Task 1: Linux Kernel Architecture

Objective:
The objective of this task is to understand and document the internal architecture of the Linux Kernel. This document includes a detailed kernel architecture diagram, descriptions of major kernel components, related commands (where applicable), and expected outputs. This is prepared as a professional submission suitable for upload to a training GitLab repository.

1. Overview of Linux Kernel

The Linux Kernel is the core component of the Linux operating system. It acts as an interface between user-space applications and the underlying hardware. The kernel manages system resources such as CPU, memory, storage, and network devices to ensure efficient and secure operation.

2. Linux Kernel Architecture Diagram

Below is a logical representation of the Linux Kernel Architecture. This diagram shows how user space interacts with kernel space and how major kernel subsystems are organized.

```
---------------------------- USER SPACE ----------------------------
 |  Applications | Shell | System Utilities | Libraries (glibc)   |
 -------------------------------------------------------------------
                            | System Calls
 -------------------------------------------------------------------
 ---------------------------- KERNEL SPACE --------------------------
 | Scheduler | Memory Mgmt | VFS | Network Stack | IPC | Security   |
 |                  Device Drivers & Hardware Interface                  |
 -------------------------------------------------------------------
                            | Hardware Access
 -------------------------------------------------------------------
 ---------------------------- HARDWARE ------------------------------
 | CPU | RAM | Disk | Network Card | I/O Devices | Peripherals       |
 -------------------------------------------------------------------
```

3. Major Components of Linux Kernel

Process Scheduler

The scheduler is responsible for managing CPU time among running processes. It decides which process should run next based on scheduling policies like CFS (Completely Fair Scheduler). This ensures optimal CPU utilization and fairness.

Memory Management

Memory management handles allocation and deallocation of RAM. It manages virtual memory, paging, swapping, and ensures process isolation so that one process cannot access another process's memory.

Virtual File System (VFS)

VFS provides a common interface for different file systems such as ext4, XFS, and NTFS. It allows applications to access files uniformly without knowing the underlying file system implementation.

File Systems

File systems define how data is stored and retrieved from storage devices. The Linux kernel supports multiple file systems and handles permissions, file metadata, and disk I/O operations.

Network Stack

The network stack implements networking protocols such as TCP/IP, UDP, and ICMP. It manages data transmission, routing, packet filtering, and communication between systems over a network.

Device Drivers

Device drivers act as a bridge between hardware devices and the kernel. They allow the kernel to communicate with hardware like printers, keyboards, disks, and network cards in a hardware-independent manner.

Inter-Process Communication (IPC)

IPC mechanisms allow processes to communicate with each other. Linux supports IPC methods such as signals, pipes, message queues, shared memory, and semaphores.

Security Module

The security module enforces access control and system security policies. It includes mechanisms like SELinux and AppArmor to restrict unauthorized access and enhance system protection.

4. Commands Used and Outputs

The following commands can be used to observe kernel-related information on a Linux system.

Command: uname -r

Description: Displays the currently running kernel version.

```
student@student-virtual-machine:~$ uname -r
6.8.0-90-generic
student@student-virtual-machine:~$
```

Command: ls /proc

Description: Shows kernel and process-related information exposed by the kernel.

```
student@student-virtual-machine:~$ ls /proc
1     161     215   238     250883  273026  31589  31858  32063   403653  47   70   854          dma            pagetypeinfo
100   1635    2157  239     251     273029  31596  31864  32101   404367  49   71   86           driver         partitions
101   165     216   24      2518    273147  31598  31880  32103   409795  5    72   863          dynamic_debug  pressure
102   1658    217   240     252     275270  31604  31891  321218  423227  51   73   87           execdomains    schedstat
103   166     218   2407    253     275272  31605  31892  32157   429     52   74   88           fb             scsi
1042  1694    219   241     254     275273  31606  31905  32158   43      54   75   89           filesystems    self
105   17      22    242     2542    275287  31614  31906  32163   44      548  76   9            fs             slabinfo
107   1725    220   2425    2549    275311  31621  31908  32175   448628  549  77   90           interrupts     softirqs
108   1744    221   2426    255     275350  31627  31909  32180   449591  55   78   909          iomem          stat
109   18      222   243     256     275367  31630  31911  322     449593  56   79   91           ioports        swaps
110   182396  223   2437    257     275392  31631  31913  32253   45      57   796  92           irq            sys
1110  182412  224   244     258     275409  31642  31915  32261   457     58   798  93           kallsyms       sysrq-trigger
113   182430  225   245     259     275410  31655  31917  32268   458185  59   80   94           kcore          sysvipc
114   19      226   246     26      275481  31695  31918  32293   458745  6    804  95           keys           thread-self
1178  1935    227   247     260     2794    31706  31921  323     458905  60   806  96           key-users      timer_list
1184  195     228   248     261     28      31713  31925  32329   458977  61   81   97           kmsg           tty
1190  196     229   249     262     2880    31718  31929  32366   459146  610  811  976          kpagecgroup    uptime
12    2       23    249917  263     2898    31739  31931  32935   459297  62   812  98           kpagecount     version
122   20      230   249972  264     29      31745  31934  33      459298  625  82   99           kpageflags     version_signature
124   207     2308  250     265     2911    31748  31941  34      459391  63   826  acpi         latency_stats  vmallocinfo
125   2073    231   250055  266     2919    31796  31949  34226   459500  64   83   bootconfig   loadavg        vmstat
1255  208     2314  250060  267     2985    31802  3195   35      459582  65   830  buddyinfo    locks          zoneinfo
126   209     232   250095  2679    2994    31811  31963  36      459793  653  831  bus          mdstat
127   2091    233   250110  268     3       31812  31977  362     459941  662  835  cgroups      meminfo
128   21      234   250278  27      30      31817  31989  3861    459995  668  836  cmdline      misc
13    210     2343  2503    2717    3010    31822  31992  389561  46      68   84   consoles     modules
14    211     2347  250571  271987  3024    31836  32     392     460003  69   846  cpuinfo      mounts
142   212     235   250725  271988  308142  31837  32000  4       460052  691  849  crypto       mpt
15    213     236   250799  271989  3090    31846  32009  400     460075  698  85   devices      mtrr
16    214     237   250841  271990  3101    31856  32038  401     460144  7    851  diskstats    net
student@student-virtual-machine:~$
```

Command: free -h

Description: Displays memory usage managed by the kernel.

```
student@student-virtual-machine:~$ free -h
               total        used        free      shared  buff/cache   available
Mem:            15Gi       3.6Gi       5.4Gi       105Mi       6.6Gi        11Gi
Swap:          2.0Gi          0B       2.0Gi
student@student-virtual-machine:~$
```

5. Conclusion

This document provides a detailed understanding of the Linux Kernel architecture. By studying its components and structure, one can better understand how Linux efficiently manages hardware and software resources. This knowledge is fundamental for system administration, kernel development, and advanced Linux usage.