23CA103 - UNIX ARCHITECTURE AND PROGRAMMING

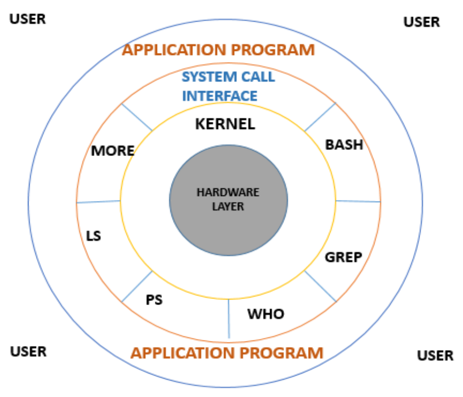
UNIT-1 INTRODUCTION

Unix Overview

Unix is a powerful, multiuser, multitasking operating system originally developed in the 1970s. It's known for its stability, flexibility, and the philosophy of building small, modular tools that do one thing well.

- **Multiuser:** Multiple users can work on the system simultaneously without interfering with each other.
- **Multitasking:** Runs multiple processes at the same time.
- **File System:** Everything is treated as a file—from documents to hardware devices.
- **Shell:** The command-line interface that interprets user commands; Unix has several shells like Bourne Shell (sh), C Shell (csh), and Bourne Again Shell (bash).
- **Portability:** Unix is designed to be portable across different hardware platforms.
- **Security:** Uses permissions and ownership to control access.
- **Tools and Utilities:** Built on a collection of small, modular programs that can be combined in powerful ways (pipes, filters, etc.).
- **Process Management:** Allows users and the system to manage running programs efficiently.

Unix Structure

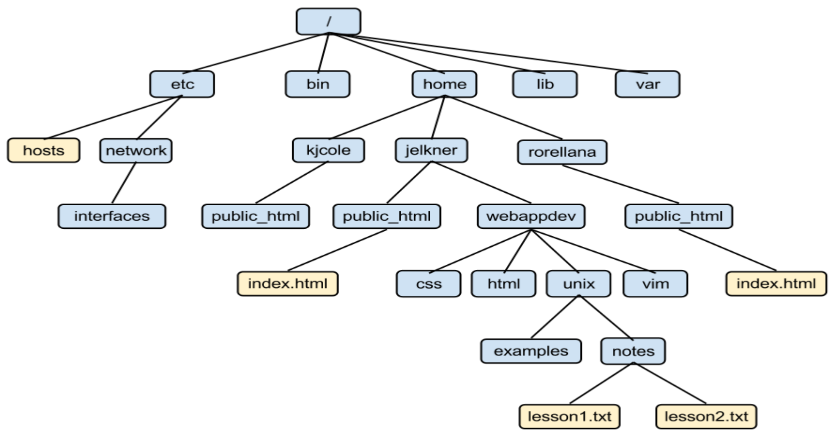


UNIX ARCHITECTURE

Unix is built with a clear, layered structure that keeps things organized and flexible. Its main components include:

- **Kernel:** The heart of Unix. It manages hardware resources, process scheduling, memory, file systems, and device control. It acts as the bridge between the hardware and user applications.
- **Shell:** The command interpreter that provides the user interface. It reads user commands, interprets them, and passes them to the kernel for execution. Common shells include the Bourne Shell (sh), C Shell (csh), and Bash (bash).
- **File System:** A hierarchical organization of files and directories, starting from the root (/). Everything in Unix is treated as a file, including hardware devices and processes.
- **Processes:** Running instances of programs managed by the kernel. Unix supports multitasking and multiuser environments by handling multiple processes simultaneously.
- **Utilities:** A rich set of small, specialized programs that perform specific tasks, like file manipulation, text processing, and system monitoring. These can be combined to perform complex operations.
- **Users and Permissions:** Unix is designed to support multiple users, each with a defined set of permissions controlling access to files and resources.

<u>Unix Structure: File System</u>



Unix organizes data in a hierarchical file system that resembles an upside-down tree. At the top, you have the root directory (/), and everything else branches out from there.

