CS60038: Assignment 1

Building and Installing the Linux Kernel and Developing a Loadable Kernel Module

Submission Deadline: August 26, 2024 EOD

Submission Instructions

1. You need to submit the assignment through CSE Moodle: `https://moodlecse.iitkgp.ac.in/`. The course joining key is: **AOSD@Stud24**.
2. Only one member from each group should submit the assignment solution as a single zip file.
3. Mention the name and roll numbers of the group members in your submission.
4. Please document your implementation properly. You should include a README file explaining how to compile and run your code, as well as the test cases that you have tested.

Objective

The objective of this assignment is to get hands-on experience in building the Linux kernel from the source. This will help us to get familiar with the process of configuring, building, compiling, installing, and booting up a kernel. The second part of this assignment aims to develop a basic loadable kernel module (LKM).

You can download the 64-bit Ubuntu 22.04 LTS Desktop Image and use that in a Virtual Machine (VM). Please note that all the assignments will be evaluated on this platform and kernel version only.

For the assignment concerning configuring and building the Linux kernel, you need to use kernel version 5.10.223.

Part A: Configuring and Building Linux Kernel

Objective

In this assignment, students need to configure, build, and install a Linux kernel from the source.

You need to make the following configurations using `menuconfig`, and then build and install the compiled kernel over the Ubuntu 22.04 LTS Desktop version. You need to download the kernel source from `https://www.kernel.org/` only. Do not use any other third-party download servers.

1. Remove IPv6 protocol.
2. Make Kernel based Virtual Machine(KVM) an inbuilt feature instead of a kernel module.

The students need to submit the config file and a report describing the changes they are observing after installing this kernel in the system.

IPv6 Protocol

Internel Protocols are used in the Network layer of the TCP/IP stack and handles the routing of packets. IPv6 is the newer version of the IP protocol that offers a much larger address space (128 bits) compared to it's predecessor IPv4 (32 bits). It also offers additional features like a simplified header which leads to faster routing, and IPSec. To check your own IP addresses, you can use the `ip address` command. Use the `-6` flag for just the IPv6 address.

KVM

Kernel-based Virtual Machine (KVM) is a free and open-source virtualization module in the Linux kernel that allows the kernel to function as a hypervisor. It supports both Hardware Assisted Virtualisation and paravirtualisation using different APIs. To check if your OS has KVM enabled, check for the existence of `/dev/kvm`.

Part B: Loadable Kernel Module

Objective

In this part of the assignment, you need to develop a loadable kernel module for doing various jobs inside the kernel space. In addition to this, you need to handle concurrency, mutual exclusion, memory management, process management, and I/O control. You need to use 64-bit Ubuntu 22.04 Desktop with Kernel 5.10.223 (which you have compiled and installed on your VM).

Task Description

In this assignment, you need to write a loadable kernel module (LKM) that provides the functionality of a **Set** of max **Capacity** $\leq N$ inside the kernel mode. Your LKM should be able to handle 32-bit integers. Upon insertion of this LKM, it will create a file at the path `/proc/partb_<roll_nos>`. This path will be world-readable and writable. A userspace program will interact with the LKM through this file.

A user-space process can interact with the LKM in the following manner only:

1. It will open the file (`/proc/partb_<roll_nos>`) in read-write mode.
2. Write one byte of data to the file to initialize the set.
   - The first byte should contain the maximum capacity N of the set. The size N should be between 1 to 100 (including 1 and 100). If N is not within this range, produce an `EINVAL` error, and the LKM is left uninitialized.
3. Next, you need to implement write calls that should pass integers to be inserted in the set (one integer at a time). LKM must produce an invalid argument error in case a wrong argument is given (i.e., unsupported type). On a successful write call, LKM will return the number of bytes written (4 bytes for 32-bit integers). LKM will produce `EACCES` error for any excess write calls ($> N$) and return `-EACCES`.
4. As integers are written one by one in Step 3, they are inserted into the set. In between, there might also be intermediate read calls as explained in Step 5.
5. Read calls should return all the elements in the set. In case of a successful call, it will return the number of bytes read (no. of elements * 4 bytes), and in case of an error, it will return `-EACCES`.

6. Userspace process closes the file.

LKM should be able to handle concurrency and separate data from multiple processes. Also, no userspace program should be able to open the file more than once simultaneously. However, the userspace process should be able to reset the LKM (for the said process) by reopening the file. LKM needs to free up any resources it allocated for the process when it (the process) closes the file. Test your LKM implementation with multiple user-space processes writing and reading data in a set.