Types of Kernel

Monolithic Kernel

- Multiple kernel source files are compiled into single kernel binary image. Such kernels are "monolithic" kernels.
- Since all functionalities present in single binary image, execution is faster.
- If any functionality fails at runtime, entire kernel may crash.
- Any modification in any component of OS, needs recompilation of the entire OS.
- Examples: BSD Unix, Windows (ntoskrnl.exe), Linux (vmlinuz), etc.

Micro-kernel

- Kernel is having minimal functionalities and remaining functionalities are implemented as independent processes called as "servers".
  - e.g. File management is done by a program called as "file server".
- These servers communicate with each other using IPC mechanism (message passing) and hence execution is little slower.
- If any component fails at runtime, only that process is terminated and rest kernel may keep functioning.
- Any modification in any component need to recompile only that component.
- Examples: Symbian, MACH, etc.

Modular Kernel

- Dynamically loadable modules (e.g. .dll / .so files) are loaded into calling process at runtime.
- In modular systems, kernel has minimal functionalities and rest of the functionalities are implemented as dynamically loadable modules.
- These modules get loaded into the kernel whenever they are called.
- As single kernel process is running, no need of IPC for the execution and thus improves performance of the system.
- Examples: Windows, Linux, etc.

Hybrid Kernel

- Mac OS X kernel is made by combination of two different kernels.
- BSD UNIX + MACH = Darwin

Linux - OS Structure

Linux components

- Static components (like monolithic kernel*)
  - Scheduler, Memory manager, IO subsystem, System calls, Process/thread management, etc.
  - All of these are compiled into single binary kernel image -- "vmlinuz".
    - /boot/vmlinuz
  - terminal> ls /boot
  - If any of these components need to modify, whole kernel should be recompiled.

- Dynamic components (like modular kernel)
  - File system managers, Device drivers, etc.
  - All of these are compiled into .ko (kernel object) files
    - /lib/modules//*
  - terminal> ls /lib/modules/
  - terminal> ls /lib/modules/uname -r
  - terminal> find /lib/modules/uname -r -iname "*.ko"
  - If any of these components need to modify, only that component needs to be recompiled and added into kernel again.

Linux kernel compilation

- Download Linux kernel from www.kernel.org.
- Extract Linux kernel using "tar" command.
- Configure the Linux kernel.
  - Which components to be compiled as static and which one to be compiled as dynamic.
  - Configure params like buffer size, timer interval, etc.
  - terminal> make menuconfig
- Compile all static components and build kernel image.
  - terminal> make bzImage
- Compile all dynamic components.
  - terminal> make modules
- Copy dynamic components into /lib/modules.
  - terminal> sudo make modules_install
- Copy kernel image into /boot.
  - terminal> sudo make install
- Reboot the system and select new kernel to boot.

Simple OS structure

- Small Operating systems like MS-DOS or few embedded OS follow a very simple structure.
- DOS operating system is made up of three files only.
  - COMMAND.COM <- command interpreter
  - MSDOS.SYS <- kernel
  - IO.SYS <- device drivers

Layered structure (architecture)

- OS is divided into multiple layers, so that each layer depends on the lower layer and provide functionality to the upper layer.
- Example: Windows, UNIX, Linux, etc.
- Windows OS have following layers
  - applications
  - system call APIs
  - system call implmenetation
  - Kernel Executive : File Mgr, Memory Mgr, Process Mgr, Scheduler, Thread Mgr, etc.
  - IO Subsystem
  - Device Drivers

    - Hardware Abstraction Layer