Historia y Evolución de Linux 4
Diferencias entre Linux y Otros Sistemas Operativos 4
Distribuciones de Linux y Raspberry Pi OS 4
Relación entre Linux y Raspberry Pi OS 5
El Kernel de Linux 5
La Shell de Linux 5
Las Distribuciones de Linux 6
Interfaz Gráfica 6
¿Qué es la Terminal en Linux? 6
Ventajas de Usar la Terminal 7
Casos Prácticos del Uso de la Terminal en Raspberry Pi 7
Usuario en Linux 8
El Superusuario o Root 8
El Comando Sudo y el Archivo Sudoers 8
Ejemplo Práctico: Instalación de Htop 9
Gestión de Usuarios 9
Permisos en Linux 9
Crear un Script en Linux 10
Modificar Permisos con chmod 11
El Comando ifconfig 11
El Comando ping 11
El Comando netstat 12

Listar Archivos y Directorios 12
Comando ls 12
Comando cd 12
Comando pwd 13
Comando tree 13
El comando tree muestra la estructura de archivos y directorios en un formato jerárquico 13
Navegación por ficheros 14
El Directorio Raíz: / 14
/home 14
3. Archivos de Configuración del Sistema: /etc 14
Archivos Variables: /var 14
Archivos Temporales: /tmp 15
Programas y Comandos del Sistema: /bin y /sbin 15
Dispositivos del Sistema: /dev 15
Montaje de Dispositivos Externos: /media y /mnt 15
Gestión de ficheros 15
Comando mkdir: Crear Directorios 15
Comando rmdir 16
Comando touch: Crear y Actualizar Archivos 16
Comando mv: Mover y Renombrar Archivos 16
Comando rm: Eliminar Archivos y Directorios 16
Comando cp: Copiar Archivos y Directorios 17

Comandos de sistema 17

Comando uname: Información del Sistema 17

Comando cat: Ver Contenidos de Archivos 17

Comando top: Monitorización de Procesos 18

Comando free: Ver Uso de Memoria 18

Comando df: Ver Espacio en Disco 18

Comando du: Ver Uso de Espacio por Directorio 18

Historia y Evolución de Linux

Linux es un sistema operativo de código abierto creado en 1991 por **Linus Torvalds**, un joven estudiante finlandés que buscaba desarrollar una alternativa libre al sistema UNIX.
Linux es el **kernel** del sistema operativo, es decir, su núcleo. Para formar un sistema operativo completo, Linux se combinó con el proyecto **GNU**, iniciado por **Richard Stallman** en 1983, que proporcionaba herramientas y utilidades fundamentales. Esta colaboración dio lugar a las distribuciones de Linux modernas.

Desde sus inicios, Linux ha crecido exponencialmente, adoptado por una vasta comunidad global. Hoy se encuentra no solo en servidores y supercomputadoras, sino también en teléfonos móviles, automóviles y dispositivos como la **Raspberry Pi**.

Diferencias entre Linux y Otros Sistemas Operativos

En comparación con **Windows** y **macOS**, Linux es un sistema operativo **de código abierto**, mientras que los otros son propietarios. Esto significa que, en Linux:

- Los usuarios tienen **control total** sobre el sistema.
- Pueden instalar software utilizando gestores de paquetes como **apt**, sin depender de sistemas centralizados.
- El control de las actualizaciones es manual, permitiendo decidir qué instalar y cuándo.

Por el contrario, en **Windows**, las actualizaciones son automáticas y, en ocasiones, invasivas.

Distribuciones de Linux y Raspberry Pi OS

Existen numerosas distribuciones de Linux, adaptadas a diferentes necesidades. Algunas de las más conocidas son **Ubuntu**, **Debian**, y **Fedora**.
En este curso, utilizaremos **Raspberry Pi OS**, una distribución basada en **Debian**, optimizada para el hardware de la Raspberry Pi. Esto incluye soporte para sus características únicas, como los pines GPIO, ideales para proyectos de hardware.

Además, hay otras distribuciones adaptadas a Raspberry Pi, como **Ubuntu for Raspberry Pi** y **Arch Linux**.

