Lead Training

2nd Batch 2025

Date: 16/07/2025

DOYENSYS
VALUE INNOVATION

DIL
Doyensys Institute of Learning

Table of contents

01 Linux Basic commands

02 Cronjobs and shell scripts basics

03 Introduction to database – Oracle Architecture

04 Getting started with oracle server

05 Managing an oracle instance

06 Database creation

07 Dictionary & Dynamic Performance views

Linux Basics OS commands

Linux Basics and OS commands

Introduction to Linux

Linux is based on the Linux kernel, which is a monolithic, Unix-like kernel. It serves as the core of the operating system, managing system resources, hardware, and providing essential services.

Linux offers a wide range of desktop environments and window managers, such as GNOME, KDE, Xfce, and others. The user interface can be highly customizable and can vary between different Linux distributions.

Linux is known for its versatility and runs on a variety of architectures, including x86, x86_64, ARM, PowerPC, and more. This makes it suitable for a wide range of devices, from servers and desktops to embedded systems.

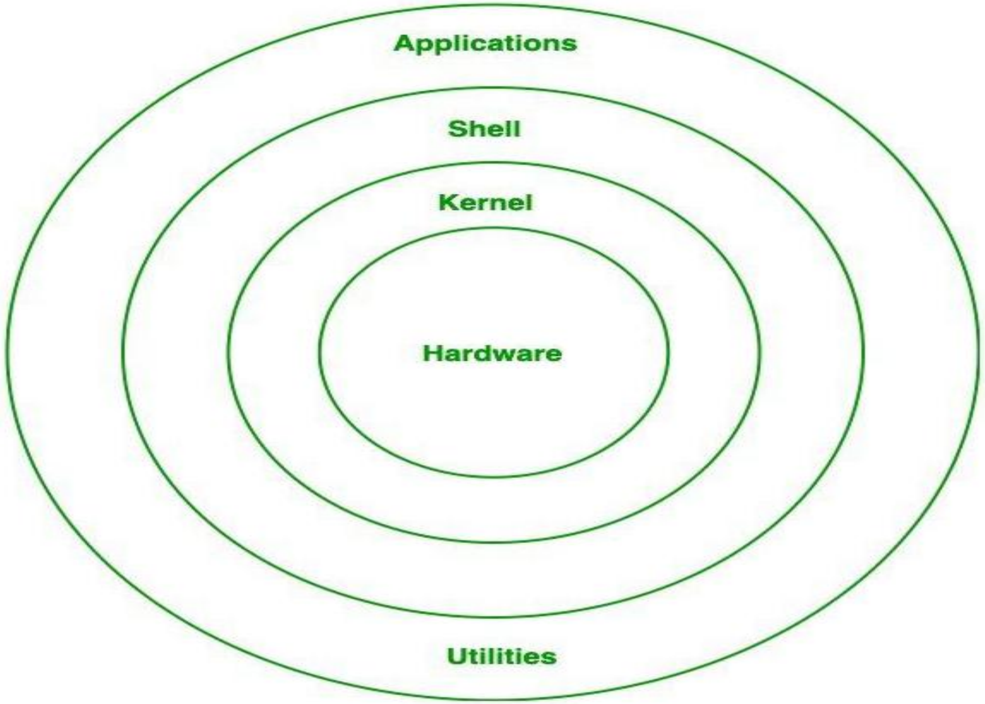
Linux is open-source, meaning the source code is freely available, and users can modify and distribute their versions under the terms of the GNU General Public License (GPL) or other open-source licenses.

Linux has a powerful and extensive command-line interface (CLI). Many tasks can be performed efficiently through the terminal, and scripting is a common practice.

Most Linux distributions use package managers (e.g., apt, yum, pacman) to install, update, and remove software. This simplifies software management and dependency resolution.

Linux Architecture

The architecture of Linux encompasses various components and layers that work together to provide the functionality of the operating system. Here's an overview of the key components in the Linux architecture:



Linux Architecture

Kernel:

At the core of the Linux architecture is the **Linux kernel**. It's responsible for managing system resources, providing essential services, and interacting with hardware components. The kernel plays a crucial role in the overall stability and performance of the operating system.

Shell:

The **shell** is the user interface to the operating system. It interprets user commands and executes them by interacting with the kernel. Linux supports various shells, including Bash (Bourne Again SHell), which is one of the most widely used.

Utilities and Commands:

Linux provides a vast set of **utilities and commands** that users can execute from the shell. These include commands for file manipulation (ls, cp, mv, rm), text processing (grep, sed, awk), and system administration (ps, top, chmod).

System Calls:

Applications interact with the kernel through **system calls**. These are functions or interfaces that allow applications to request services from the kernel, such as file operations, process management, and communication with hardware.

