Linux and Shell Scripting

By Rajath Kumar K S

Contents

You'll get a prospect of Unix/Linux and Shell Scripting Training what you're going to learn for the next ten days.

01 Unix to Linux

02 Why Linux ?

03 What We'll Learn

04 Core Concepts of Linux

05 Linux File System & Storage

06 Shell and Bash

07 Deep dive into Linux Commands

WHO AM I ?

I’m RAJATH KUMAR K S 🇮🇳

I have been into the industry for more than a decade in building and delivering applications across domains.

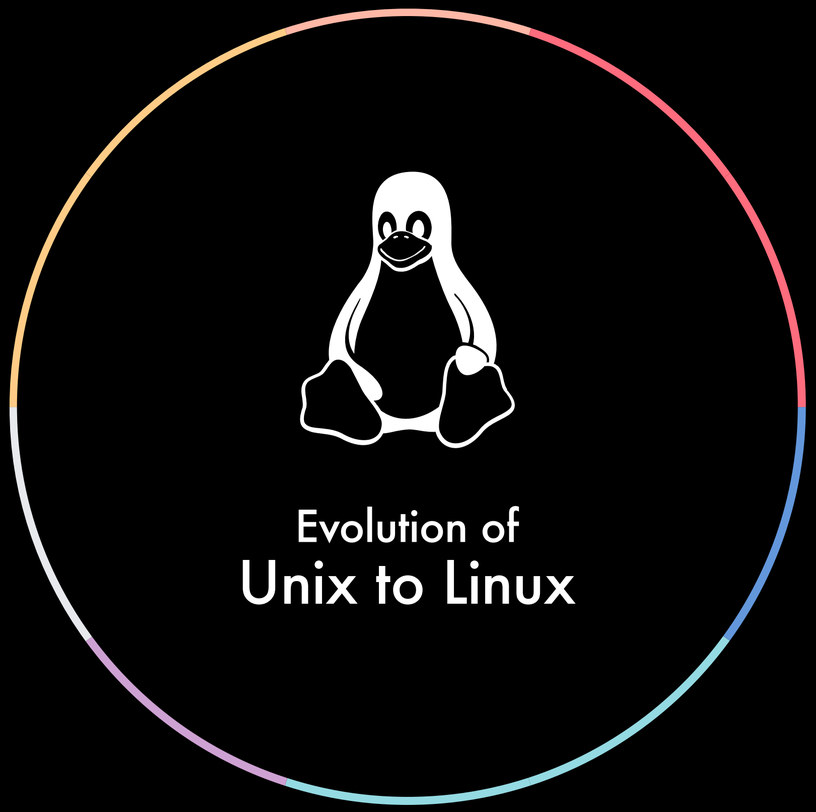
Expertise in C, Embedded, Linux, IoT, Python, ML, DL, GenAI 🔥 and Backend Frameworks such as Django, Flask, FastAPI.

3x Azure Certified, 2x AWS Certified 🌦️

Currently heading company as a Thought Leader of Software Development and Research at Elementure Private Limited, Bangalore 🏦

Teaching came with my passion, Trained across companies and organisation including ISRO, HP, Oracle, TCS, Bosch, Bharath Electronics to name a few. 📚

01 Unix to Linux



1960’S - ORIGIN OF UNIX

- Developed by Ken Thompson, Dennis Ritchie and Others
- at AT & T Bell Laboratories.
- They’ll develop C Programming Later in 1970’s
- Initially created for a mini computer called PDP-7

1970’S - PORTABLE TO C

- Portability through the C Programming Language
- Adoption by academic institutions.
- Development of various Unix flavors (eg. BSD)

1980’S - EXPANSION OF UNIX

- Commercial adoption by companies like IBM, HP and Sun Microsystems.
- Standardisation efforts, including POSIX
- Emergence of System V and BSD Variants

01 Unix to Linux

1990’S - THE BIRTH OF LINUX

- Linus Torvalds initiates Linux in 1991
- Inspired by the Minix Operating System.
- Introduced the Linux Kernel 1.0 in 1994
- Open-Source from the beginning, licensed under the GPL

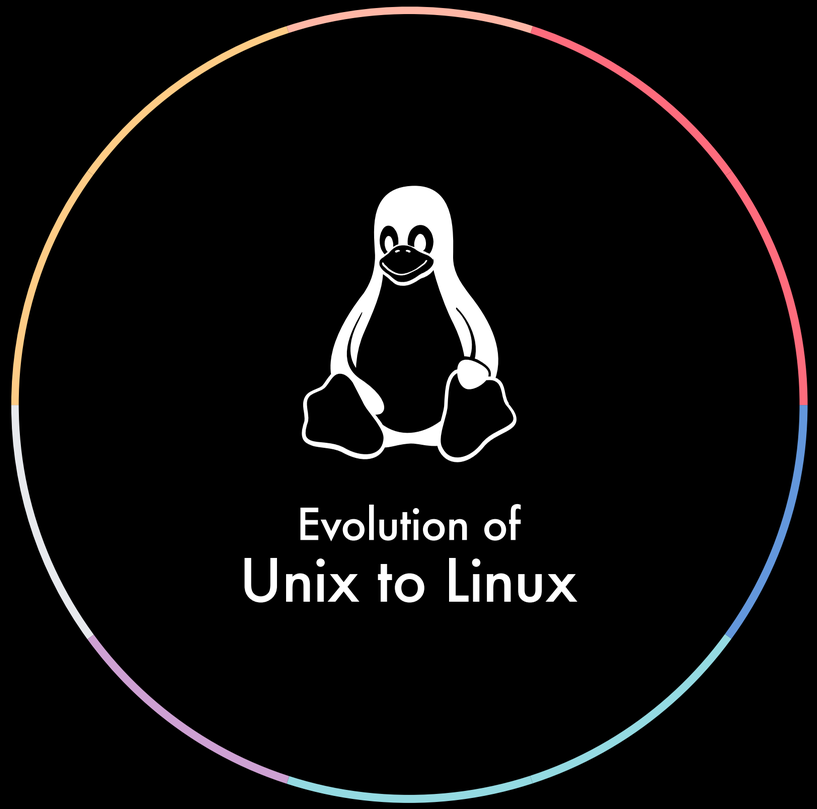
1995’S - GROWTH OF LINUX

- Rapid Development and Community Contributions.
- Formation of key distributions (Eg. Debian, Red Hat, Slackware)
- Development of GUIs like KDE and GNOME
- Increasing adoption in academia and industry

2000’S - WIDESPREAD ADOPTION

- Growth of Distributions like Ubuntu and Fedora.
- Advances in Kernel Development and Hardware Support
- Widespread enterprise adoption (eg., servers, supercomputers)

MODERN LINUX



Dominance in Server / Cloud

Utilization in Embedded Devices

Employs in Personal Computers
Such as Ubuntu, Fedora and so on

Wide Adoption in Mobile OS

Practices in Cyber Security

Increasing Adoption in 5G
infrastructure, IoT, AI and So On

02 Why Linux ?

WHAT MAKES LINUX POPULAR

47.8% of Developers prefer Linux for their primary OS

Performance and Customization

- Linux Requires fewer resources; can run on systems with as little as 256 MB RAM
- Average Linux Server Uptime is over 99.99%, with many servers running non-stop for years (Netcraft 2023)
- Over 600 different distributions tailored for various needs (Distrowatch 2024)

INSIGHTS

500+

100% of top fastest supercomputers in the world, they run on Linux

97 %

of the top 1 million websites that we go to they run on Linux,

71.85 %

share on mobile OS market in the name of Android, which is based on Linux,

DATE	BASED ON
03/08/24	STACKOVERFLOW

02 Why Linux ?

WHAT MAKES LINUX POPULAR

86% of Containers run on Linux according to Docker

Cost Efficiency and Security

- $ 0 Licensing Fees: Linux is Open Source and Free, resulting in significant cost savings for businesses.
- Linux experiences less than 5% of malware attacks compared to windows, Over 90% of vulnerabilities patched within a week of found (Kaspersky lab, 2023 and ZDNet, 2023)

INSIGHTS

86000+

active developers contributing to the Linux kernel (Linux Foundation, 2024)

59 %

Total Cost of Ownership (TCO), reduced compared to proprietary systems (Gartner 2023)

80 %

Linux systems can reduce maintenance costs by up to 80%, (IDC 2023)

DATE	BASED ON
03/08/24	MULTIPLE SOURCE

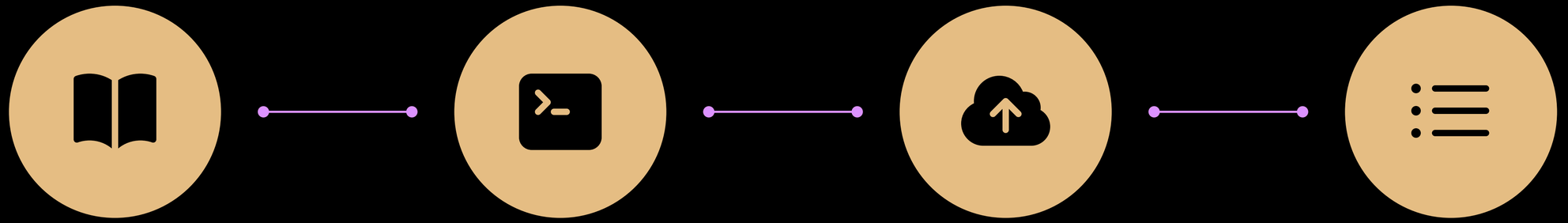
03 What We'll Learn

01 Intro to Unix/Linux and Basic Commands
02 Advance File and Directory Management
03 Text Processing and Regular Expressions
04 Intro to Shell Scripting
05 Control Structures and Functions
06 Working with Processes and Job Control
07 Advance Shell Scripting Techniques
08 Working with Unix File System and Networking
09 Advanced Networking & Shell Scripting for System Administration
10 Best Practices of Shell Scripting

40 Hours
20+ Deployable Scripts
05 Assignments
150 Quizzes

10 DAYS

03 How We'll Learn

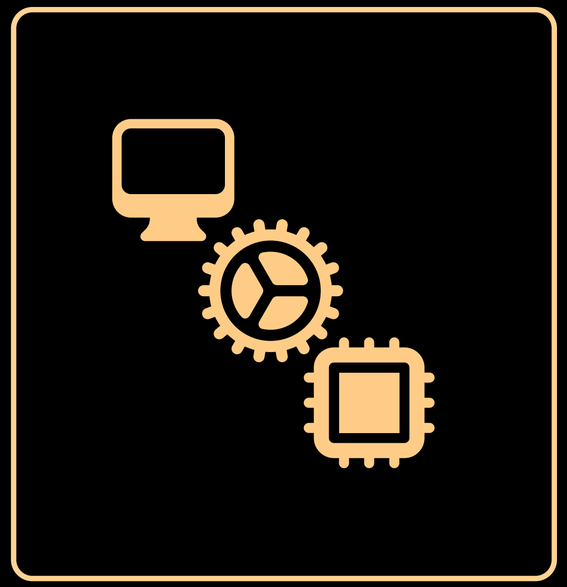


Theory + Labs

Projects or
Deployable Scripts

Quiz + Assignments

04 Core Concepts of Linux



01 Linux Kernel

02 Kernel Space and User Space

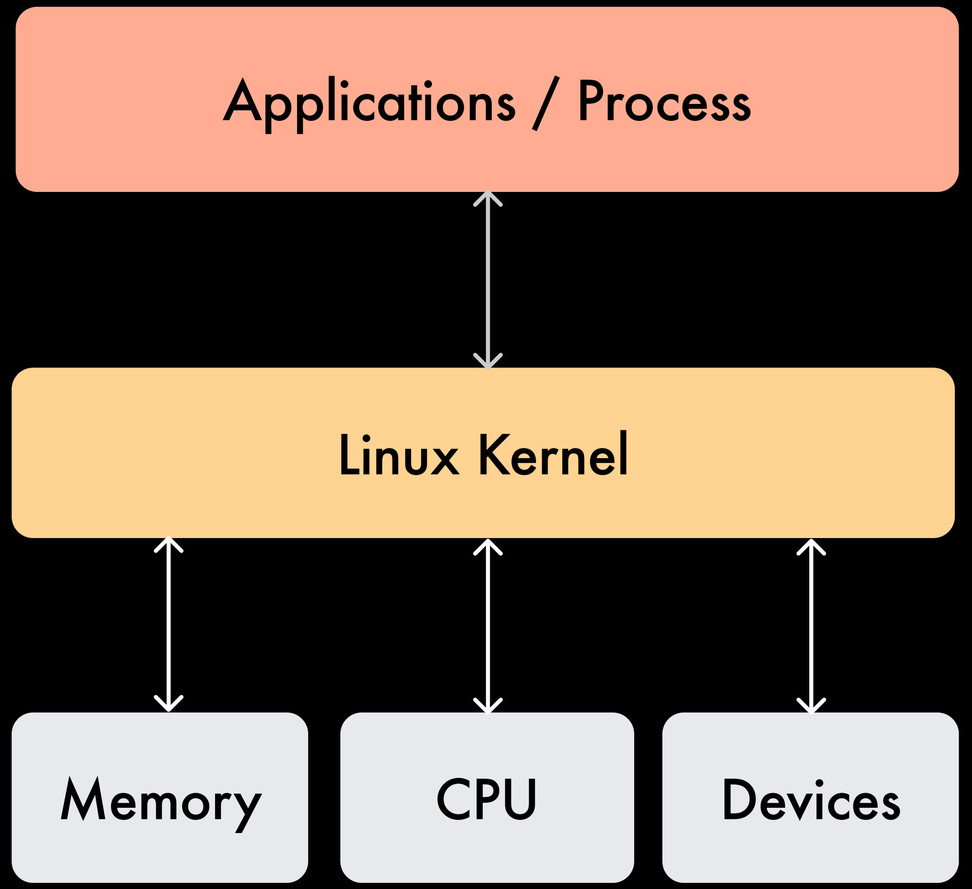
03 Working with Hardware

04 Linux Boot Sequence

05 SYSTEMD TARGETS (Run Levels)

4.1 What is Kernel ?

Kernel is the Major Component of an Operating System and is the core interface between a computer's hardware and its processes. It communicates between the two managing resources as efficiently as possible.

