Notes 4: Linux FS

File System

Definition - The way files are stored and organized. Description - Linux uses the hierarchical directory structure (tree-like pattern of folders). Formula - File System = Superblock + Inode Table + Data Blocks + Directory Structure Examples:

1. ext4(Fourth Extended File System)
2. XFS
3. Btrfs (B-Tree File System)

Path Name

Definition - Which indicates the location of the file in the filesystem (like an address). Description - It describes how to navigate through the directory hierarchy to reach the desired file or folder. Formula - Absolute Pathname Formula: Absolute Pathname = / + [directory] + / + [subdirectory] +...+ / + [filename] Relative Pathname Formula: Relative Pathname = [current_directory] + /+ [subdirectory] +...+ / + [filename] Examples: Absolute Pathname: /home/user/documents/file.txt Relative Pathname: /home/user/using documents/file.txt

Absolute Pathname

Definition - the location of a file starting a the root of the file system. Description - It provides the complete location of a file or directory in the filesystem, regardless of the current working directory. Formula - Absolute Pathname Formula: Absolute Pathname = / + [directory] + / + [subdirectory] +...+ / + [filename] Examples: Absolute Pathname: /home/user/documents/file.txt

Relative Pathname

Definition - the location of a file starting from the current working directory that is located inside the current working directory. Description - a relative pathname does not start with a / (root) but begins from the current directory, denoted by. Formula - Relative Pathname Formula: Relative Pathname = [current_directory] + /+ [subdirectory] +...+ / + [filename] Examples: Relative Pathname: /home/user/using documents/file.txt

YOUR HOME directory and THE HOME directory

Your Home directory - is the specific to you and holds your personal files The Home directory - is a general directory that contains the home directories of all users on the system. Formula -Your Home directory = /home/username The Home directory = /home Examples: /home/john /home/alice /home

Parent Directory

Definition - is the directory that contains the current directory. It is represented by two dots (..) in the filesystem. Description - The parent directory in Linux is the directory that directly contains the current directory. Formula - (..) Examples: Parent Directory: (cd..) / (cd ../..)

Child Directory or Subdirectory

Definition - is essentially a directory nested within another directory(known as the parent directory). It's a way of organizing files in a hierarchical structure. Description - Each directory within this structure can contain files and other directories, which are called subdirectories(or child directories). Formula - mkdir -p Documents/Work mkdir -p Documents/Personal Examples: mkdir Project mkdir Project1 Project2 Project3 mkdir -p Project/Code/Java

Bash Special Characters

Definition - are quite the toolkit for power users and subscripts. Description - Bash special characters include asterisk, question mark, brackets, pipe, ampersand, semicolon, backticks, etc. Formula - (file * matches file1, file2.txt), (ls | grep 'txt'), (echo "Hello "World"")

Examples: Asterisk - ls *.txt Question Mark - ls file?.txt Brackets - ls file[12].txt Pipes - cat file.txt | grep "hello"

Environment Variables

Definition - are dynamic values that affect the way processes behave on a system. Description - They store configuration settings and other information, such as file paths, directory locations, user details, and preferences. Imagine them as the backstage crew at a theater—essential for making sure everything runs smoothly, even though the audience (users) don't always see them. Formula - VARIABLE_NAME=value export VARIABLE_NAME MY_VARIABLE="Hello, World!" export MY_VARIABLE

Examples: export PATH=$PATH:/usr/local/bin export HOME=/home/username export USER=username export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH

User Defined Variables

Definition - Are custom variables that you, the user, create for your own scripts and processes. Description - They work similarly to environment variables but are typically used within the context of a particular shell session or script. Formula - VARIABLE_NAME=value echo $VARIABLE_NAME

Examples: GREETING="Hello, Copilot!" echo $GREETING

NUMBER=42 echo "The answer to life, the universe, and everything is $NUMBER"

#!/bin/bash FILE_PATH="/home/user/documents/file.txt" if [-f "$FILE_PATH"]; then echo "File exists." else echo "File does not exist." fi

Why do we need to use $ with variables in bash shell scripting?

Using $ before a variable in Bash scripting lets the shell know you want to access the value of the variable, rather than its name. It's kind of like saying, "Hey, show me what's inside this box" instead of "Here's the label on the box." Without $, Bash would just treat it as plain text.