File System Hierarchy

The File System Hierarchy Standard (FHS) defines the structure and organization of files and directories in a Unix-like operating system. Linux adheres to this standard to ensure consistency across distributions. Here's an overview of the key directories in the Linux file system hierarchy:

1. **/ (Root Directory):**
   - The root directory is the top-level directory in the file system hierarchy.
   - All other directories and files are contained within the root directory.
2. **/bin (Binaries):**
   - Essential binary executable files (commands) required for system boot and repair.
   - Common commands like ls, cp, mv, rm, etc., are located here.
3. **/boot (Boot Loader Files):**
   - Contains files needed for the system boot process, including the kernel, initrd (initial RAM disk), and boot loader configuration files.
4. **/dev (Device Files):**
   - Device files representing hardware devices are stored here.
   - Examples include /dev/sda for the first SATA hard drive and /dev/tty for terminal devices.
5. **/etc (Configuration Files):**
   - Configuration files and directories used by system-wide software and services.
   - System-wide configuration files for various applications are stored here.
6. **/home (Home Directories):**
   - Home directories for user accounts are located here.
   - Each user typically has a subdirectory in /home with their username.

File System Hierarchy

7./lib and /lib64 (Libraries):

- Essential shared libraries and kernel modules required for system boot and operation.
- /lib contains 32-bit libraries, while /lib64 contains 64-bit libraries on some systems.

8./media and /mnt (Removable Media Mount Points):

- Directories for mounting removable media, such as USB drives or CD-ROMs.
- /media is often used for automatic mounting, while /mnt is a generic mount point for manual mounting.

9./opt (Optional Software Packages):

- Typically used for installing additional software packages.
- Third-party applications may be installed in subdirectories of /opt.

10./proc (Process Information):

- A virtual file system that provides information about processes and system resources.
- Information is presented in a file-like structure and can be accessed to retrieve real-time data.

11./root (Root User's Home Directory):

- The home directory for the root user.

File System Hierarchy

12./sbin **(System Binaries):**

- Essential system binaries used for system administration and maintenance.
- Commands like fdisk, ifconfig, and iptables are often located here.

13./srv **(Service Data):**

- Data for services provided by the system.
- Subdirectories may be created for specific services, such as /srv/www for web server data.

14./tmp **(Temporary Files):**

- A directory for temporary files used by applications and users.
- Files in this directory are typically deleted upon system reboot.

15./usr **(User Binaries and Data):**

- Secondary hierarchy for user data, applications, and libraries.
- It includes subdirectories like /usr/bin, /usr/lib, /usr/include, etc.

16./var **(Variable Data):**

- Variable data that may change during the course of system operation.
- Includes directories like /var/log for log files and /var/spool for print and mail spools.

Useful OS Commands

File System Commands:

- **ls**: List files and directories.
  - Example: ls -l
- **cd**: Change directory.
  - Example: cd /path/to/directory
- **pwd**: Print working directory.
  - Example: pwd
- **cp**: Copy files or directories.
  - Example: cp file.txt /destination/directory
- **mv**: Move or rename files or directories.
  - Example: mv file.txt newname.txt
- **rm**: Remove or delete files or directories.
  - Example: rm file.txt

File and Text Manipulation Commands:

- **cat**: Display the content of a file.
  - Example: cat file.txt
- **grep**: Search for a pattern in files.
  - Example: grep "pattern" file.txt
- **sed**: Stream editor for text transformation.
  - Example: sed 's/old/new/' file.txt
- **awk**: Pattern scanning and text processing tool.
  - Example: awk '{print $1}' file.txt

Useful OS Commands

User and Permission Management Commands:

- **useradd**: Add a new user.
  - Example: useradd username
- **passwd**: Change user password.
  - Example: passwd username
- **chown**: Change file or directory ownership.
  - Example: chown user:group file.txt
- **chmod**: Change file permissions.
  - Example: chmod 755 file.txt

Process Management Commands:

- **ps**: Display information about active processes.
  - Example: ps aux
- **kill**: Terminate a process.
  - Example: kill -9 PID
- **top**: Display real-time system statistics.
  - Example: top

System Information Commands:

- **uname**: Display system information.
  - Example: uname -a
- **df**: Display disk space usage.
  - Example: df -h
- **free**: Display system memory usage.
  - Example: free -m

Useful OS Commands

Network and Connectivity Commands:

- **ping**: Test network connectivity.
  - Example: ping example.com
- **netstat**: Display network statistics.
  - Example: netstat -an
- **traceroute**: Display the route that packets take.
  - Example: traceroute example.com

Directory Commands:

- **mkdir - Create a directory:**
  - Example: mkdir new_directory
- **rmdir - Remove an empty directory:**
  - Example: rmdir empty_directory
- **find - Search for files and directories:**
  - Example: find /path/to/search -name "filename"
- **tree - Display directory structure as a tree:**
  - Example: tree

File Commands:

- **more/less - Display file content one screen at a time:**
  - Example: more file.txt, less file.txt
- **head - Display the first few lines of a file:**
  - Example: head file.txt
- **tail - Display the last few lines of a file:**
  - Example: tail file.txt
- **touch - Create an empty file or update file timestamps:**
  - Example: touch newfile.txt
- **nano/vim/emacs - Text editors for file editing:**
  - Example: nano filename, vim filename, emacs filename
- **diff - Compare two files:**
  - Example: diff file1.txt file2.txt
- **grep - Search for a pattern in files:**
  - Example: grep "pattern" file.txt
- **stat - Display file or file system status:**
  - Example: stat file.txt