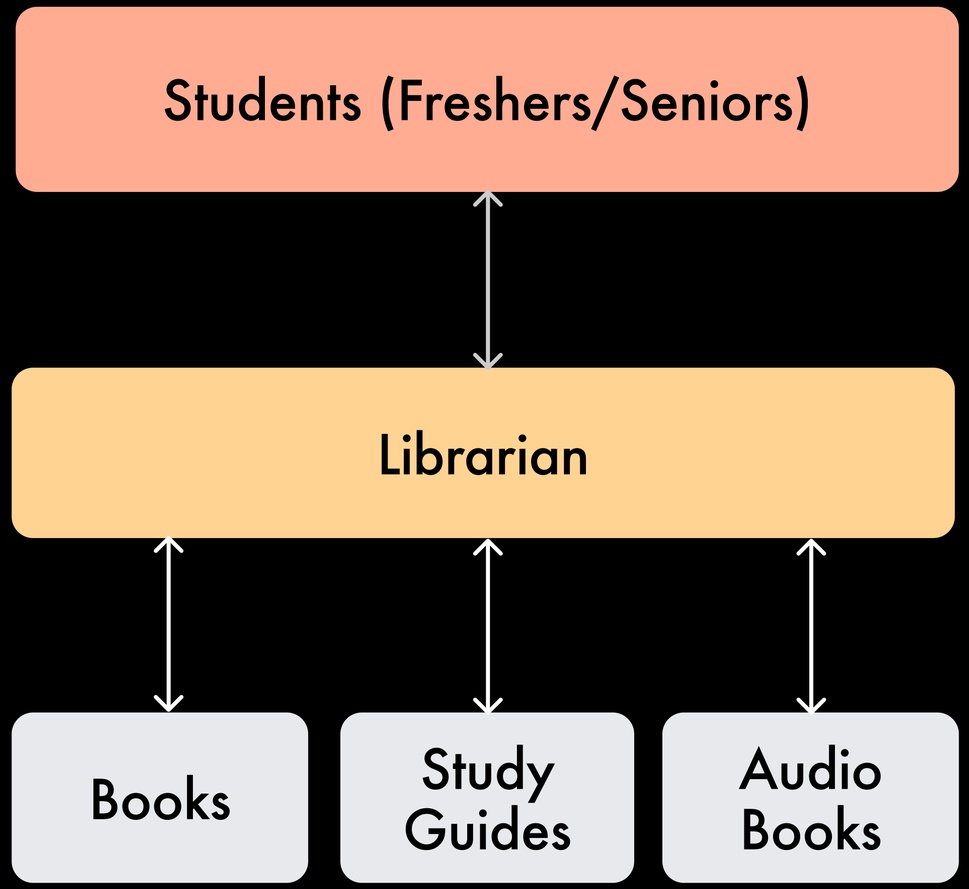


4.1 What is Kernel ?

LINUX KERNEL

Kernel is the Major Component of an Operating System and is the core interface between a computer's hardware and its processes. It communicates between the two managing resources as efficiently as possible.

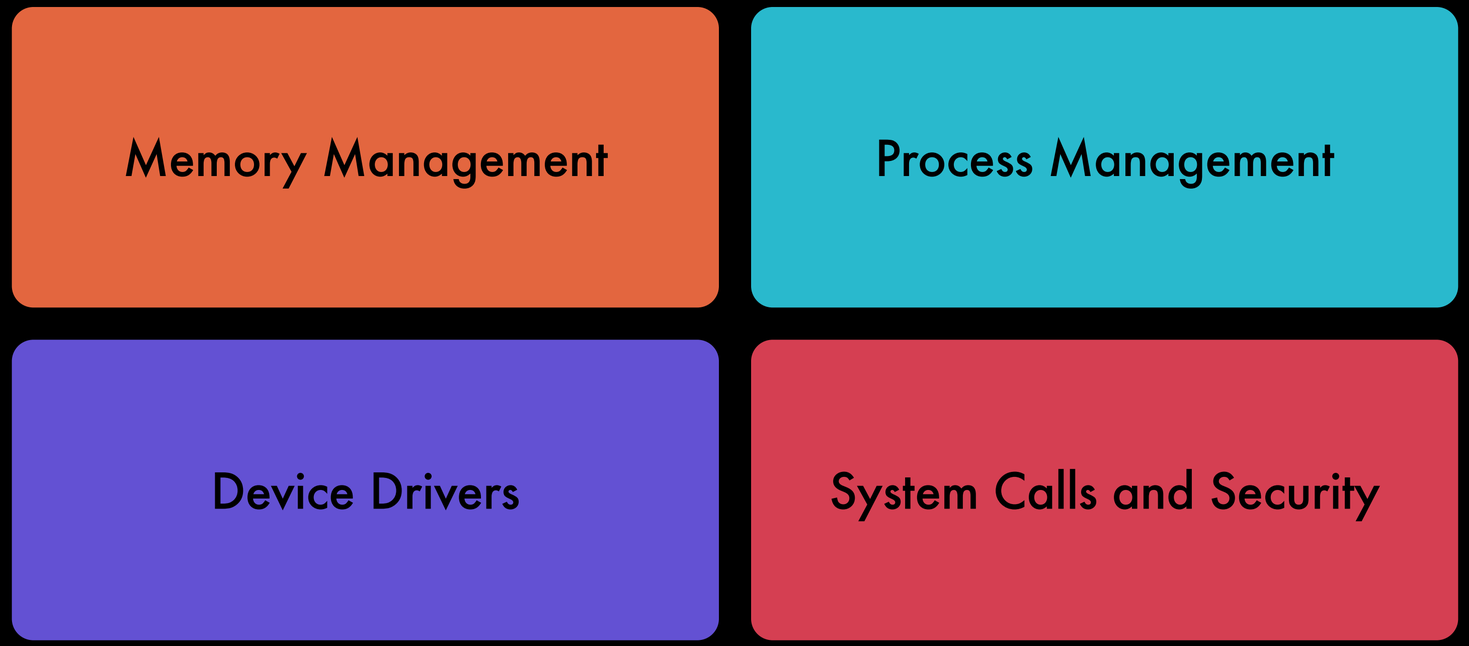
LIBRARY



4.1 What is Kernel ?

LINUX KERNEL

RESPOSNIBILITIES OF KERNEL



4.1 What is Kernel ?

LINUX KERNEL

How to Check Kernel Version ?

```
rajath@rhelpro ~ (0.025s)
uname -r
5.14.0-427.26.1.el9_4.x86_64
```

In my case, I'm running on Red Hat Enterprise Linux (v 9.4). Which uses the Linux Kernel version 5.14.0-427.XX.X..

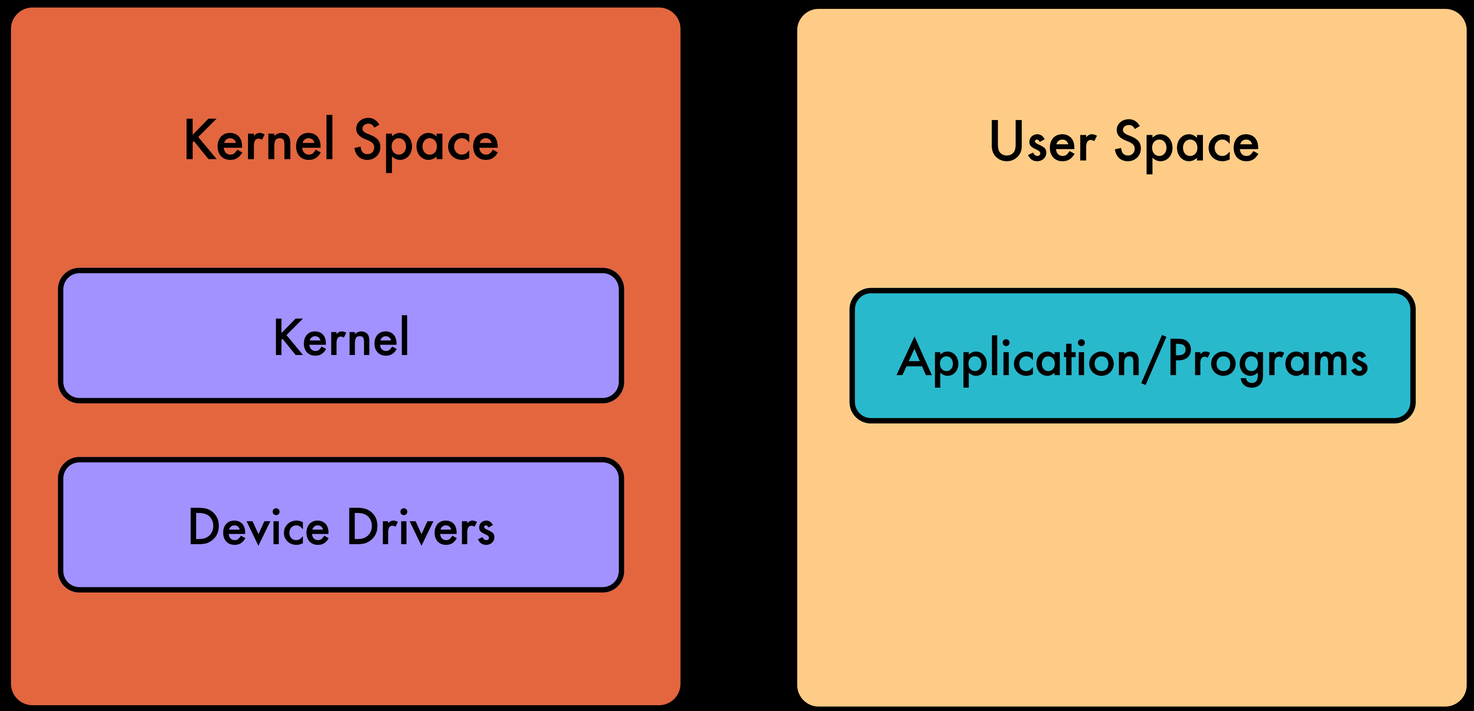
For the latest version of Linux Kernel Release, Please refer to the website

https://www.kernel.org

4.2 What is Kernel Space & User Space ?

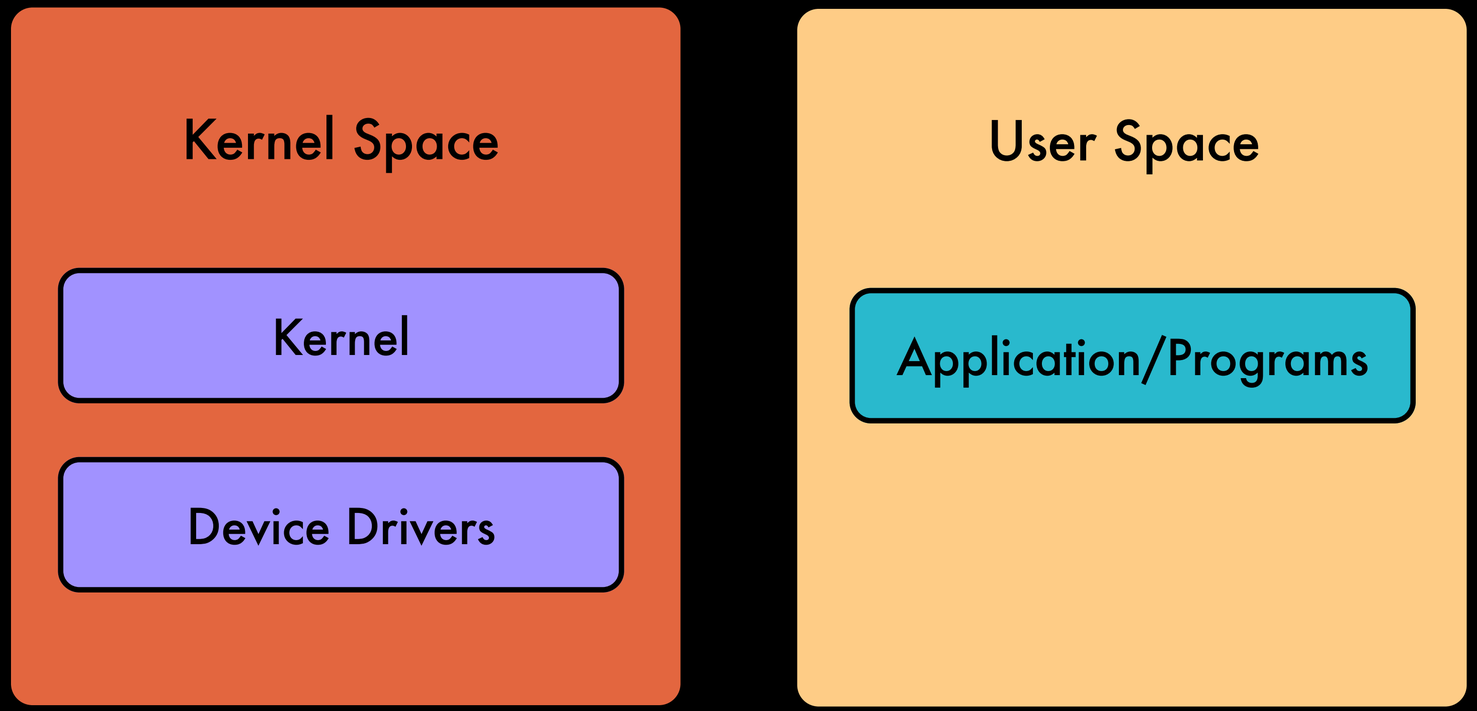
Memory is divided into two areas known as Kernel Space and User Space. These are synonymous to the terms Kernel and User Mode.