Relación entre Linux y Raspberry Pi OS

Raspberry Pi OS, al estar basado en Debian, comparte muchas características con esta distribución, pero está específicamente diseñado para aprovechar al máximo las capacidades de la Raspberry Pi, haciéndolo ideal tanto para aprender Linux como para proyectos educativos y experimentales.

El Kernel de Linux

El **kernel** es el **núcleo del sistema operativo**. Se encarga de gestionar la comunicación entre el hardware y el software. Podemos pensar en él como un puente que conecta los programas que utilizamos con los componentes físicos del dispositivo, como la **memoria**, el **procesador**, y los **dispositivos de entrada/salida**.

Por ejemplo, cuando conectamos un **dispositivo USB** a nuestra **Raspberry Pi**, el kernel reconoce el dispositivo, carga los controladores necesarios y lo hace accesible. Gracias al kernel, podemos utilizar el USB para **almacenar** o **transferir archivos** de manera sencilla.

Además, el kernel gestiona los **recursos del sistema**, distribuyendo la carga de trabajo entre el procesador y la memoria, permitiendo que varios programas se ejecuten simultáneamente sin interferir entre sí.

La Shell de Linux

La **shell** es la interfaz que utilizamos para interactuar con el sistema operativo mediante comandos. En Linux, cuando abrimos una ventana de **terminal**, estamos utilizando la shell para enviar instrucciones al sistema. Esta shell traduce los comandos que escribimos y los pasa al kernel para que se ejecuten.

Uno de los aspectos más útiles de la shell es la capacidad de **automatizar tareas** mediante **scripts**. Un **script** es un archivo que contiene una serie de comandos que se ejecutan de forma secuencial. Por ejemplo, podrías crear un script que automáticamente:

- **Limpie archivos temporales**.
- **Realice una copia de seguridad diaria**.

Estos scripts optimizan la gestión del sistema y ahorran tiempo en tareas rutinarias.

Las Distribuciones de Linux

Linux no es un único sistema operativo, sino que existen muchas versiones llamadas **distribuciones**. Cada distribución combina el **kernel** con herramientas y software adaptados a distintos usos.

Algunas de las distribuciones más populares son **Ubuntu**, **Debian**, y **Fedora**. Estas distribuciones utilizan diferentes **gestores de paquetes** para gestionar software:

- En **Debian** y **Raspberry Pi OS**, se utiliza **apt**.
- En **Fedora**, se utiliza **dnf**.

Por ejemplo, en **Raspberry Pi OS**, puedes actualizar todo el sistema con el comando: sudo apt update && sudo apt upgrade.
Este comando descarga e instala actualizaciones de forma controlada, manteniendo el sistema seguro y actualizado sin interrupciones inesperadas.

Interfaz Gráfica

Una **interfaz gráfica de usuario** o **GUI** (Graphical User Interface) es el entorno que la mayoría asociamos con sistemas operativos modernos. En una GUI, interactuamos con el sistema operativo mediante elementos visuales como ventanas, iconos, menús y botones. Este tipo de interfaz es intuitivo y accesible, facilitando su uso para quienes no están familiarizados con comandos o terminales.

Por ejemplo, en **Raspberry Pi OS**, puedes:

- Acceder al explorador de archivos con un clic.
- Mover carpetas arrastrándolas.
- Instalar software a través de un gestor gráfico.

Sin embargo, las interfaces gráficas **consumen más recursos** del sistema, lo que puede ralentizar el rendimiento en dispositivos con **hardware limitado**, como la **Raspberry Pi**. Por esta razón, la terminal es una alternativa más rápida y eficiente para tareas avanzadas.

¿Qué es la Terminal en Linux?

En contraste, la **terminal** es una **interfaz de línea de comandos** o **CLI** (Command Line Interface), que nos permite interactuar con el sistema escribiendo comandos. En lugar

de hacer clic en iconos o menús, los usuarios introducen instrucciones directamente para ejecutar tareas.

Por ejemplo:

- Usar mv para mover archivos.
- Usar cp para copiarlos.

Aunque al principio puede parecer menos intuitiva, la terminal ofrece **más control y flexibilidad**, convirtiéndose en la herramienta preferida de los usuarios avanzados. En **Raspberry Pi OS**, puedes abrir la terminal desde el menú principal o con el atajo **Ctrl + Alt + T**.

