode to perform operations on their behalf.



Architecture of Window NT kernel

Executive services make low level kernel mode portion. It lies in file **ntoskrnl.exe**. It deals with Input/output manager, process and security management. These divided system into subsystems like cache manager, object manager, configuration manager, Local Procedure Call (LPC), Process Structure, memory manager and Security Reference Monitor (SRM).

Kernel is lie between executive and Hardware Abstraction Layer (HAL). It provides synchronization, threading for multiple processes, scheduling, interrupt handling. Also, it is responsible for initialization of device drivers at booting up which is necessary for operating system to work properly.

Kernel Mode Drivers interact with hardware devices. For each of drivers they have well defined routines and internal routines for exporting to rest of operating system. They exist in three layer drivers: highest level driver include file system drivers; intermediate level driver include function drivers generally rely on lowest level driver to function; lowest level drivers directly control hardware.

HAL is an abstraction layer between hardware and rest of OS that provide a platform on which kernel runs. In particular it does not abstract the instruction set. If there is need of abstracting the instruction set, it may be done with help of kernel.

VII. WEB REFERENCES

[1]. https://www.gnu.org/software/hurd/microkernel/mach/history.html
[2]. https://developer.apple.com/library/content/documentation/Darwin/Conceptual/KernelProgramming/Architecture/Architecture.html
[3]. https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/
[4]. https://www.tutorialspoint.com/android/android_architecture.htm
[5]. http://hiqes.com/android-linux-kernel-drivers/
[6]. https://source.android.com/devices/architecture/ha
[7]. https://www.cs.mcgill.ca/~rwest/wikispeedia/wpcd/wp/a/Architecture_of_Windows_NT.htm