KERNEL SPACE



4.2 What is Kernel Space & User Space ?

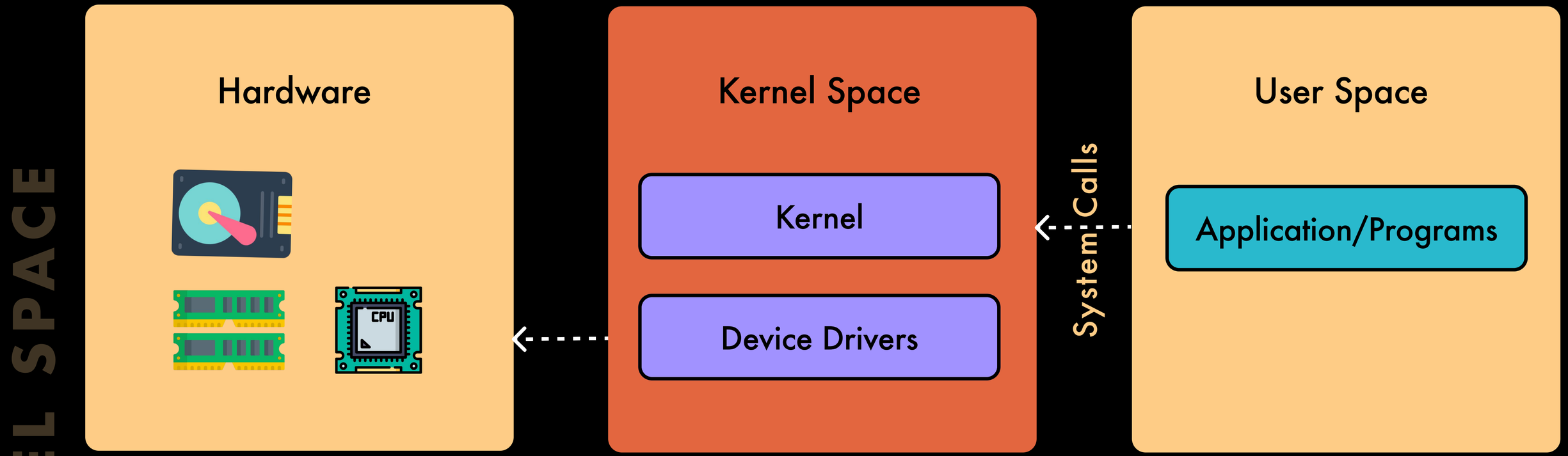
KERNEL SPACE



- Kernel Code
- Kernel Extensions
- Device Drivers

- Editors
- C / Python / Java
- Docker Containers

4.2 What is Kernel Space & User Space ?



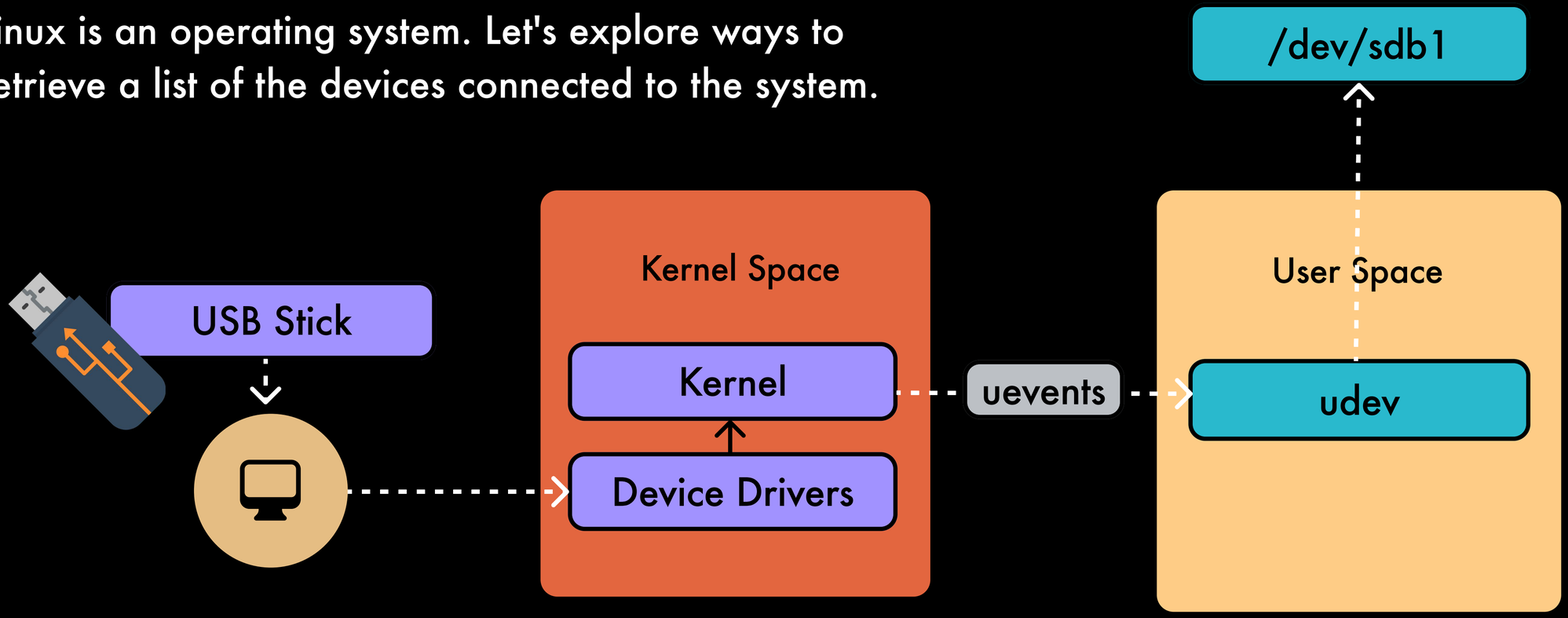
- Open a File
- Write to a File
- List Processes
- Define a variable

4.3 Working with Hardware

Linux identifies and manages hardware devices attached to the system.

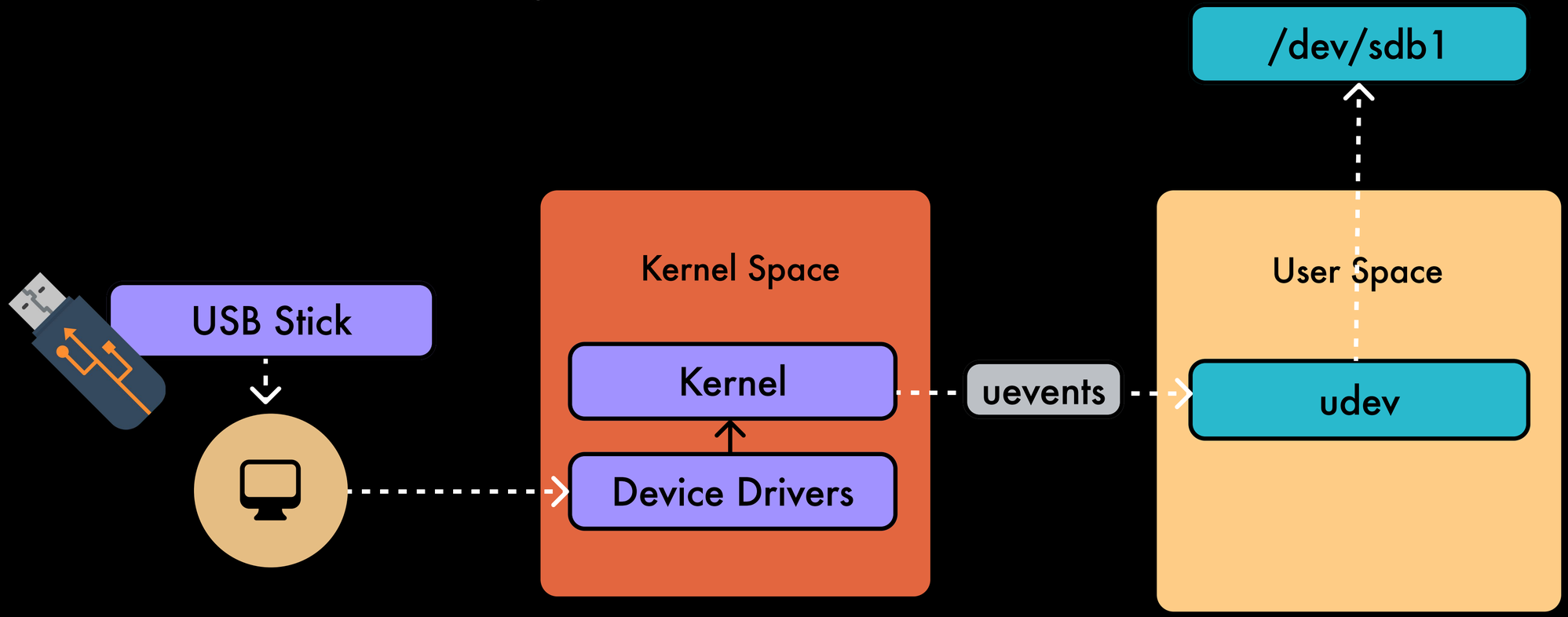
Linux is an operating system. Let's explore ways to retrieve a list of the devices connected to the system.

HARDWARE



4.3 Working with Hardware

Let us comprehend this through an example. Let us consider a scenario where a PC is connected to a USB Stick. How does Linux recognize the USB Stick?



4.3 Working with Hardware

HARDWARE

Using DMESG

Dmesg is a tool used to display messages from an area of the kernel called Ring Buffer.

These messages also contain logs from hardware devices that the kernel detects.

Using UDEVADM

The udevadm utility is a management tool for udev. The udevadm info command queries the udev database for device information.

4.3 Working with Hardware

HARDWARE

We're using grep since dmesg generates all the messages from the Kernel

Using DMESG

~ dmesg | grep -i usb

```
rajath@rhelpro ~ (0.032s)
dmesg | grep -i usb
[    7.321973] usbcore: registered new interface driver uvcvideo
[  392.419686] usb 2-1.1: new high-speed USB device number 4 using ehci-pci
[  392.499084] usb 2-1.1: New USB device found, idVendor=0781, idProduct=5567, bcdDevice= 1.00
[  392.499098] usb 2-1.1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[  392.499104] usb 2-1.1: Product: Cruzer Blade
[  392.499107] usb 2-1.1: Manufacturer: SanDisk
[  392.499111] usb 2-1.1: SerialNumber: 030244050823231418 31
[  392.520739] usb-storage 2-1.1:1.0: USB Mass Storage device detected
[  392.520890] scsi host6: usb-storage 2-1.1:1.0
[  392.520964] usbcore: registered new interface driver usb-storage
[  392.524910] usbcore: registered new interface driver uas
```

SanDisk made USB. which is Inserted

4.3 Working with Hardware

HARDWARE

Using UDEVADM

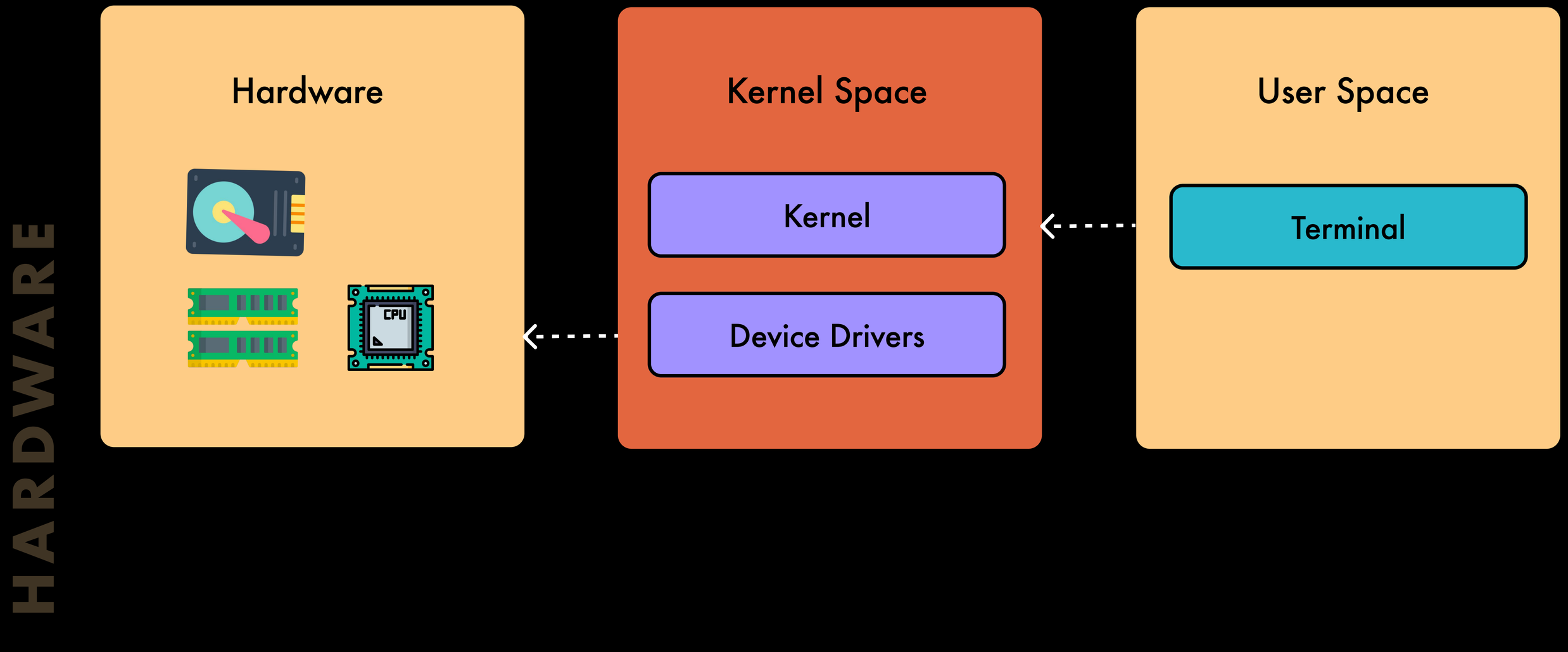
~ udevadm monitor

```
rajath@rhelpro ~ (42.015s)
udevadm monitor

UDEV  [1980.815438] add      /devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0/block/sdb (block)
UDEV  [1981.008917] add      /devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0/block/sdb/sdb1 (block)
UDEV  [1981.014033] bind     /devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0 (scsi)
KERNEL[1981.142405] change   /devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0/block/sdb/sdb1 (block)
UDEV  [1981.356497] change   /devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0/block/sdb/sdb1 (block)
```

~ udevadm info --query=path --name=/dev/sdb

```
rajath@rhelpro ~ (0.031s)
udevadm info --query=path --name=/dev/sdb

/devices/pci0000:00/0000:00:1d.0/usb2/2-1/2-1.5/2-1.5:1.0/host6/target6:0:0/6:0:0:0/block/sdb
```

4.3 Working with Hardware



4.3 Working with Hardware

HARDWARE

Using **`lspci`** command stands for list PCI. This command used to display all the PCI devices. Examples, Ethernet Cards, Video Cards, Wireless Adaptors and So On.