Ventajas de Usar la Terminal

Aunque la interfaz gráfica es más accesible, la terminal tiene muchas ventajas, especialmente en dispositivos como la **Raspberry Pi**. Algunas de estas ventajas son:

1. **Mayor control y flexibilidad**: Permite realizar tareas avanzadas como instalar software, cambiar permisos de archivos o acceder a configuraciones ocultas.
2. **Automatización**: Puedes escribir **scripts** para automatizar tareas repetitivas, como respaldos automáticos.
3. **Eficiencia de recursos**: Trabajar desde la terminal consume **menos recursos** del sistema que una GUI.
4. **Acceso remoto**: Puedes administrar tu Raspberry Pi de manera remota usando **SSH**.
5. **Gestión de múltiples máquinas**: Ideal para administradores de sistemas que manejan varios servidores.

Casos Prácticos del Uso de la Terminal en Raspberry Pi

En **Raspberry Pi**, la terminal se usa comúnmente para:

1. Listar archivos en el directorio actual con ls. Solo escribe ls y presiona **Enter**.
2. Mover un archivo de una carpeta a otra con mv archivo.txt /home/pi/documentos.
3. Actualizar el sistema con

```
sudo apt update && sudo apt upgrade
```

Estos ejemplos básicos te ayudarán a familiarizarte con la terminal. A medida que avancemos en el curso, exploraremos cómo utilizarla para controlar tareas más complejas.

Usuario en Linux

En Linux, un **usuario** es cualquier persona o programa que interactúa con el sistema operativo. Cada usuario tiene un **perfil** con configuraciones y permisos específicos que determinan lo que pueden hacer dentro del sistema. Esto permite que múltiples personas utilicen el mismo dispositivo sin interferir entre sí y asegura que las tareas administrativas estén restringidas a ciertos usuarios.

Cada usuario tiene:

- Un **nombre de usuario**.
- Un **UID** (User Identifier).
- Uno o varios **grupos**, que sirven para compartir acceso a archivos y recursos de manera eficiente.

Además, cada usuario tiene un directorio personal en **/home**, donde se almacenan sus archivos y configuraciones. Por ejemplo, si tu nombre de usuario es "pi", tu directorio personal será **/home/pi**.
Puedes visualizar los directorios personales con el comando:

```
ls /home
```

El Superusuario o Root

En Linux, el **superusuario** llamado **root** tiene permisos completos sobre el sistema. Root puede:

- Modificar cualquier archivo.
- Instalar o eliminar programas.
- Cambiar configuraciones críticas del sistema.

Sin embargo, trabajar directamente como root es **peligroso**, ya que cualquier error podría comprometer seriamente el sistema. Por esta razón, en distribuciones como **Raspberry Pi OS**, se utiliza el comando **sudo** para realizar tareas administrativas con privilegios elevados de manera segura.

El Comando Sudo y el Archivo Sudoers

El comando **sudo** permite a un usuario ejecutar comandos con privilegios de **superusuario** sin necesidad de iniciar sesión como root. Por ejemplo, para actualizar el sistema, puedes usar:

```
sudo apt update && sudo apt upgrade
```

El archivo **/etc/sudoers** controla quién puede usar **sudo** y para qué comandos. Para editar este archivo de manera segura, utiliza el comando visudo, que verifica la sintaxis antes de guardar. Algunos elementos clave del archivo son:

- **Defaults env_reset**: Restablece las variables de entorno para mayor seguridad.
- **Defaults secure_path**: Define rutas seguras para comandos ejecutados con sudo.
- **root ALL=(ALL:ALL) ALL**: Especifica que el usuario root puede ejecutar cualquier comando.

Ejemplo Práctico: Instalación de Htop

Para instalar el programa **htop**, útil para monitorizar recursos del sistema, escribe:

```
sudo apt install htop
```

Después, ejecuta **htop** con el comando:

```
htop
```

Esto abrirá una herramienta interactiva para monitorear CPU, memoria y procesos en tiempo real.