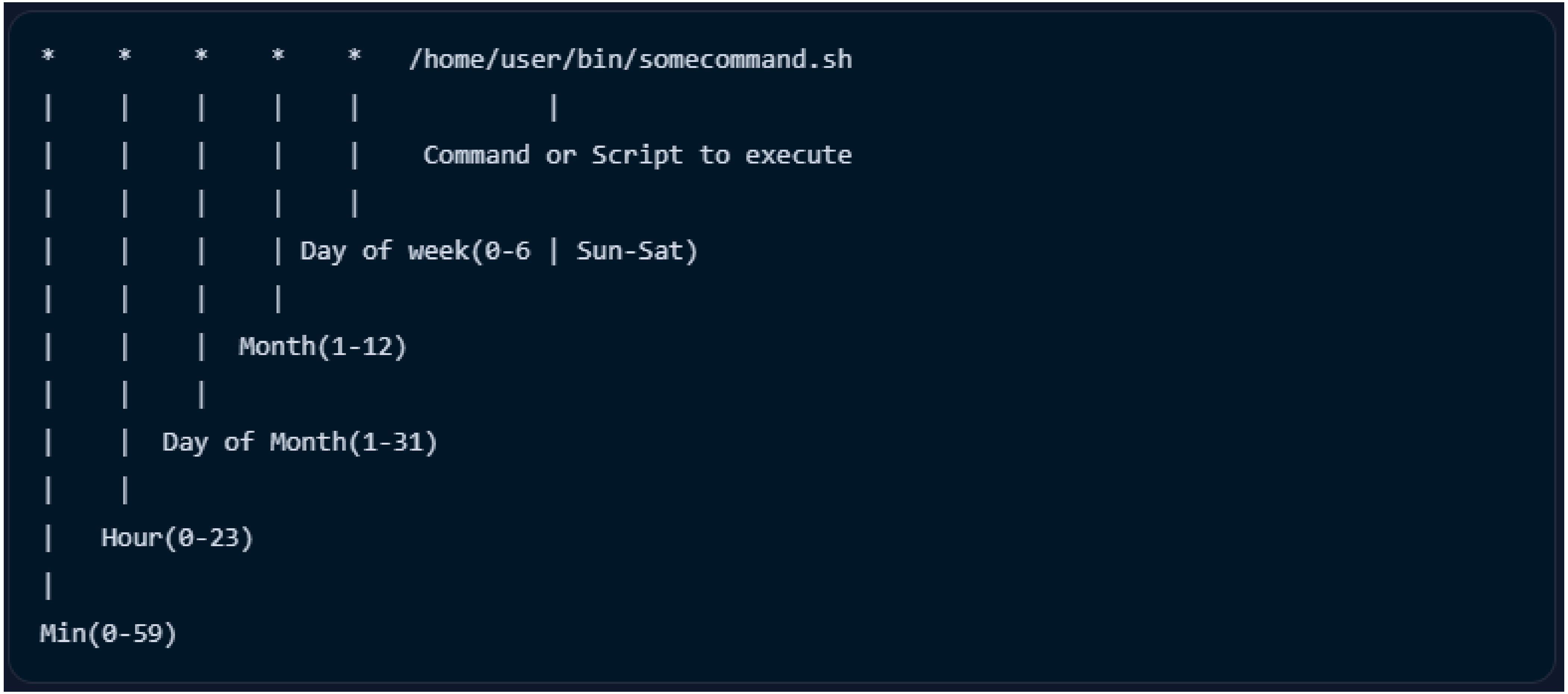
Useful OS Commands

- **ln -s - Create a symbolic link:**
  - Example: ln -s /path/to/target link_name
  - This creates a symbolic link named link_name that points to the file or directory at /path/to/target.
- **readlink - Display the target of a symbolic link:**
  - Example: readlink link_name
  - Shows the actual path to which the symbolic link points.
- **ln - Create a hard link:**
  - Example: ln /path/to/target link_name
  - Creates a hard link named link_name to the file at /path/to/target.
- **ls -i - Display the inode number of a file:**
  - Example: ls -i file.txt
  - Inodes are unique identifiers for files. Hard links share the same inode.
- **unlink - Remove a link:**
  - Example: unlink link_name
  - Removes the link but doesn't delete the target file.
- **file - Determine file type:**
  - Example: file filename
  - Helps identify whether a given file is a symbolic link or a regular file.

DOYENSYS
VALUE INNOVATION

Crontab jobs & shell script basics

Crontab Job Schedule Syntax



```
*    *    *    *    *   /home/user/bin/somecommand.sh
|    |    |    |    |            |
|    |    |    |    |    Command or Script to execute
|    |    |    |    |
|    |    |    | Day of week(0-6 | Sun-Sat)
|    |    |    |
|    |    |  Month(1-12)
|    |    |
|    |  Day of Month(1-31)
|    |
|   Hour(0-23)
|
Min(0-59)
```

Shell scripting basics

Bash shell scripting:-

Shell:-
command language interpreter, for the GNU operating system
cat /etc/shells

sh - bourn shell original shell used in unix, first shell
bash - bourn again shell was reinvented, improved version of sh.