- Root Directory (/): The starting point of the Unix file system.
- Directories: Containers for files and other directories (subdirectories).
- Files: Basic units of storage. Unix treats everything as a file — whether it's a text document, a device, or a process.
- Absolute vs Relative Paths:
  - *Absolute Path:* Full path starting from root, e.g., /home/user/docs.
  - *Relative Path:* Path relative to the current directory, e.g., docs/report.txt.
- File Types:
  - Regular files (text or binary)
  - Directories
  - Special files (devices, sockets, pipes)
- Permissions: Unix controls access with read (r), write (w), and execute (x) permissions for user, group, and others.
- Inodes: Each file is represented by an inode, containing metadata like ownership, permissions, timestamps, and file location.
- Mounting: File systems can be mounted to specific directories, allowing access to storage devices or remote filesystems in the hierarchy.

Unix Essential Commands

Unix commands are the building blocks for interacting with the Unix operating system. They allow users to navigate the file system, manage files and processes, and perform various system tasks efficiently.

File and Directory Management

- ls
  Lists files and directories in the current directory.
  Example: ls -l (detailed list with permissions, size, date)

- cd
  Changes the current directory.
  Example: cd /home/user moves to the user’s home directory.
- pwd
  Prints the full path of the current directory.
- mkdir
  Creates a new directory.
  Example: mkdir new_folder
- rmdir
  Removes an empty directory.
  Example: rmdir old_folder
- cp
  Copies files or directories.
  Example: cp file1.txt file2.txt (copies file1.txt to file2.txt)
- mv
  Moves or renames files or directories.
  Example: mv oldname.txt newname.txt
- rm
  Removes files or directories. Use with caution!
  Example: rm file.txt or rm -r folder (recursive delete)

Viewing and Searching File Contents

- cat
  Displays the contents of a file.
  Example: cat notes.txt
- less
  Views file content page by page (better for large files).
  Example: less logfile.log
- head / tail
  Displays the first or last lines of a file.
  Example: head -n 10 file.txt (first 10 lines)
  Example: tail -n 20 file.txt (last 20 lines)
- grep
  Searches for a specific pattern in files.
  Example: grep "error" logfile.log
- find
  Searches for files or directories based on criteria.
  Example: find /home -name "*.txt"

Permissions and Ownership

- chmod
  Changes file or directory permissions.
  Example: chmod 755 script.sh
- chown
  Changes the owner or group of a file or directory.
  Example: chown user:group file.txt

Process Management

- ps
  Lists running processes.
  Example: ps aux (all processes)
- kill
  Sends signals to processes, often to terminate them.
  Example: kill 1234 (kills process with PID 1234)
- top
  Displays real-time system and process information.

System and Disk Utilities

- df
  Shows disk space usage of file systems.
  Example: df -h (human-readable format)
- du
  Shows disk usage for files and directories.
  Example: du -sh /var/log

Networking and Remote Access

- ssh
  Securely connects to a remote machine via command line.
  Example: ssh user@remote_host

Miscellaneous

- echo
  Prints text or variables to the terminal.
  Example: echo "Hello, Unix!"

- tar
  Archives files into a compressed file (tarball).
  Example: tar -czvf archive.tar.gz folder/
- man
  Displays the manual page for a command.
  Example: man ls

Directory and File Commands

UNIX is a powerful operating system widely used for managing files and directories through a command-line interface. Mastering its basic commands is essential for efficient navigation and file management.

1. Listing Files and Directories (ls)

The ls command lists the contents of a directory. By default, it shows only visible files.

ls -l displays detailed info such as permissions, ownership, size, and modification date.

ls -a shows all files, including hidden ones (those starting with a dot).

Example:ls -la

2. Changing Directories (cd)

The cd command changes the current working directory.

cd /path/to/directory moves to a specified directory.

cd .. moves up one directory level.

cd alone returns to the home directory.

Example:cd /usr/local

3. Displaying Current Directory (pwd)

pwd prints the full path of the current directory, helping users know their location in the file system.

Example:pwd