Gestión de Usuarios

1. **Crear un usuario con permisos limitados**:

```
sudo adduser usuario_invitado
sudo usermod -G "" usuario_invitado
```

2. **Crear un usuario con permisos administrativos**:

```
sudo adduser usuario_administrador
sudo usermod -aG sudo usuario_administrador
```

3. **Verificar permisos**:

Inicia sesión como los usuarios creados y prueba comandos con y sin sudo para verificar sus permisos.

Permisos en Linux

En Linux, cada archivo y directorio tiene un conjunto de **permisos** que controlan quién puede acceder a ellos y qué acciones pueden realizar. Estos permisos se dividen en tres categorías:

1. **Lectura (r)**: Permite ver el contenido de un archivo o listar los archivos de un directorio.

2. **Escritura (w)**: Permite modificar un archivo o agregar/eliminar archivos en un directorio.
3. **Ejecución (x)**: Permite ejecutar un archivo (si es un programa o script) o acceder a un directorio.

Los permisos aplican a tres tipos de usuarios:

- **Propietario**: El usuario que creó el archivo o directorio.
- **Grupo**: Un conjunto de usuarios con acceso compartido.
- **Otros**: Todos los demás usuarios del sistema.

Crear un Script en Linux

Los scripts son archivos ejecutables que contienen una serie de comandos. Aquí aprenderás a crear un script que imprime "Hola Mundo".

1. **Crear el archivo del script**
   Abre una terminal y utiliza un editor de texto, como nano, para crear un archivo llamado hola_mundo.sh:

```
nano hola_mundo.sh
```

Escribe el siguiente contenido en el archivo:

```
#!/bin/bash
echo "Hola Mundo"
```

- **#!/bin/bash**: Le dice al sistema que use Bash para ejecutar el script.
- **echo**: Imprime "Hola Mundo" en la terminal.

Guarda el archivo presionando **Ctrl + O**, luego **Enter**, y sal del editor con **Ctrl + X**.

2. **Verificar permisos**
   Lista los permisos del archivo con:

```
ls -l
```

Los permisos tendrán un formato como -rw-r--r--, indicando que:

- El propietario puede leer y escribir.
- El grupo y otros solo pueden leer.

3. **Dar permisos de ejecución**
   Para ejecutar el script, añade permisos de ejecución con:

```
chmod +x hola_mundo.sh
```

Luego, ejecuta el script con:

```
./hola_mundo.sh
```

Modificar Permisos con chmod

El comando chmod permite cambiar los permisos de archivos y directorios. Ejemplos comunes incluyen:

- **Dar ejecución a todos**: chmod +x archivo.sh
- **Dar ejecución al propietario**: chmod u+x archivo.sh
- **Dar ejecución al grupo**: chmod g+x archivo.sh
- **Dar ejecución a otros**: chmod o+x archivo.sh
- **Quitar ejecución a todos**: chmod -x archivo.sh

El Comando ifconfig

El comando **ifconfig** se utiliza para ver la configuración de las interfaces de red en Linux. Permite obtener información sobre direcciones **IP**, **máscaras de subred**, y el estado de las interfaces de red.

Para visualizar la configuración actual, escribe:

```
ifconfig
```

Esto muestra detalles de cada interfaz, como:

1. **eth0 (Ethernet)**:
    a. Dirección IP: 192.168.80.47.
    b. Máscara de subred: 255.255.255.0.
    c. Dirección de transmisión: 192.168.80.255.
    d. Dirección MAC: d8:3a:dd:19:f4:4e.
2. **lo (Loopback)**:
    a. Dirección IP: 127.0.0.1.
    b. Utilizado para comunicaciones internas.
3. **wlan0 (Wi-Fi)**:
    a. Dirección IP: 192.168.1.47.
    b. Máscara de subred: 255.255.255.0.
    c. Activo y funcionando correctamente.

El Comando ping

El comando **ping** envía paquetes de prueba a una dirección IP o dominio, verificando si la conexión es estable y funcional. Para probar la conectividad con Google, escribe:

ping google.com

Esto muestra:

- Respuesta de los paquetes.
- Tiempo de respuesta.
- Porcentaje de paquetes perdidos.

Presiona **Ctrl + C** para detener el comando.

El Comando netstat

netstat es útil para monitorear conexiones de red y verificar puertos en uso. Esto permite identificar servicios activos y detectar posibles problemas de seguridad.

