Assignment#1
Compiling and Replacing the Linux Kernel

Introduction

Welcome to one of the most essential tasks for any aspiring systems engineer: compiling and installing your very own Linux kernel! If that sounds intimidating, don't worry—it's a fantastic learning experience, and you'll come out the other side with a much deeper understanding of how your system works.

So, what is the **kernel** anyway? Think of it as the heart of the operating system. It manages all interactions between your software and hardware. The kernel ensures that your apps can access memory, processors, and devices in an orderly manner. It's like the conductor of an orchestra, making sure that every piece of hardware and software plays nicely together. Fun fact: without the kernel, your computer is just a bunch of expensive metal and plastic doing nothing!

A Brief History of the Linux Kernel

The Linux kernel was first released by Linus Torvalds in 1991. Fun fact: Linus originally called it *Freax*, but a friend convinced him to upload it under the name *Linux* (a combination of his name and Unix). The rest, as they say, is history! Today, the Linux kernel powers everything from your phone to most of the internet.

Your Task

Your job is to compile your very own version of the Linux kernel and replace your system's current kernel with it. This isn't just about building something—it's about replacing the core of your system with your own creation. It's like upgrading your car's engine… while it's running! (Don't worry, we won't let you crash!)

Instructions

1. Preparation:

- Make sure you have the required dependencies for kernel compilation. You'll need tools like `gcc`, `make`, `ncurses`, and more.
- **Back up your system!** This is critical—if something goes wrong, you want a way back.
- Download the latest stable Linux kernel source code from kernel.org.

2. Compiling the Kernel:

- Extract the kernel source and navigate to the directory.
- Run `make menuconfig` to customize your kernel configuration. You can go with the default configuration for now (unless you want to explore).
- Compile the kernel using `make` (and grab a coffee—it'll take a while).

3. Installing the Compiled Kernel:

- Once compiled, install the modules: `sudo make modules_install`.
- Copy the kernel image and related files to `/boot`: `sudo make install`.

4. Updating the Bootloader:

- Update your bootloader configuration (GRUB in most cases) to include your newly compiled kernel.
- Reboot your system and select the new kernel from the boot menu.

5. Verification:

After rebooting, verify that your system is running the newly compiled kernel by running `uname -r`.

Bonus Challenge

Want to be adventurous? Tweak some kernel settings or modules to explore how your changes affect system performance. Just remember to document everything you change!

Submission Guidelines

- Write a brief report detailing the steps you took and the challenges you faced.
- Include screenshots of the compilation process and the final running kernel.

Academic Integrity

Do not copy from someone else. Any copied submissions will receive a zero. Work independently and ensure that the code and configuration you submit is entirely your own.

Deadline

The assignment must be submitted by Friday, October 4th, 2024, at 6:30 PM. Late submissions will not be accepted.

A Little Humor to Keep You Going

Remember, compiling a kernel is like making pizza. If you leave out the right ingredients (dependencies), or mess up the recipe (configuration), you'll end up with a mess! But when it's done right, it's *chef's kiss*.

Good luck, and may your system boot successfully on the first try (but let's be real, it probably won't)!