2.1 Fundamentos de la Línea de Comandos

Introducción: La línea de comandos es una interfaz fundamental para interactuar con sistemas Linux. Aunque las interfaces gráficas (GUI) son comunes, la línea de comandos (CLI - Command Line Interface) ofrece mayor potencia, flexibilidad y eficiencia para muchas tareas administrativas y de desarrollo. Entender sus fundamentos es esencial.

Componentes Clave:

1. **Terminal (o Consola):** Es la ventana o programa que te permite acceder a la línea de comandos. Antiguamente eran dispositivos físicos, hoy suelen ser emuladores de terminal (como GNOME Terminal, Konsole, xterm, o la consola virtual accesible con `Ctrl+Alt+F1` a `F6`). Proporciona la interfaz visual para interactuar con el Shell.
2. **Shell:** Es el intérprete de comandos. Es el programa que *lee* los comandos que escribes, los *interpreta* y le *pide* al sistema operativo (kernel) que los *ejecute*. Hay varios shells disponibles en Linux, siendo **Bash (Bourne Again SHell)** el más común y el predeterminado en la mayoría de las distribuciones. Otros shells populares incluyen Zsh, Fish, Ksh, etc. El shell también proporciona funcionalidades como historial de comandos, autocompletado (con la tecla `Tab`), gestión de trabajos, scripting, etc. Puedes saber qué shell estás usando con el comando `echo $SHELL`.
3. **Prompt:** Es el símbolo o texto que muestra el shell para indicarte que está listo para recibir un comando. Suele incluir información útil como tu nombre de usuario, el nombre del host (máquina), y el directorio actual. Un prompt típico podría ser `usuario@hostname:~$`. El símbolo `$` indica un usuario normal, mientras que # suele indicar el usuario root (superusuario).

Sintaxis Básica de un Comando: La estructura general de un comando en Linux es:

Bash

```
comando [opciones] [argumentos]
```

- **Comando:** El nombre del programa o utilidad que quieres ejecutar (e.g., `ls`, `cp`, `mkdir`). Los comandos son sensibles a mayúsculas y minúsculas.
- **Opciones (o Flags/Switches):** Modifican el comportamiento del comando. Suelen empezar con un guion (`-`) para opciones cortas (una sola letra, a veces se pueden agrupar, e.g., `ls -la`) o dos guiones (`--`) para opciones largas (más descriptivas, e.g., `ls --all --human-readable`).
- **Argumentos:** Indican sobre qué debe actuar el comando (e.g., nombres de archivo, directorios, texto).

Ejemplo Desglosado: `ls -l /home/usuario`

- `ls`: El comando (listar contenido de directorio).

- `-l`: Una opción (mostrar formato largo).
- `/home/usuario`: Un argumento (el directorio cuyo contenido queremos listar).

Ejecución de Comandos:

1. Escribes el comando completo en el prompt.
2. Pulsas `Enter`.
3. El shell interpreta la línea: identifica el comando, las opciones y los argumentos.
4. El shell busca el programa (`ls` en el ejemplo) en los directorios definidos en la variable de entorno `PATH`.
5. Si lo encuentra, le pide al kernel que lo ejecute, pasándole las opciones y argumentos.
6. El comando se ejecuta y, generalmente, muestra su salida en la terminal.
7. El shell muestra de nuevo el prompt, listo para el siguiente comando.

Variables de Entorno: Son variables que almacenan información sobre el entorno del shell y del sistema, como `PATH` (dónde buscar comandos), `HOME` (directorio personal del usuario), `USER` (nombre de usuario), `SHELL` (shell actual). Se pueden ver con `echo $NOMBRE_VARIABLE` (e.g., `echo $PATH`) y listar todas con `env` o `printenv`.

Pipes (`|`) y Redirecciones (`>`, `>>`, `<`):

- **Pipe (`|`):** Envía la salida estándar de un comando a la entrada estándar de otro. Permite encadenar comandos. Ejemplo: `ls -l | grep ".txt"` (lista en formato largo y filtra las líneas que contienen ".txt").
- **Redirección de Salida (`>`):** Envía la salida estándar de un comando a un archivo (sobrescribiéndolo). Ejemplo: `ls -l > lista_archivos.txt`.
- **Redirección de Salida (Anexar) (`>>`):** Envía la salida estándar de un comando a un archivo (añadiéndola al final). Ejemplo: `date >> log.txt`.
- **Redirección de Entrada (`<`):** Toma la entrada estándar de un comando desde un archivo. Ejemplo: `sort < archivo_desordenado.txt`.

Fuentes y Más Información:

- Tutorial de Linux Comandos básicos (FING - PDF) (Explica sintaxis y ejecución)
- 25 comandos de Linux básicos【Para principiantes】 (Axarnet) (Contiene ejemplos de pipes)
- Introducción a la línea de comandos de Linux (YouTube - Webinar)
-