Date :
Roll no:

Experiment No 3

Installing, Configuring, and Monitoring Kernel Modules

Aim: To demonstrate Installing, Configuring, and Monitoring Kernel Modules

To demonstrate the installation, configuration, and monitoring of kernel modules using commands such as lsmod, insmod, modprobe, modinfo, dmesg, and rmmod.

Theory

Kernel modules are pieces of code that can be loaded and unloaded into the kernel upon demand without rebooting the system. They extend the functionality of the kernel without the need to recompile it. The Linux kernel supports dynamic loading and unloading of modules to provide additional functionality, such as device drivers, file systems, and system calls.

1. **Kernel Modules:**
    - **Loading Modules:** Modules can be dynamically loaded into the kernel using commands like insmod and modprobe.
    - **Removing Modules:** Modules can be removed using rmmod.
    - **Monitoring Modules:** Commands like lsmod and dmesg are used to monitor the status and messages related to kernel modules.
    - **Module Information:** modinfo provides detailed information about a kernel module.

Materials

- A computer or virtual machine with a Linux operating system.
- Root or sudo access to the system.

Procedure

1. **Viewing Loaded Modules:**
    - **Command:** lsmod
    - **Description:** Lists all currently loaded kernel modules.
    - Execution:

```
lsmod
```

Output: Displays the module name, size, and usage count.

2. **Loading a Kernel Module:**

- **Command:** insmod
- **Description:** Installs a module into the kernel.
- Execution:

```
sudo insmod /path/to/module.ko
```

3. **Using modprobe to Load a Module:**

- **Command:** modprobe
- **Description:** Loads a module along with its dependencies.
- **Execution:**

```
sudo modprobe module_name
```

4. **Viewing Module Information:**

- **Command:** modinfo
- **Description:** Displays information about a kernel module.
- **Execution:**

```
modinfo module_name
```

5. **Checking Kernel Messages:**

- **Command:** dmesg
- **Description:** Displays kernel messages, useful for diagnosing issues related to module loading.
- **Execution:**

```
dmesg | grep module_name
```

6. **Removing a Kernel Module:**

- **Command:** rmmod
- **Description:** Removes a module from the kernel.
- **Execution:**

```
sudo rmmod module_name
```

7. **Using modprobe to Remove a Module:**

- **Command:** modprobe -r
- **Description:** Removes a module along with its dependencies.
- **Execution:**

```
sudo modprobe -r module_name
```

Conclusion

This experiment demonstrates the process of installing, configuring, and monitoring kernel modules in a Linux system. By using commands such as lsmod, insmod, modprobe, modinfo, dmesg, and rmmod, users can dynamically manage kernel functionality. This capability is essential for extending system features and troubleshooting hardware and software issues without needing to reboot the system.