PCI Stands for Peripheral Component Interconnect

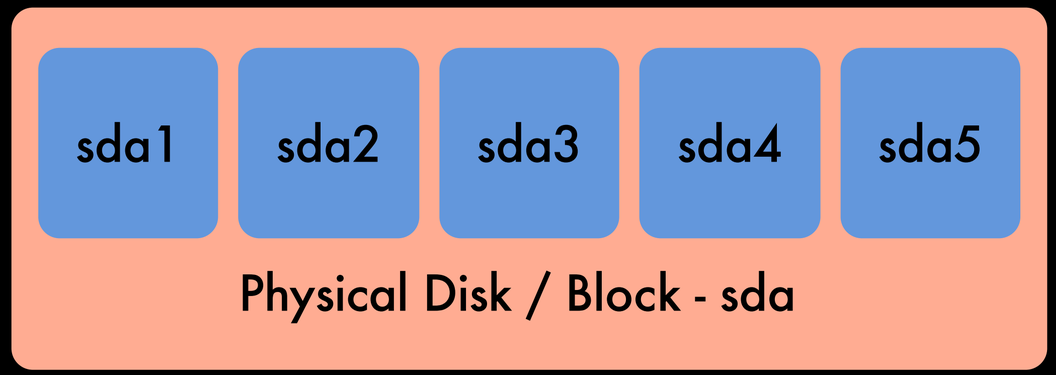
```
rajath@rhelpro ~ (0.7s)
lspci

00:00.0 Host bridge: Intel Corporation 2nd Generation Core Processor Family DRAM Controller (rev 09)
00:01.0 PCI bridge: Intel Corporation Xeon E3-1200/2nd Generation Core Processor Family PCI Express Root Port (rev 09)
00:02.0 VGA compatible controller: Intel Corporation 2nd Generation Core Processor Family Integrated Graphics Controller (rev 09)
00:16.0 Communication controller: Intel Corporation 6 Series/C200 Series Chipset Family MEI Controller #1 (rev 04)
00:1a.0 USB controller: Intel Corporation 6 Series/C200 Series Chipset Family USB Enhanced Host Controller #2 (rev 04)
00:1b.0 Audio device: Intel Corporation 6 Series/C200 Series Chipset Family High Definition Audio Controller (rev 04)
00:1c.0 PCI bridge: Intel Corporation 6 Series/C200 Series Chipset Family PCI Express Root Port 1 (rev b4)
00:1c.3 PCI bridge: Intel Corporation 6 Series/C200 Series Chipset Family PCI Express Root Port 4 (rev b4)
00:1d.0 USB controller: Intel Corporation 6 Series/C200 Series Chipset Family USB Enhanced Host Controller #1 (rev 04)
00:1f.0 ISA bridge: Intel Corporation HM65 Express Chipset LPC Controller (rev 04)
00:1f.2 SATA controller: Intel Corporation 6 Series/C200 Series Chipset Family 6 port Mobile SATA AHCI Controller (rev 04)
00:1f.3 SMBus: Intel Corporation 6 Series/C200 Series Chipset Family SMBus Controller (rev 04)
01:00.0 3D controller: NVIDIA Corporation GF119M [GeForce GT 520MX] (rev a1)
02:00.0 Network controller: Intel Corporation Centrino Wireless-N 130 (rev 34)
03:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168/8211/8411 PCI Express Gigabit Ethernet Controller (rev 06)
```

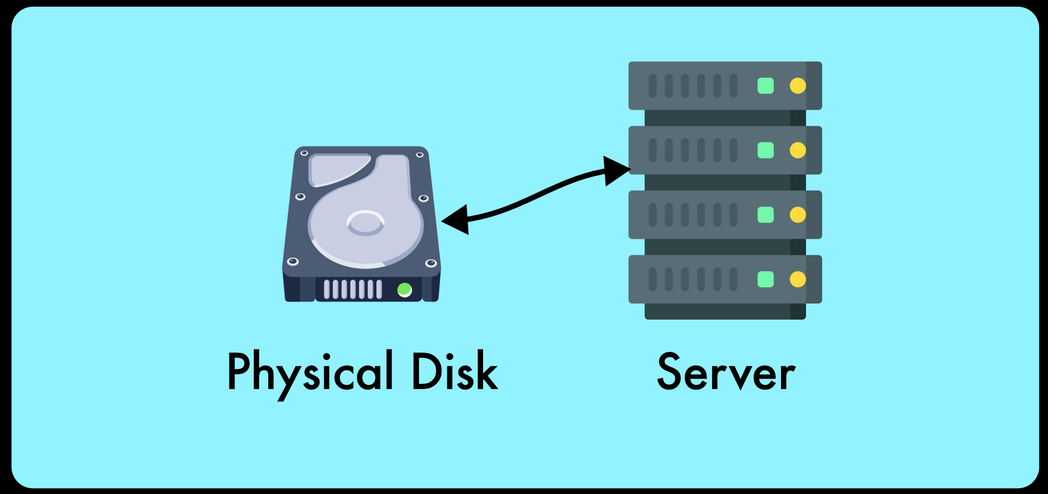
4.3 Working with Hardware

HARDWARE

Using **`lsblk`** lists information about block devices, Here sda is the physical disk. The sda1 to sda2 are the partitions created for the disk.

```
rajath@rhelpro ~ (0.026s)
lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda      8:0    0 223.6G  0 disk
├─sda1   8:1    0     1G  0 part /boot
├─sda2   8:2    0   7.8G  0 part [SWAP]
├─sda3   8:3    0    70G  0 part /
├─sda4   8:4    0     1K  0 part
└─sda5   8:5    0 144.8G  0 part /home
sdb      8:16   1  28.7G  0 disk
└─sdb1   8:17   1  28.7G  0 part /run/media/rajath/rajath-usb
```





4.3 Working with Hardware

HARDWARE

To display information about the CPU architecture use the **`lscpu`** command.

```
rajath@rhelpro ~ (0.039s)
lscpu

Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         36 bits physical, 48 bits virtual
  Byte Order:            Little Endian
CPU(s):                  4
  On-line CPU(s) list:   0-3
Vendor ID:               GenuineIntel
  Model name:            Intel(R) Core(TM) i5-2450M CPU @ 2.50GHz
    CPU family:          6
    Model:               42
    Thread(s) per core:  2
    Core(s) per socket:  2
    Socket(s):           1
    Stepping:            7
    CPU(s) scaling MHz:  83%
    CPU max MHz:         3100.0000
    CPU min MHz:         800.0000
    BogoMIPS:            4989.01
```

The command provides detailed information such as,

- The number of cores
- Threads
- model
- Vendor
- Speed

So On.,

4.3 Working with Hardware

HARDWARE

The **lsmem** command can be used to list the available memory in the system.

```
rajath@rhelpro ~ (0.025s)
lsmem --summary

Memory block size:         128M
Total online memory:         8G
Total offline memory:        0B
```

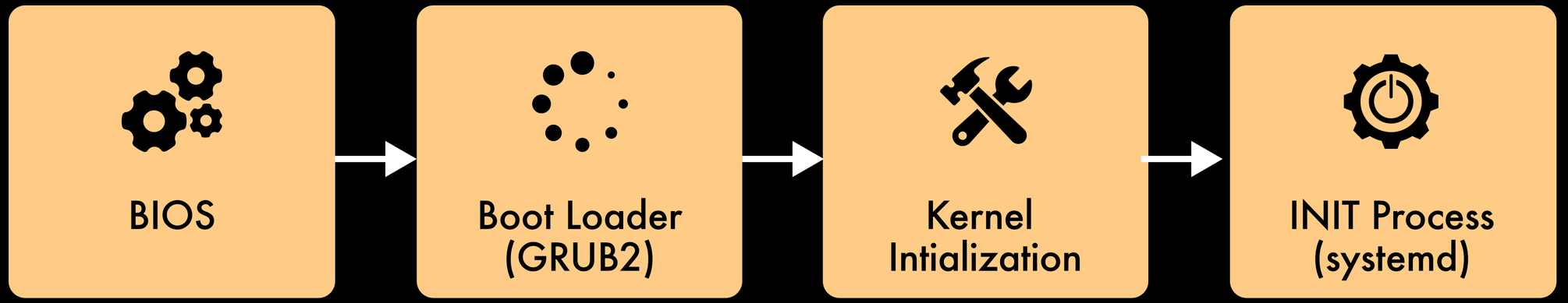
The **free** command can be used to see total vs free memory.

```
rajath@rhelpro ~ (0.026s)
free -g

               total        used        free
Mem:               7           2           3
Swap:              7           0           7
```

free -g to get info in GB.
free -m to get info in MB.
free -k to get info in KB.

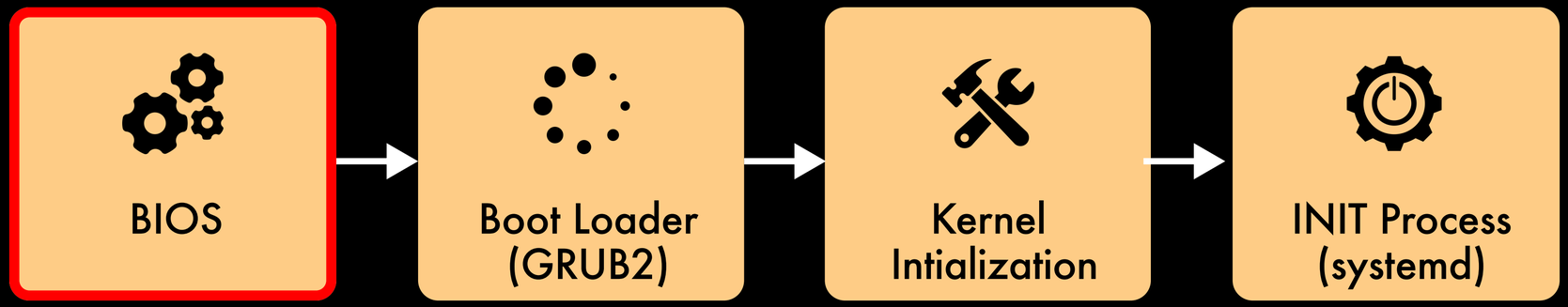
4.4 Linux Boot Sequence

BOOT SEQUENCE



4.4 Linux Boot Sequence

BOOT SEQUENCE

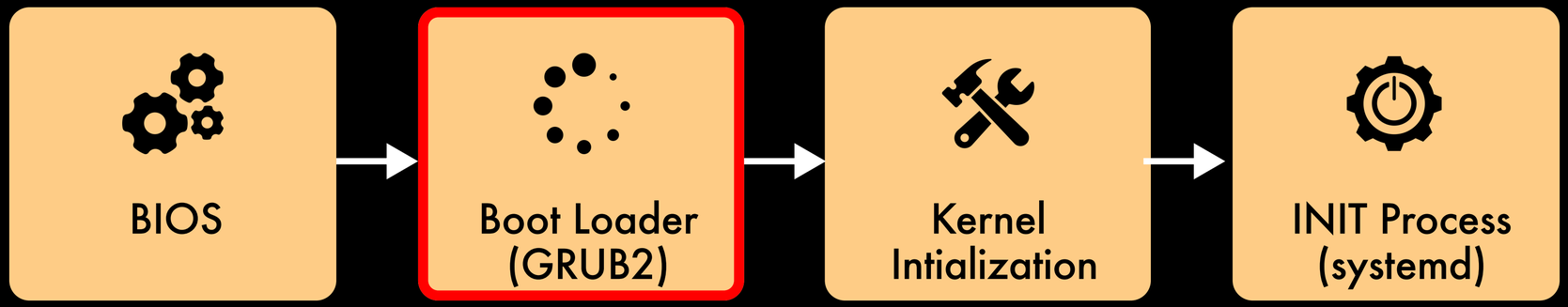


BIOS Intialization

- When the system is powered on, the BIOS (Basic Input/Output System) or UEFI (Unified Extensible Firmware Interface) firmware is executed. It performs initial hardware checks, such as testing the memory and initializing hardware components.
- The BIOS/UEFI firmware locates the boot device (such as a hard drive, SSD, or USB drive) and loads the bootloader from the boot sector.

