**LINUX EXERCISE (LAB 07)**

Exercise 1. Why is the Linux kernel source code very large in size while the compiled Linux kernel files have relatively small sizes? Describe the directory structure of the Linux kernel source code (including which directories and their contents, and for important directories, describe the meaning of each subdirectory).

The Linux kernel source code is large because it not only contains Linux kernel, but it also includes the entire collection of code, documentation, and configuration files that make up the Linux kernel and associated subsystems. The source code encompasses a wide range of functionalities, device drivers, file systems, networking protocols, and other components necessary for a complete operating system.

The exact directory structure may evolve with each kernel version. There are some important directories:

arch: Contains architecture-specific code, with subdirectories for various processor architectures (e.g., x86, ARM, PowerPC). Each subdirectory contains architecture-specific code, including low-level initialization, interrupt handling, and device drivers.

block: Contains block device-related code, including disk I/O, file systems, and storage-related operations.

crypto: Includes cryptographic algorithms and support for encryption and decryption operations.

drivers: Contains device driver code, organized into various subdirectories based on device types (e.g., network, USB, sound). Each subdirectory contains code for specific device drivers.

fs: Contains file system-related code, including support for different file systems like ext4, NTFS, and FAT.

include: Contains header files that define data structures, constants, function prototypes, and macros used throughout the kernel.

init: Contains the kernel initialization code, including the main entry point and early boot process.

ipc: Includes code related to interprocess communication mechanisms such as pipes, shared memory, and message queues.

kernel: Contains core kernel code, including process management, memory management, system calls, and scheduler.

lib: Contains utility functions, data structures, and common code used across the kernel.

mm: Contains memory management-related code, including virtual memory, page allocation, and memory mapping.

net: Contains networking-related code, including protocols, network device drivers, and socket operations.

scripts: Contains scripts used for various build and configuration tasks, such as generating configuration files and building kernel images.

security: Includes code related to security modules, access control, and permissions.

sound: Contains code related to sound card drivers, audio frameworks, and sound processing.

Exercise 2. Where should you store the Linux kernel on your system? What are the compressed and uncompressed sizes of the kernel in your Linux system?

I should store the Linux kernel in the /boot directory on a Linux system.

Exercise 3. Build a custom kernel version for the current computer system of the student (compile and install the new kernel). Note that the student should perform this task on a virtualized system (VMWare, VirtualBox, QEMU), not on a physical system.

Download the kernel:

A picture containing text, screenshot, font

Description automatically generated

Install necessary package:

A picture containing text, screenshot, software, operating system

Description automatically generated

Unzip kernel file:

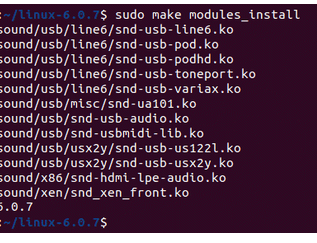


Build kernel:

A computer screen shot of a program

Description automatically generated with low confidence

Install required modules



Install kernel:

A picture containing text, screenshot, font

Description automatically generated

Tức là ta đang chạy linux với kernel version X, ta phải tải 1 bản kernel khác về và build và thay thế OS của ta sang dung kernel mới đó