Para ver conexiones activas, usa:

```
netstat
```

Esto muestra:

1. **Conexiones de Internet activas (TCP/UDP)**:
    a. Estado de conexión (e.g., **ESTABLISHED**, **LISTEN**).
    b. Puertos utilizados (e.g., 80 para HTTP, 22 para SSH).
2. **Conexiones internas (UNIX domain sockets)**:
    a. Comunicación entre procesos locales, como DBus o systemd.

Listar Archivos y Directorios

El comando ls muestra los archivos y directorios en el directorio actual.

Comando ls

Lista básica:

```
ls
```

Mostrar archivos ocultos:

```
ls -a
```

Lista detallada con permisos, tamaños y fechas:

```
ls -l
```

Tamaños legibles (KB, MB, GB):

```
ls -lh
```

Listar recursivamente archivos en subdirectorios:

```
ls -R
```

Comando cd

El comando cd permite navegar entre directorios.

Moverse a un directorio específico:

```
cd Documentos
```

Volver al directorio anterior:

```
cd ..
```

Ir al directorio raíz:

```
cd /
```

Ir al directorio personal del usuario:

```
cd ~
```

Comando pwd

El comando pwd muestra la ruta completa del directorio actual, útil para saber dónde estás trabajando.

```
pwd
```

Comando tree

El comando tree muestra la estructura de archivos y directorios en un formato jerárquico.

Instalar tree:

```
sudo apt install tree
```

Mostrar la estructura básica:

```
tree
```

Incluir archivos ocultos:

```
tree -a
```

Especificar niveles de profundidad:

```
tree -L 2
```

Mostrar tamaños de archivos:

```
tree -h
```

Navegación por ficheros

El Directorio Raíz: /

El directorio raíz es el punto de inicio de todo el sistema de archivos en Linux. Todos los archivos y directorios están organizados bajo este directorio, representado por /.

Navegar al directorio raíz:

```
cd /
```

Contiene todos los demás directorios del sistema. No se recomienda modificar nada directamente aquí, ya que alberga componentes críticos del sistema.

/home

Cada usuario tiene un subdirectorio dentro de /home, donde se almacenan sus archivos y configuraciones personales. Por ejemplo, el directorio personal de un usuario llamado usuario1 sería:

```
/home/usuario1
```

Navegar al directorio personal:

```
cd /home
```

3. Archivos de Configuración del Sistema: /etc

- **Descripción**:
  Contiene los archivos de configuración del sistema y de las aplicaciones instaladas.
- **Ejemplos importantes**:
  - **/etc/fstab**: Define cómo se deben montar los sistemas de archivos.

```
cat /etc/fstab
```

- **/etc/hostname**: Contiene el nombre del host del sistema.
- **/etc/hosts**: Asocia nombres a direcciones IP localmente.
- **/etc/sudoers**: Define permisos para usar sudo.

Archivos Variables: /var

- **Descripción**:
  Almacena datos que cambian constantemente, como logs del sistema, colas de impresión y datos de servicios.
- **Subdirectorios importantes**:
  - /var/log: Registros del sistema.
  - /var/cache: Archivos temporales de aplicaciones.
  - /var/spool: Archivos en cola para ser procesados.

Archivos Temporales: /tmp

- **Descripción**:
  Espacio para archivos temporales que pueden eliminarse automáticamente tras un reinicio. Útil para operaciones rápidas de entrada y salida de datos.
- **Ejemplo**:
  Directorios temporales generados por SSH o servicios gráficos como WayVNC.

Programas y Comandos del Sistema: /bin y /sbin

- **/bin**: Contiene comandos esenciales disponibles para todos los usuarios (e.g., ls, cp, mv).
- **/sbin**: Comandos del sistema reservados para el superusuario (e.g., ifconfig, reboot).

Dispositivos del Sistema: /dev

- **Descripción**:
  Representa dispositivos conectados al sistema, como discos duros, unidades USB y periféricos.
- **Ejemplo**:
  /dev/sda para un disco duro o /dev/tty para dispositivos serie.

Montaje de Dispositivos Externos: /media y /mnt

- **/media**: Directorio para dispositivos montados automáticamente, como memorias USB.
- **/mnt**: Usado para montar dispositivos manualmente.