4.4 Linux Boot Sequence

BOOT SEQUENCE

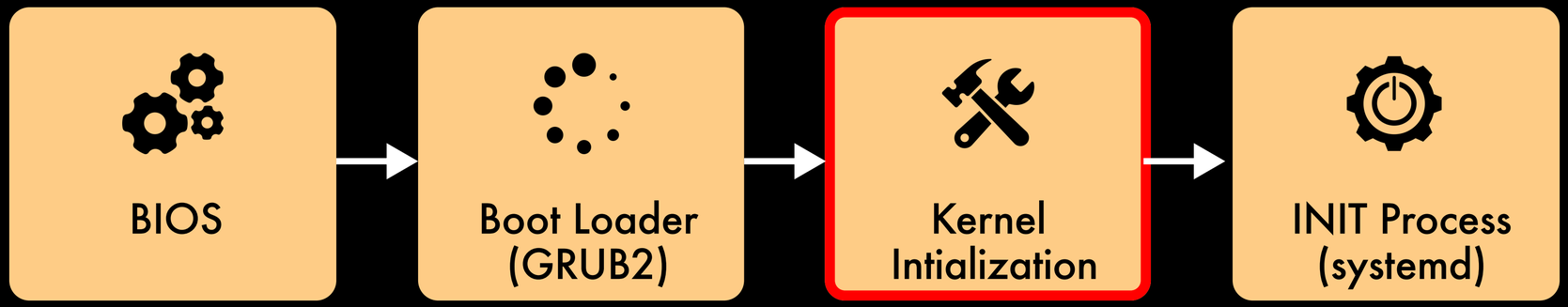


Bootloader Execution

- The bootloader (e.g., GRUB - Grand Unified Bootloader, LILO - Linux Loader, or systemd-boot) is loaded into memory. The bootloader is responsible for loading the Linux kernel into memory.
- The bootloader presents a menu of available kernels and boot options (if configured). The user can select a kernel or let the default kernel boot automatically after a timeout.

4.4 Linux Boot Sequence

BOOT SEQUENCE

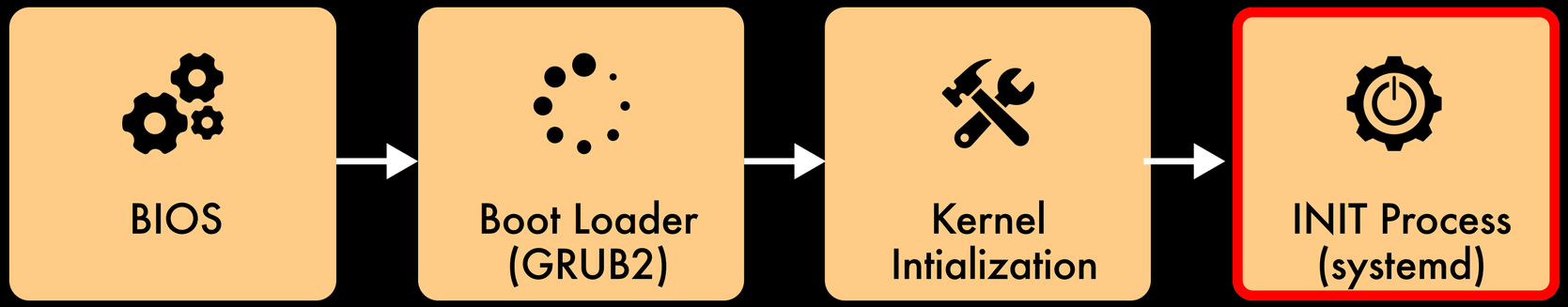


Kernel Loading and Initialization

- The chosen Linux kernel gets loaded into memory. The bootloader hands over control to the kernel along with the initial RAM disk (initrd or initramfs), which includes important drivers and scripts required to mount the root filesystem.
- The kernel then mounts the root filesystem specified in the bootloader configuration (e.g., /) and starts the init process (PID 1), which is the initial user-space process.

4.4 Linux Boot Sequence

BOOT SEQUENCE



Init (Systemd) Execution

- The init process (nowadays often replaced by systemd, Upstart, or other init systems) is responsible for bringing up the user-space environment.
- The init system reads its configuration files (e.g., /etc/inittab for traditional SysV init or unit files for systemd) and starts the necessary services and daemons.

4.5 Systemd Targets (Runlevels)

Runlevels are an obsolete way to start and stop groups of services used in SysV init. systemd provides a compatibility layer that maps runlevels to targets.

RUNLEVELS

- The system enters the default runlevel (for SysV init) or target (for systemd) as specified in the configuration.
- Services and daemons configured for the default runlevel/target are started. These include network services, display managers, and other system services.

Run level	Target
0	poweroff.target
1	rescue.target
2, 3, 4	multi-user.target
5	graphical.target
6	reboot.target

4.5 Systemd Targets (Runlevels)

RUNLEVELS

The **`runlevel`** command is used to find out current runlevel of the system. A runlevel is a mode of operation in Unix and Unix-based operating systems that determines what services and processes should be running.

```
rajath@rhelpro ~ (0.026s)
runlevel
N 5
```

5 Linux File System and Storage

01 File Types in Linux

02 Hierarchy

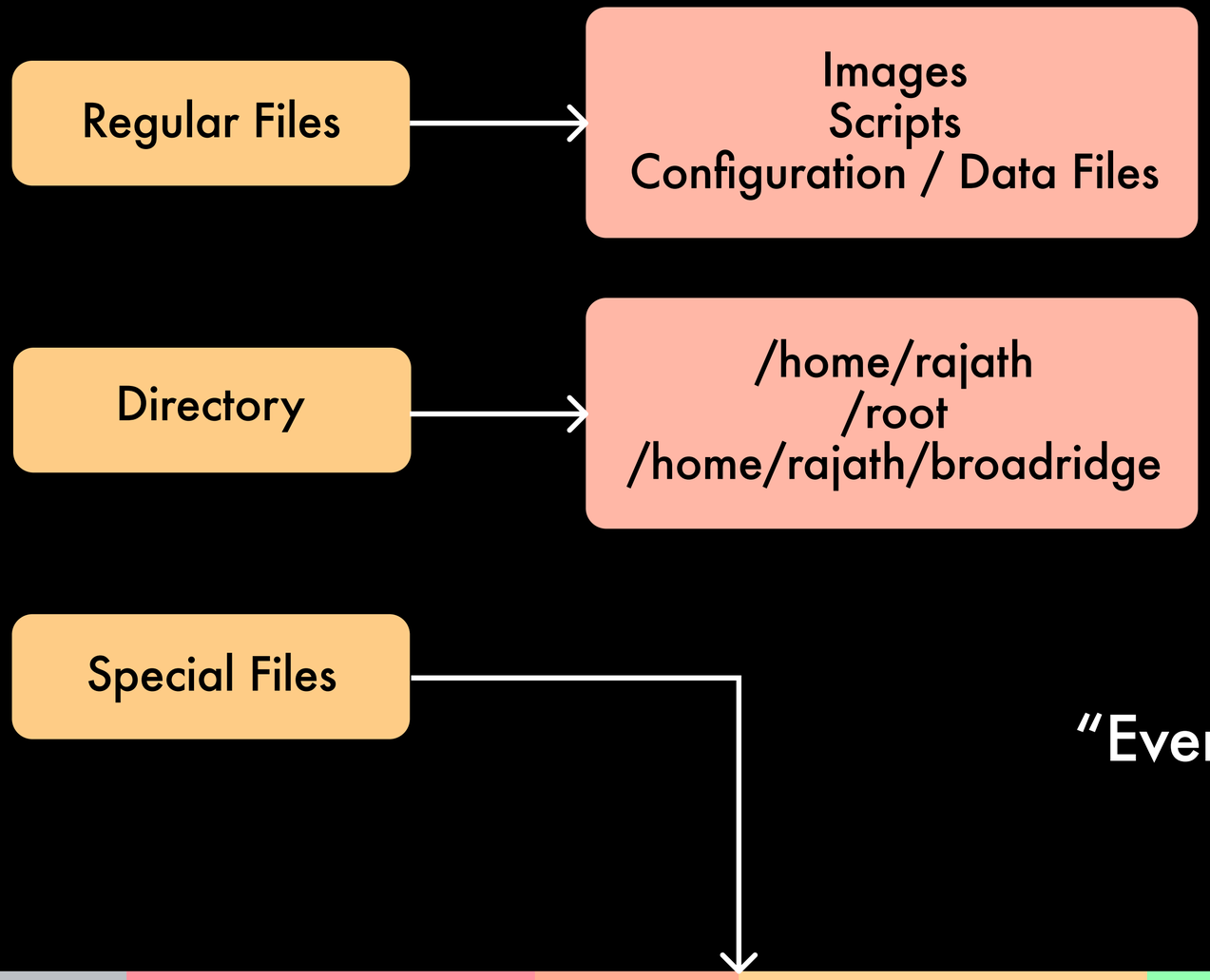
03 Storage & Disk Partitions

04 File Systems in Linux

05 DAS NAS SAN NFS Storage Type & LVM

5.1 File Types in Linux

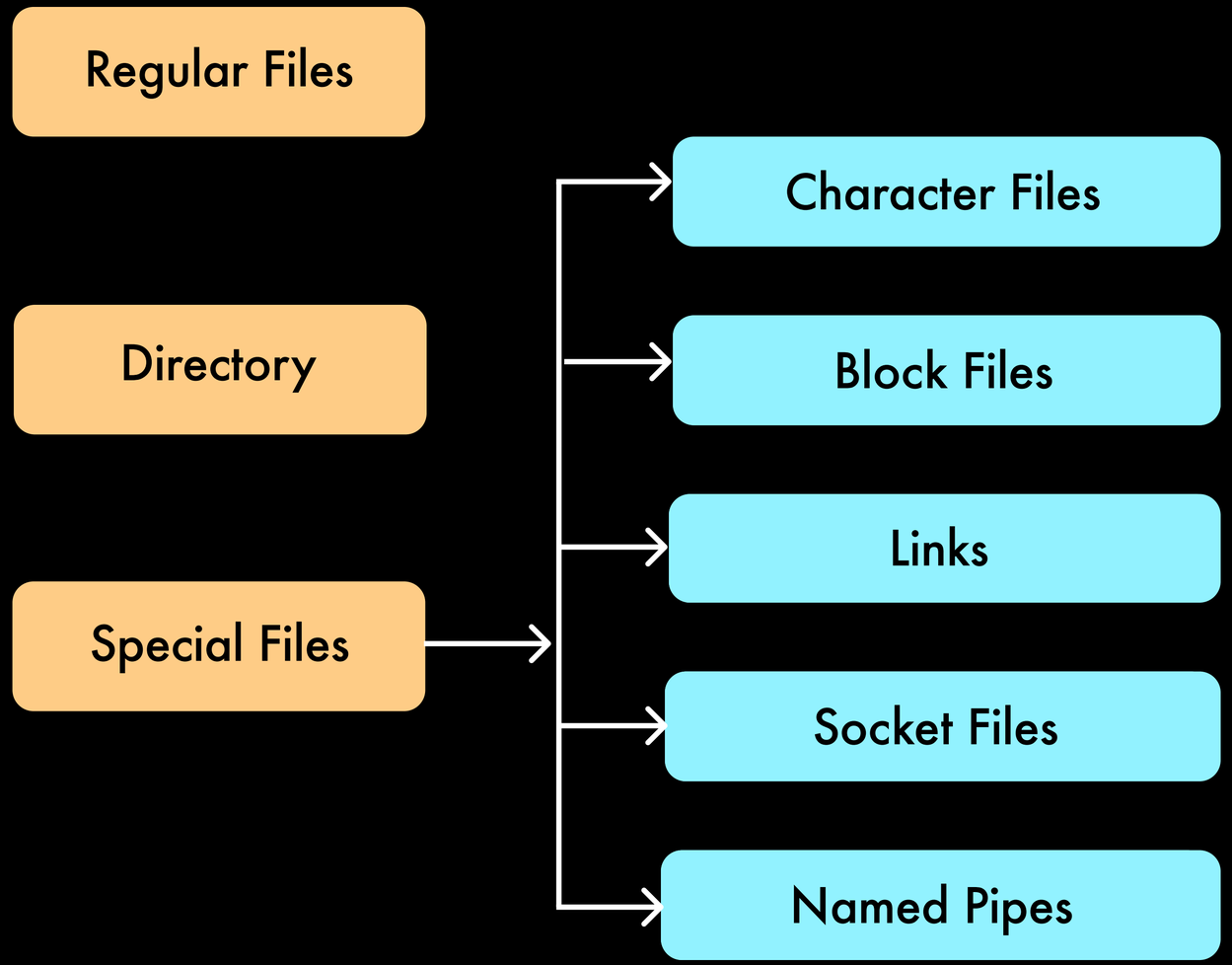
FILE TYPES



"Everything is a file in Linux"

5.1 File Types in Linux

FILE TYPES



That allows kernel to communicate to I/O devices serially, like Keyboard, Mouse..

