Diego Trevino

CS 470

Lab 1

Linux Commands Explanations

cp

The cp command is used to copy files or directories from one location to another. It is commonly used when creating backups or duplicating files. For example, running cp notes.txt notes_backup.txt creates a copy of the file named notes.txt and saves it as notes_backup.txt.

ps

The ps command displays information about the processes currently running on the system. It helps users see which programs are active. For example, ps aux shows a detailed list of all running processes along with their resource usage.

ls

The ls command lists files and directories in the current directory. It is one of the most frequently used Linux commands. For example, ls -l shows files with additional details such as permissions, size, and modification date.

mv

The mv command is used to move or rename files and directories. For example, mv oldname.txt newname.txt renames a file from oldname.txt to newname.txt.

rm

The rm command removes files or directories from the system. It should be used carefully since deleted files are not easily recovered. For example, rm temp.txt deletes the file named temp.txt.

mkdir

The mkdir command creates new directories. It is useful when organizing files into folders. For example, mkdir Lab1 creates a directory called Lab1.

rmdir

The rmdir command removes empty directories. If the directory is not empty, the command will fail. For example, rmdir oldFolder deletes the directory named oldFolder if it is empty.

pwd

The pwd command prints the full path of the current working directory. This is helpful for knowing exactly where you are in the file system. For example, running pwd might return /home/diego.

stat

The stat command displays detailed information about a file, such as size, permissions, and timestamps. For example, stat create_files_with_subdirs.sh shows metadata about the script file.

tar

The tar command is used to archive multiple files into a single file, often for backup or transfer purposes. For example, tar -czvf lab1.tar.gz OS-Lab1/ creates a compressed archive of the OS-Lab1 directory.

apt

The apt command is used to install, update, and manage software packages on Ubuntu systems. For example, sudo apt update updates the list of available software packages on the system.

echo

The echo command prints text to the terminal or writes text to a file when combined with redirection. For example, echo "Hello Linux" > hello.txt writes the text “Hello Linux” into a file named hello.txt.

cat

The cat command displays the contents of a file in the terminal. It is often used to

quickly view text files. For example, cat hello.txt prints the contents of the file to the screen.

more

The more command allows users to view the contents of a file one page at a time. It is useful for large files. For example, more script.log displays the log file gradually instead of all at once.

less

The less command is similar to more but provides more control, such as scrolling and searching. For example, less script.log lets the user scroll through the log file using the keyboard.

date

The date command displays the current date and time. It can also format the output. For example, date '+%Y-%m-%d_%H-%M-%S' prints the date and time in a custom format.

time

The time command measures how long a command takes to execute. For example, time ./create_files_with_subdirs.sh shows how long the script took to run.

kill

The kill command sends a signal to terminate a running process using its process ID (PID). For example, kill 1234 stops the process with PID 1234.

history

The history command shows a list of previously executed commands. This is helpful for reviewing or reusing commands. For example, history | tail shows the most recent commands.

chmod

The chmod command changes file permissions. It is commonly used to make scripts executable. For example, chmod +x create_files_with_subdirs.sh allows the script to be run.

chown

The chown command changes the owner or group of a file. It usually requires administrator privileges. For example, sudo chown diego:diego create_files_with_subdirs.sh sets the file owner to the user diego.

Script Purpose and Explanation

The purpose of this script is to automate the creation of directories and files using a Bash script in a Linux environment. The script creates a main directory

named using the current date and time, then generates ten subdirectories inside it. Each subdirectory contains text files that store the names of different programming languages.

The script uses loops to efficiently create multiple directories and files without repeating code. It also includes basic error handling to ensure that directories and files are created successfully. All actions performed by the script are recorded in a log file with timestamps so the execution process can be tracked and verified.