Ejemplo de montaje manual:

```
sudo mount /dev/sda1 /mnt
```

Gestión de ficheros

Comando mkdir: Crear Directorios

El comando mkdir se utiliza para crear nuevos directorios.

- **Crear un directorio**:

```
mkdir carpeta_prueba
```

- **Crear varios directorios**:

```
mkdir carpeta3 carpeta4
```

- **Crear un directorio en una ruta específica**:

```
mkdir /home/minitower/Desktop/carpeta5
```

- **Crear directorios padres con -p**:

```
mkdir -p /home/minitower/Desktop/carpeta7/carpeta7.1
```

Comando rmdir

El comando rmdir se utiliza para eliminar directorios vacíos

Eliminar un subdirectorio vacío:

```
rmdir nombre_del_directorio
```

Eliminar un directorio con subdirectorios vacíos:

```
rmdir -p carpeta7/carpeta7.1
```

Comando touch: Crear y Actualizar Archivos

El comando touch sirve para crear archivos vacíos o actualizar las marcas de tiempo de archivos existentes.

Crear un archivo vacío:

```
touch archivo.txt
```

Crear varios archivos:

```
touch archivo1.txt archivo2.txt
```

Actualizar las marcas de tiempo:

```
touch archivo.txt
```

Especificar una fecha y hora:

```
touch -t 202401012359 archivo2.txt
```

Comando mv: Mover y Renombrar Archivos

El comando mv se utiliza para mover archivos o cambiar sus nombres.

Mover un archivo a otro directorio:

```
mv archivo1.txt /home/minitower/Desktop/carpeta4
```

Renombrar un archivo:

```
mv archivo1.txt archivo_nuevo_nombre.txt
```

Mover varios archivos a un directorio:

```
mv archivo1.txt archivo2.txt /home/minitower/Desktop/carpeta4
```

Comando rm: Eliminar Archivos y Directorios

El comando rm elimina archivos y directorios.

Eliminar un archivo:

```
rm archivo.txt
```

Eliminar un directorio y su contenido:

```
rm -r carpeta3
```

Eliminar sin confirmación:

```
rm -f archivo.txt
```

Eliminar con permisos elevados:

```
sudo rm archivo.txt
```

Comando cp: Copiar Archivos y Directorios

El comando cp copia archivos y directorios a otras ubicaciones.

Copiar un archivo:

```
cp archivo.txt /home/minitower/Desktop/carpeta6
```

Copiar y renombrar un archivo:

```
cp archivo.txt nuevo_nombre.txt
```

Copiar varios archivos:

```
cp archivo1.txt archivo2.txt /home/minitower/Desktop/carpeta6
```

Copiar un directorio y su contenido:

```
cp -r carpeta_origen /home/minitower/Desktop
```

Comandos de sistema

Comando uname: Información del Sistema

El comando uname proporciona información básica sobre el sistema operativo.

Muestra el nombre del kernel:

```
uname
```

Información detallada (nombre, versión, arquitectura):

```
uname -a
```

Comando cat: Ver Contenidos de Archivos

El comando cat permite visualizar el contenido de archivos directamente en la terminal.

Ejemplo: Ver información de la CPU:

```
cat /proc/cpuinfo
```

Mostrar memoria disponible:

```
cat /proc/meminfo
```

Comando top: Monitorización de Procesos

El comando top muestra en tiempo real los procesos que consumen recursos en el sistema.

Uso básico:

```
top
```

Muestra información como:

- Uso de CPU.
- Uso de memoria.
- Tareas activas.

Presiona **q** para salir del comando.

Comando free: Ver Uso de Memoria

El comando free muestra un resumen del uso de memoria RAM y swap.

```
free -h
```

La opción -h muestra la información en un formato legible (e.g., MB, GB).

Comando df: Ver Espacio en Disco

El comando df reporta el uso del espacio en disco para cada sistema de archivos montado.

```
df -h
```

Espacio utilizado y disponible:

- Used: Espacio ocupado.
- Available: Espacio libre.

Comando du: Ver Uso de Espacio por Directorio

El comando du muestra el tamaño de archivos y directorios.

```
du -h
```

Directorio específico:

```
du -h /home
```