These represents block devices such as ROM / RAM. Useful to write & read data in chunks

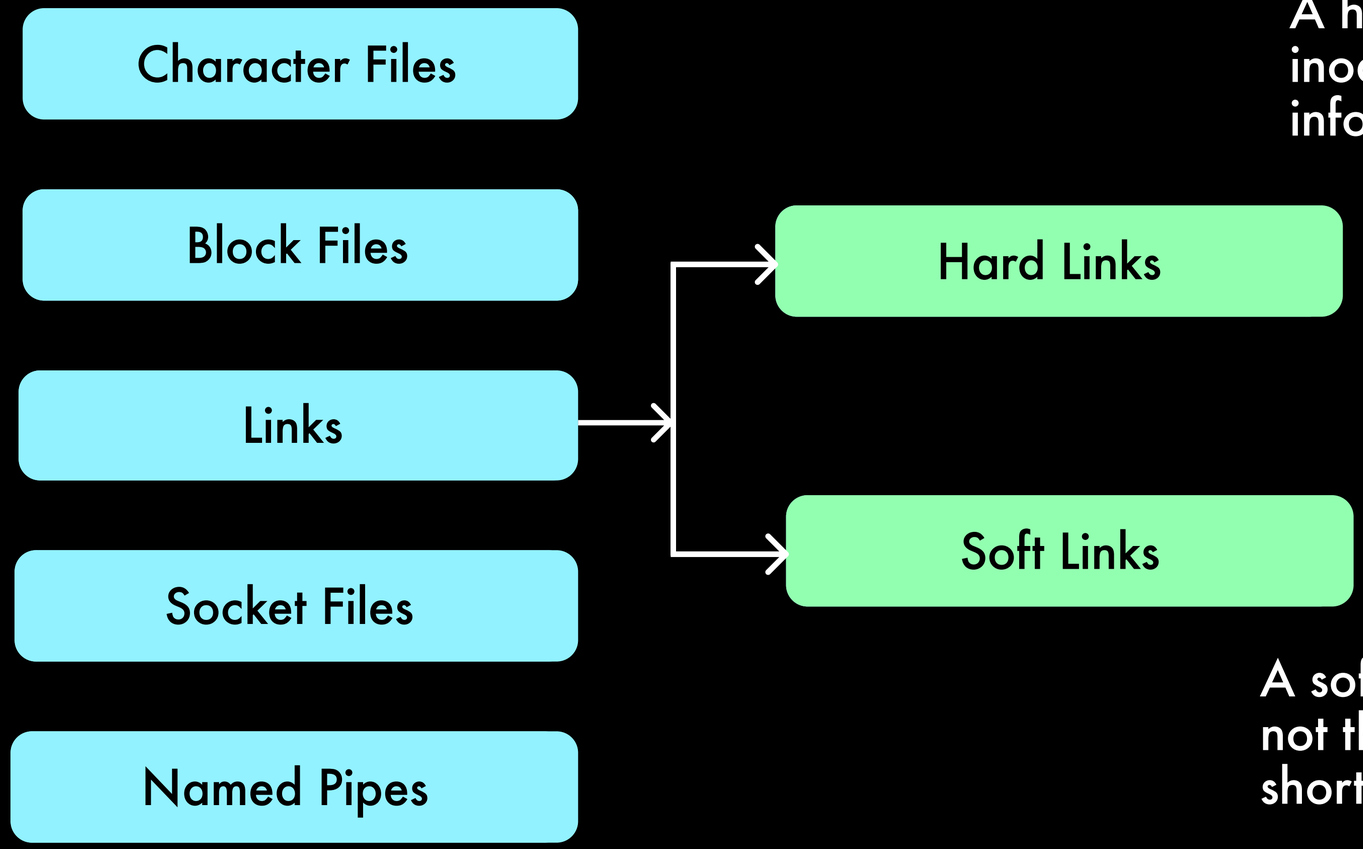
Links in Linux is a way to associate two or more file names to the same set of data.

Socket file is a special file that enabled communication between two processes.

Named Pipes connect one process to another for unidirectional data flow.

5.1 File Types in Linux

FILE TYPES



A hardlink is a direct reference to the file's inode (the data structure that stores information about a file).

A softlink is a reference to the file name, not the inode. It acts as a pointer or shortcut to the file.

5.1 File Types in Linux

FILE TYPES

The **`file`** command can be used to get the information about file type

```
rajath@rhelpro:~/test (0.021s)
file app
app: directory

rajath@rhelpro ~/test (0.035s)
file hello.sh
hello.sh: ASCII text
```

Similarly **`ls -l`** command also can be used.

```
rajath@rhelpro ~/test (0.026s)
ls -l
total 4
drwxr-xr-x 2 rajath rajath  6 Aug  4 16:36 app
-rwxr-xr-x 1 rajath rajath 23 Aug  4 16:37 hello.sh
```

`drwxr-xr-x`: File type and permissions

- **`d`** indicates it's a directory
- **`rwxr-xr-x`** indicates the permissions (read, write, execute for the owner, read and execute for the group, and read and execute for others)

5.1 File Types in Linux

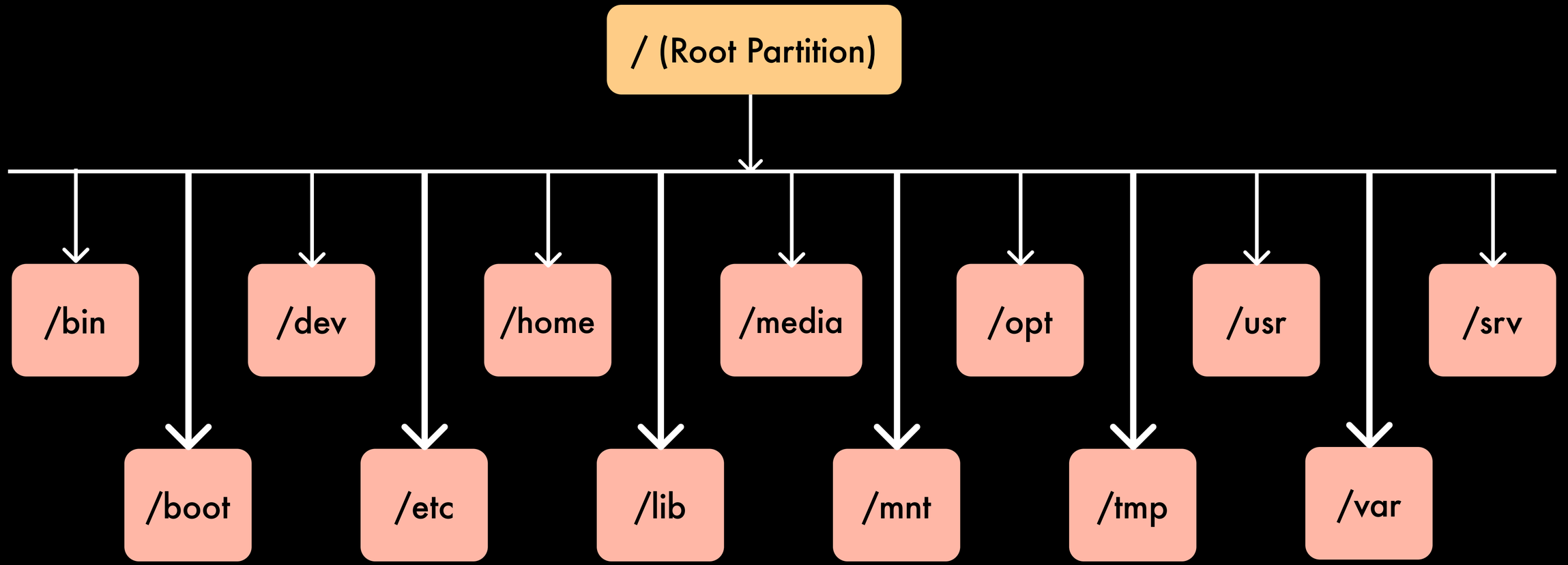
FILE TYPES

```
rajath@rhelpro ~/test (0.026s)
ls -l

total 4
drwxr-xr-x 2 rajath rajath  6 Aug  4 16:36 app
-rwxr-xr-x 1 rajath rajath 23 Aug  4 16:37 hello.sh
```

We can identify the file types by checking the first letter in the ls -l output

File Type	Identifier
Directory	d
Regular File	-
Character Device	c
Link	l
Socket File	s
Pipe	p
Block Device	b

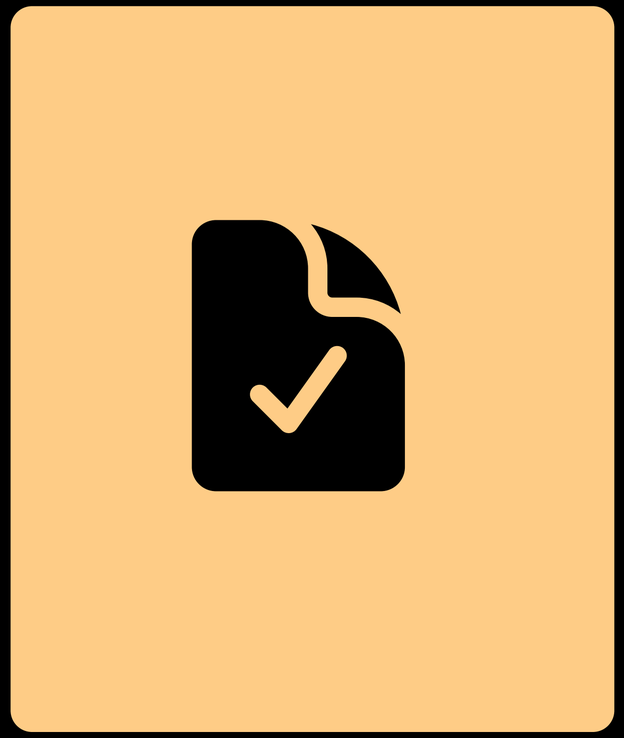
5.2 Hierarchy



HIERARCHY

5.2 Hierarchy

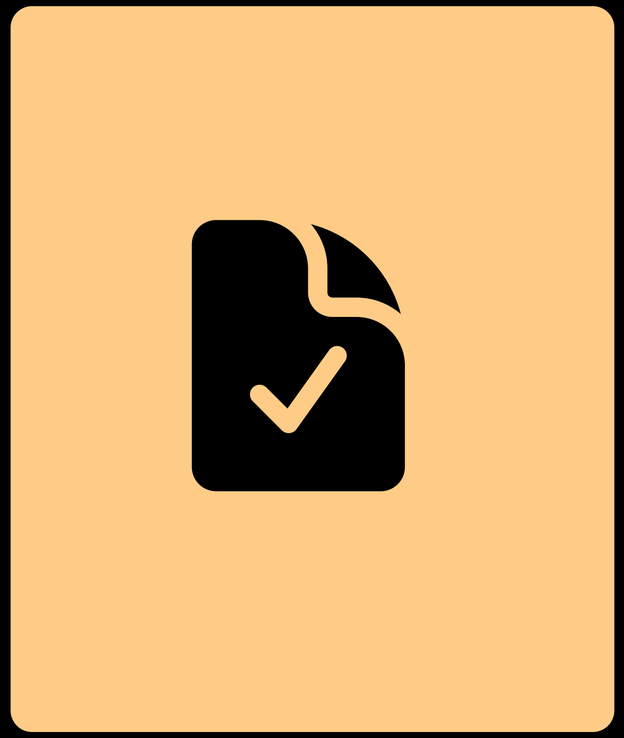
HIERARCHY



/	Root Directory, all other directories stem from here
/bin	Contains essential command-line utilities needed for the system
/boot	Static files for the boot process, such as the kernel and initial RAM, Disk images.
/dev	Device files represent hardware component and peripherals
/etc	Contains System wide configuration files.
/home	Users personal directories
/lib	Shared libraries and kernel modules

5.2 Hierarchy

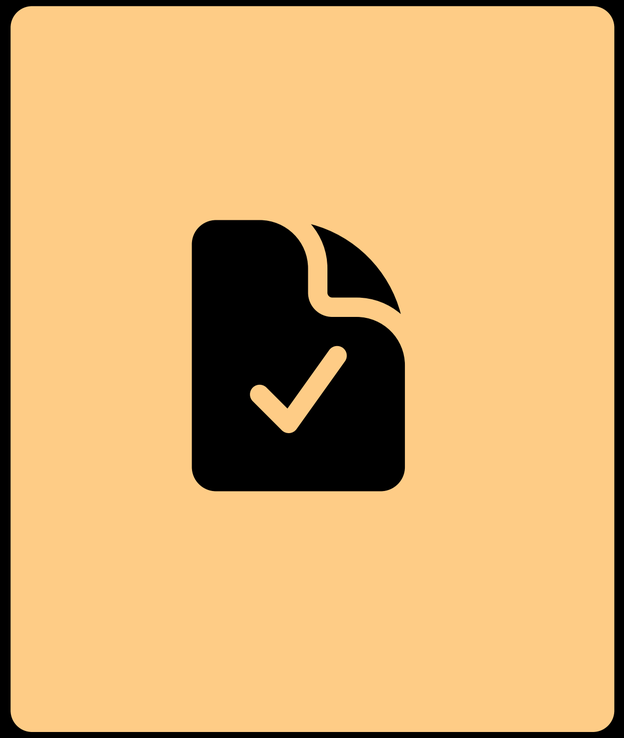
HIERARCHY



/media	Contains mount points for removable media such as USB drives
/mnt	Used for temporarily mounting filesystems.
/opt	Contains add-on application software packages.
/proc	Virtual filesystem with system process and kernel parameter info.
/root	Home directory for the root user.
/run	Contains data relevant to the running system since the last boot
/sbin	Contains essential system binaries used for system administration.

5.2 Hierarchy

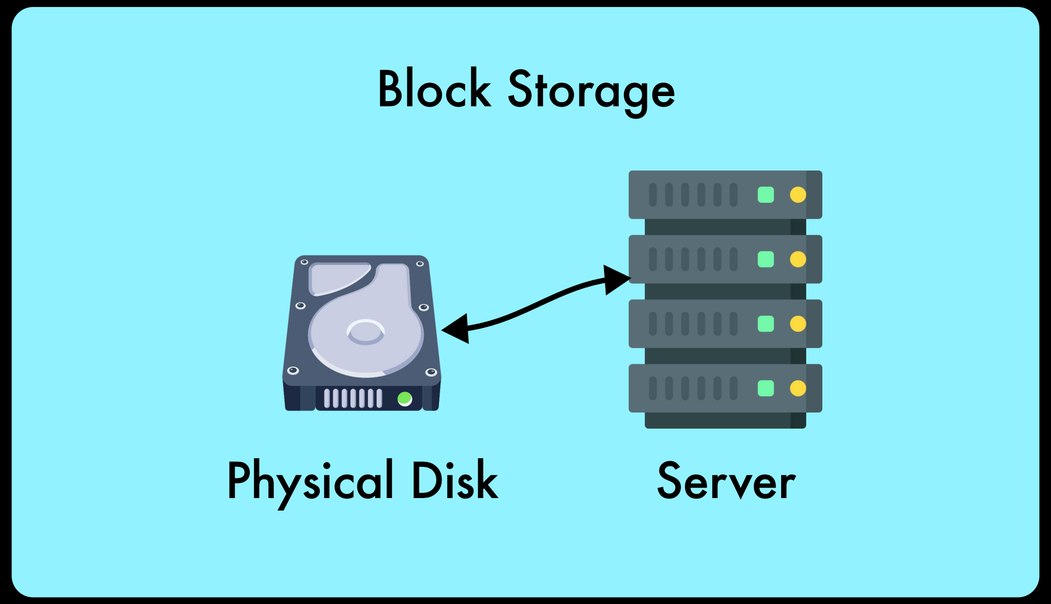
HIERARCHY



/srv	Contains information about the system and kernel, similar to /proc.
/tmp	Temporary files created by users and applications.
/usr	User utilities and applications.
/var	Variable data files such as logs, spool files, and temporary files.
/afs	Andrew File System (AFS), which is a distributed file system that enables cooperative file sharing
/sys	Provides a structured way to access information about devices & drivers.

5.3 Storage and Disk Partitions

STORAGE



Block devices represents a piece of hardware that can store data. Traditional spinning hard disk or solid-state drives are examples of block storage in Linux.

Using **`lsblk`** lists information about block devices.

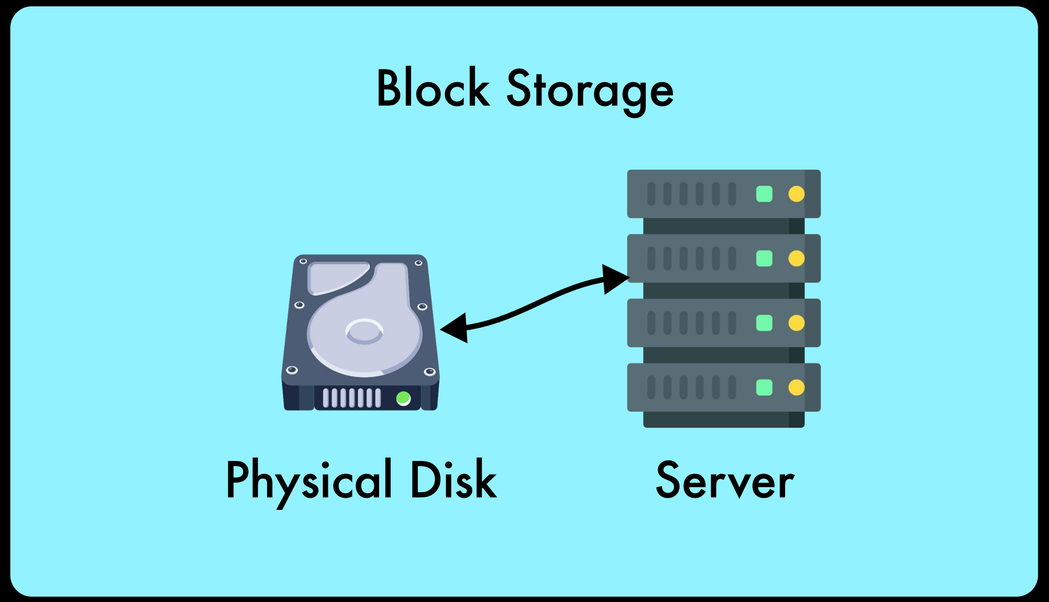
```
rajath@rhelpro ~ (0.036s)
lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda      8:0    0 223.6G  0 disk
├─sda1   8:1    0     1G  0 part /boot
├─sda2   8:2    0   7.8G  0 part [SWAP]
├─sda3   8:3    0    70G  0 part /
├─sda4   8:4    0     1K  0 part
└─sda5   8:5    0 144.8G  0 part /home
```

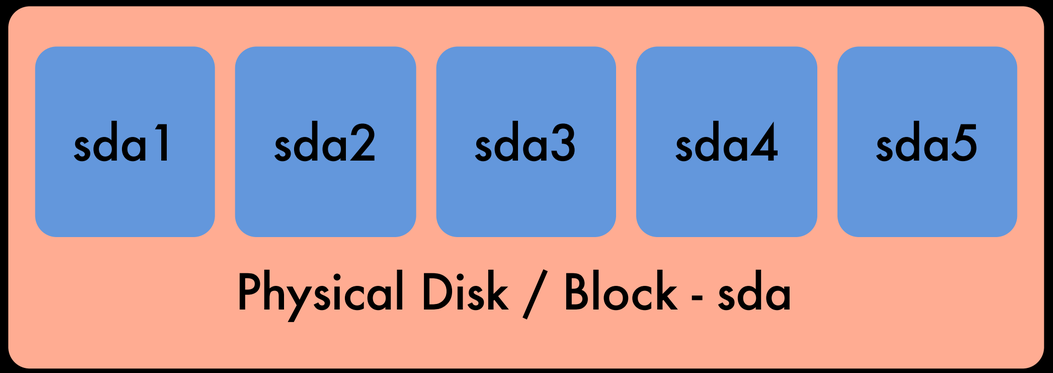
Also **`ls -l`** in **/dev** directory lists information about block devices.

```
rajath@rhelpro ~ (0.028s)
ls -l /dev/ | grep "^b"

brw-rw----  1 root   disk      8,   0 Aug  4 16:31 sda
brw-rw----  1 root   disk      8,   1 Aug  4 16:32 sda1
brw-rw----  1 root   disk      8,   2 Aug  4 16:32 sda2
brw-rw----  1 root   disk      8,   3 Aug  4 16:32 sda3
brw-rw----  1 root   disk      8,   4 Aug  4 16:32 sda4
brw-rw----  1 root   disk      8,   5 Aug  4 16:32 sda5
```

5.3 Storage and Disk Partitions

STORAGE






```
rajath@rhelpro ~ (0.036s)
lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda      8:0    0 223.6G  0 disk
├─sda1   8:1    0     1G  0 part /boot
├─sda2   8:2    0   7.8G  0 part [SWAP]
├─sda3   8:3    0    70G  0 part /
├─sda4   8:4    0     1K  0 part
└─sda5   8:5    0 144.8G  0 part /home
```

Major Number	Device Type
1	RAM or USB
3	HARD DISK or CD ROM
6	PARALLEL PRINTERS
8	SCSI DISK

5.3 Storage and Disk Partitions

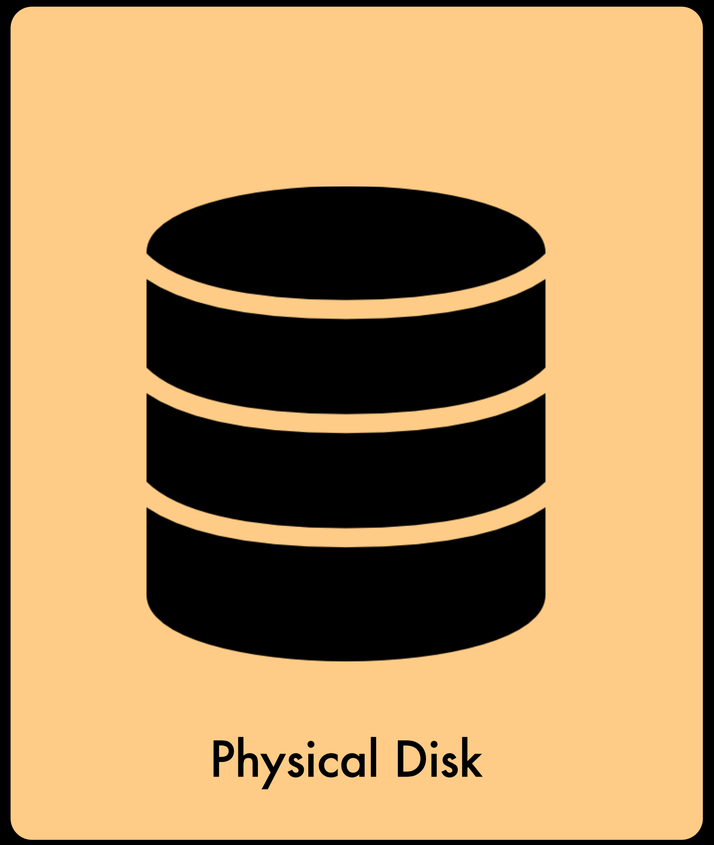
STORAGE

To display information about the partition table we can use **`fdisk -l`** command.

```
rajath@rhelpro ~ (2.475s)
sudo fdisk -l /dev/sda

[sudo] password for rajath:
Disk /dev/sda: 223.57 GiB, 240057409536 bytes, 468862128 sectors
Disk model: KINGSTON SA400S3
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xa9e6f836

Device     Boot     Start       End   Sectors   Size Id Type
/dev/sda1  *         2048   2099199   2097152     1G 83 Linux
/dev/sda2         2099200  18448383  16349184   7.8G 82 Linux swap /
/dev/sda3        18448384 165249023 146800640    70G 83 Linux
/dev/sda4       165249024 468860927 303611904 144.8G  f W95 Ext'd (LB
/dev/sda5       165251072 468860927 303609856 144.8G 83 Linux
```

5.3 Storage and Disk Partitions

DISK PARTITIONS



Primary

The primary partition is a type of partition that can be used to boot an operating system.

Extended

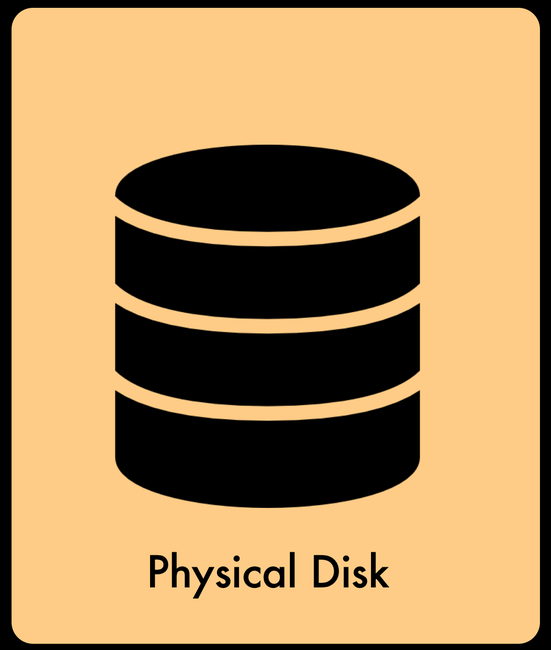
An Extended partition is like a disk drive in its own right. It has a partition table that points one or more logical partitions.

Logical

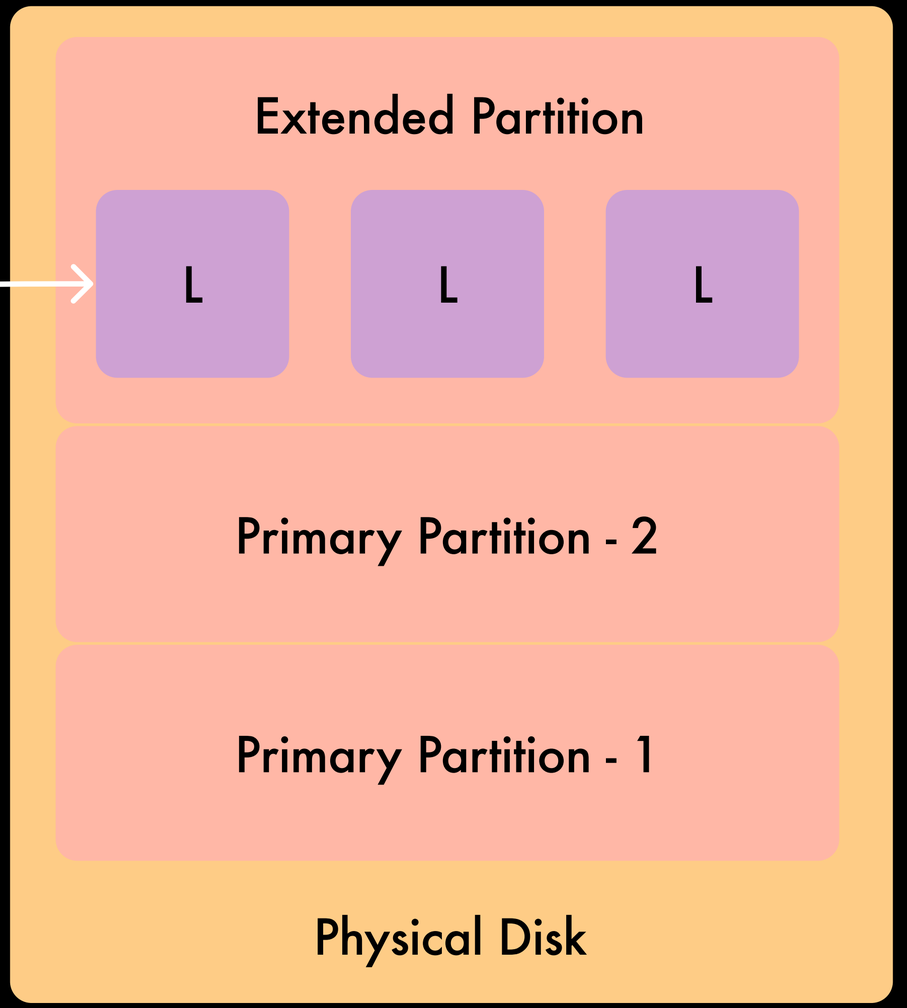
Logical partitions are those created within an extended partition.

5.3 Storage and Disk Partitions

DISK PARTITIONS



Logical Partition



Partition Scheme / Table

Disk Partition is defined by a partitioning scheme also known as partition table

5.3 Storage and Disk Partitions

DISK PARTITIONS

Master Boot Record (MBR)

The Master Boot Record (MBR) is the first sector of a storage device (usually a hard disk) that contains the bootloader and the partition table.

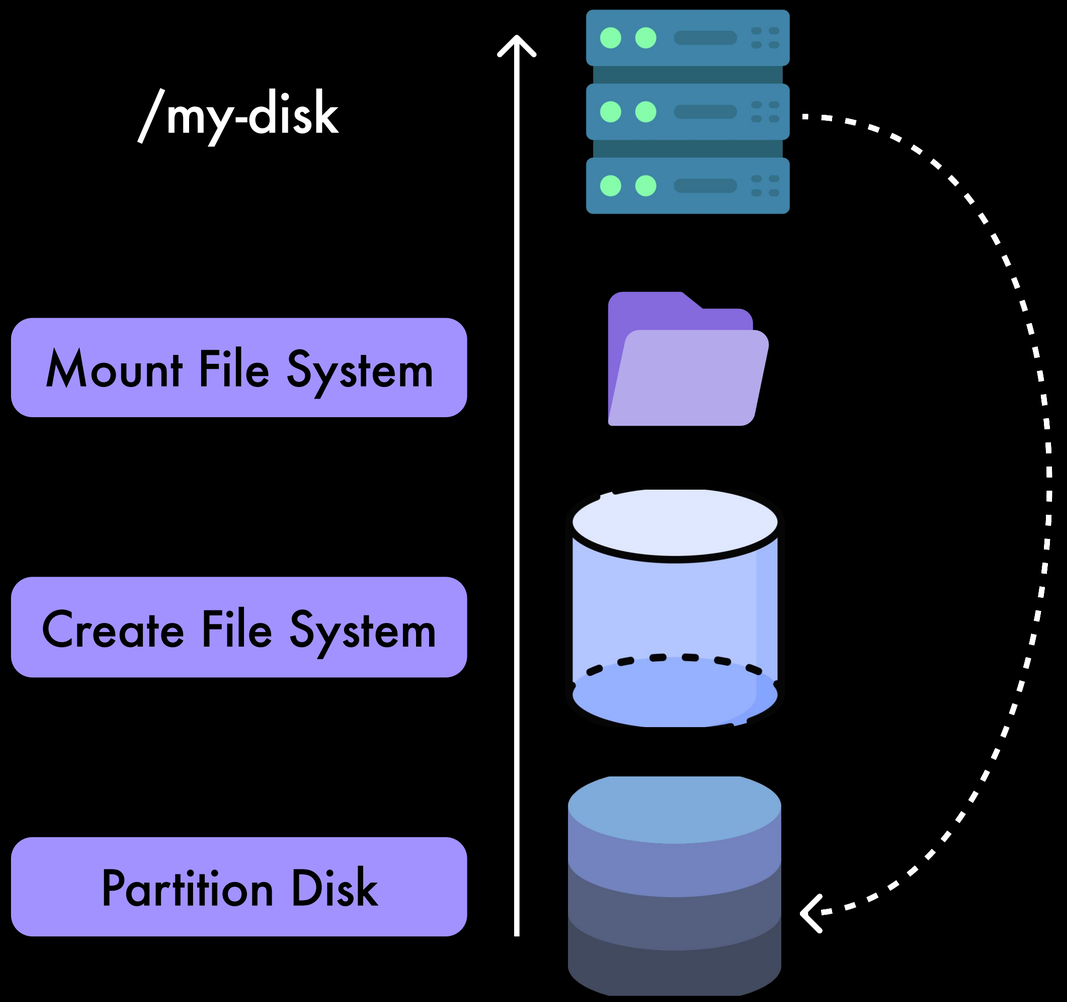
GUID Partition Table (GPT)

The GUID Partition Table (GPT) is a standard for the layout of the partition table on a physical storage device, using globally unique identifiers (GUID).

Feature	MBR	GPT
Partition Limit	4 Primary Partitions	Up to 128 Partitions
Disk Size Limit	Up to 2 TB	Over 2 TB
Redundancy	No	Yes (Backup Header)
Boot Compatibility	BIOS	UEFI (Unified Extensible Firmware Interface)

5.4 File Systems in Linux

FILE SYSTEMS



The File System defines the structure for storing data on a disk. Once a filesystem is created, a directory can be mounted, allowing for data read and write operations.

5.4 File Systems in Linux

FILE SYSTEMS

Commonly Used Linux File System

1. EXT4 (Fourth Extended Filesystem)

   The default file system for many linux distributions.

   - Supports large file sizes and volumes.
   - Extents for efficient storage.

   General purpose use, suitable for most Linux Installations

2. XFS (Extended Filesystem)

   High performance 64-bit Journaling file System.

   - Scalable and efficient with large files.
   - Online defragmentation and resizing.
   - Metadata journaling for data integrity.

   Ideal for high-performance workloads, large data sets, and servers.

5.4 File Systems in Linux

FILE SYSTEMS

Commonly Used Linux File System

3. BTRFS (B-Tree File System)

Modern file system with advanced features.

- Copy-on-write (COW) for snapshots and clones.
- Integrated volume management.
- Built-in RAID support.

Data Management, Snapshots and Dynamic Disk Configurations

4. F2FS (Flash-Friendly File System)

Designed specifically for NAND flash memory.

- Optimized for SSDs and flash storage.
- Reduces write amplification.
- Enhances performance and lifespan of flash devices.

Mobile devices and flash-based storage.

5.4 File Systems in Linux

FILE SYSTEMS

Commonly Used Linux File System

5. ZFS (Zettabyte File System)

   Advanced file system originally developed by Sun Microsystems.

   - High storage capacities.
   - Data integrity verification and repair.
   - Support for snapshots and clones.

   High-availability systems, storage servers, and data integrity-critical applications

To display information about the block devices, including file system types we can use **`lsblk -f`** command.

```
rajath@rhelpro ~ (0.029s)
lsblk -f
NAME   FSTYPE FSVER LABEL UUID                                 FSAVAIL
sda
├─sda1 xfs                 0e53c296-e2c9-4368-8d13-953c3c7b8efb  532.7M
├─sda2 swap   1            f112ba76-8ac0-459f-beaa-db9f1dd2bf0d
├─sda3 xfs                 98da070a-6052-4821-94e7-0ef8e29d14d0   61.6G
├─sda4
└─sda5 xfs                 855de843-48b0-4ea7-910e-fb285ea6bf5c  141.7G
```

5.5 Storage Types

STORAGE TYPES

DAS, NAS and SAN

DAS = Direct Attached Storage

Storage that is directly connected to a single computer or server without a network in between.

NAS = Network Attached Storage

A dedicated file storage device that provides shared storage over a network using file-based protocols.

SAN = Storage Area Network

A high-speed network that connects multiple servers to a centralized pool of block-level storage.

5.5 Storage Types

STORAGE TYPES

NFS

NFS = Network File System

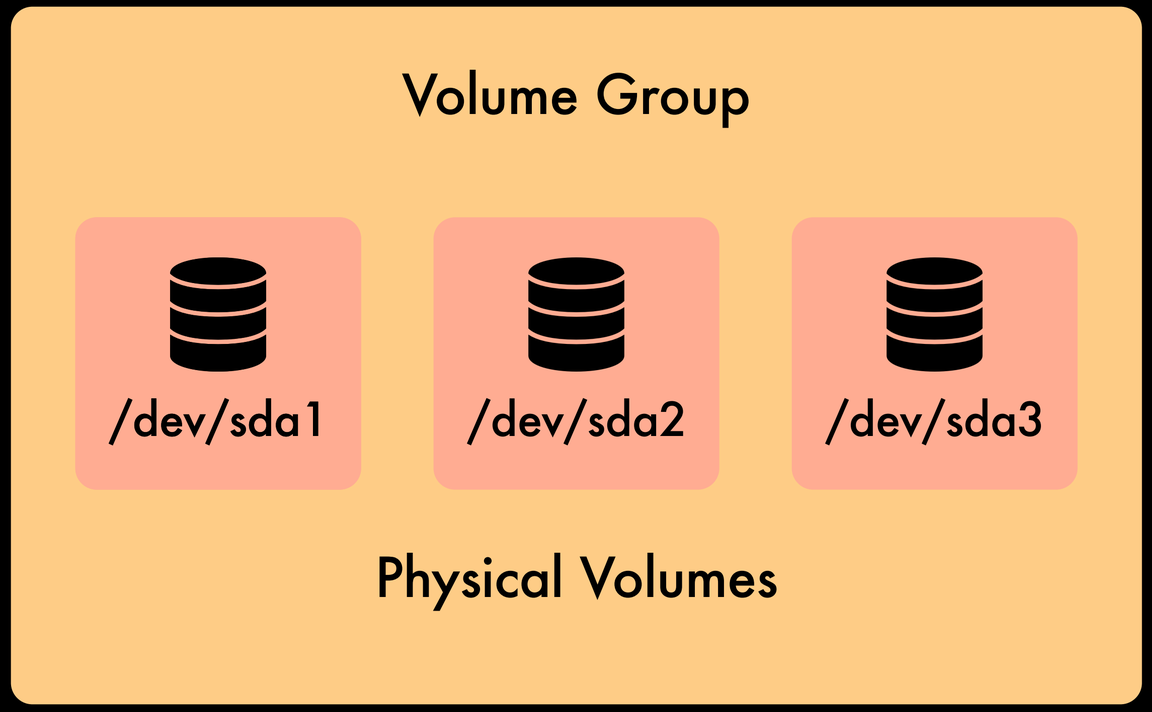
Network File System (NFS) is a distributed file system protocol that allows users to access files over a network as if they were located on their own local storage.

- NFS allows multiple clients to access the same files simultaneously over a network.
- Provides a way to centralize data storage and make it accessible from multiple locations.
- Supports various authentication and access control mechanisms, including UNIX file permissions and access control lists (ACLs).
- Files accessed via NFS appear as if they are part of the local file system, providing a seamless user experience.
- NFS is platform-independent, allowing file sharing between different types of systems (e.g., Linux, UNIX, and even Windows with appropriate software).

5.5 Storage Types

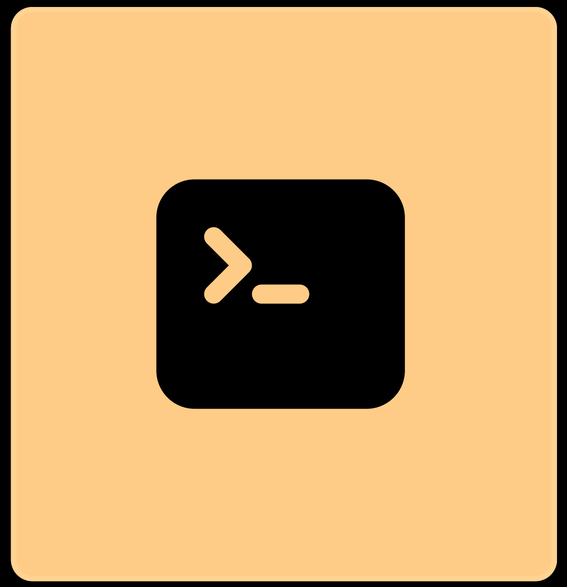
LOGICAL VOLUME

LVM = Logical Valume Manager



Logical Volume Manager (LVM) is a powerful storage management system in Linux that allows for flexible disk management. It abstracts the physical storage devices to provide more flexible allocation and management of storage resources.

6 Shell and Bash



01 Shell

02 Different Types of Shell

03 Bash

04 Command Line Help

05 Bash Script

6.1 Shell

A shell is a command-line interpreter that provides a user interface for accessing the services of the operating system. It allows users to execute commands, scripts, and programs, and it also provides the means for managing files and processes.

6.2 Different Types of Shell

SHELL & BASH

1. Bourne Shell (sh):

- The original Unix shell developed by Stephen Bourne.
- Known for its simplicity and scripting capabilities.

2. Bash (Bourne Again Shell):

- Enhanced version of Bourne Shell by Brian Fox for GNU Project.
- Default in many Linux distros.

6.2 Different Types of Shell

3. C Shell (csh):

- Developed by Bill Joy, inspired by the C programming language.
- Known for its scripting syntax that resembles C.

4. Korn Shell (ksh):

- Developed by David Korn, combining Bourne and C shell features.
- Known for powerful scripting.

6.2 Different Types of Shell

5. Z Shell (zsh):

- Customizable shell with features from Bash, ksh, and tcsh.
- Known for interactive use like tab completion and command history.

6. Fish (Friendly Interactive Shell):

- A user-friendly shell with many interactive features.
- Known for its intuitive syntax and usability.

6.3 Bash (Bourne Again Shell)

Bash is one of the most popular and widely used shells in Linux. It combines features from the Bourne Shell (sh) and the C Shell (csh) to provide a comprehensive and powerful shell environment.

1. Command Execution
2. Scripting
3. Command History
4. Tab Completion
5. Job Control
6. Environment Customization
7. Redirection and Pipelines
8. BuiltIn Commands & Utilities

6.4 Command Line Help

SHELL & BASH

To get to understand about the command you can use **`whatis`** command. It'll return one line information about the command.

```
rajath@rhelpro ~ (0.028s)
whatis date

date (1)              - print or set the system date and time
date (1p)             - write the date and time
```

```
rajath@rhelpro ~ (0.057s)
man date | col -b | cat

DATE(1)                                          User Commands                                          DATE(1)

NAME
       date - print or set the system date and time

SYNOPSIS
       date [OPTION]... [+FORMAT]
       date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]
```

Most of the commands, internal or external come bundled with man pages.

These provide information about the command in detail, often with examples.

6.4 Command Line Help

SHELL & BASH

Several commands provide a “- h” or a “- - help” option to provide users with the options and use cases available.

```
rajath@rhelpro ~ (0.026s)
date --help

Usage: date [OPTION]... [+FORMAT]
  or:  date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]
Display the current time in the given FORMAT, or set the system date.
```

This handy little command will search through the man page names and descriptions for instances of a keyword.

```
rajath@rhelpro ~ (0.077s)
apropos modpr

modprobe (8)          - Add and remove modules from the Linux Kernel
modprobe.conf (5)     - Configuration directory for modprobe
modprobe.d (5)        - Configuration directory for modprobe
```

6.5 Bash Script

SHELL & BASH

```
rajath@rhelpro ~/test (0.025s)
cat hello.sh
#!/bin/bash

# This is a simple Bash script example

# Define a variable
name="Broadridge"

# Function to print a greeting
greet() {
  echo "Hello, $name!"
}

# Call the function
greet
```

```
rajath@rhelpro ~/test (0.021s)
./hello.sh
Hello, Broadridge!
```

A Bash script is a plain text file containing a series of commands that are executed by the Bash shell, a popular command-line interpreter used in Unix-like operating systems.

7. Deep Dive into Linux Commands

Let’s Unleash the Power of the Command Line"!