PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ FACULTAD DE CIENCIAS E INGENIERÍA

MTR343

TECNOLOGÍAS DE AUTOMATIZACIÓN

Laboratorio N° 1

INTRODUCCIÓN A LINUX BÁSICO

2024-2

CONTENIDO

1. Objetivos del laboratorio

• Introducir a los estudiantes al entorno Linux, destacando la importancia de los sistemas operativos basados en Unix/Linux en el ámbito tecnológico y profesional.

• Enseñar los comandos básicos más utilizados en Linux, como ls, cd, cp, mv, rm, entre otros, y su aplicación para la gestión eficiente de archivos y directorios.

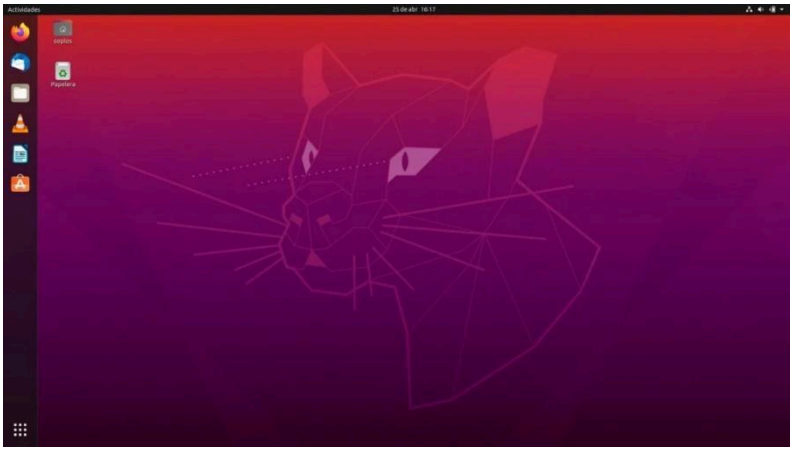
• Capacitar a los estudiantes en la navegación y manipulación del sistema de archivos mediante la terminal, optimizando su capacidad para realizar tareas administrativas.

• Proveer conocimiento sobre la gestión de procesos en Linux, incluyendo cómo visualizar, iniciar y finalizar procesos utilizando comandos como ps, top, y kill.

• Fomentar la práctica y el dominio de las operaciones en línea de comandos para que los estudiantes puedan interactuar de manera efectiva con el sistema operativo, realizando configuraciones básicas y tareas comunes.

2. Fundamentos de Linux/Ubuntu

Ubuntu es una de las distribuciones más populares y accesibles de Linux, conocida por su facilidad de uso, estabilidad y seguridad. Se destaca por ofrecer una plataforma robusta tanto para usuarios nuevos como para desarrolladores experimentados, lo que lo convierte en una excelente opción para aprender sobre sistemas operativos basados en Linux.



Algunas de las bondades de Ubuntu incluyen:

• **Interfaz de usuario amigable:** Ubuntu ofrece un entorno de escritorio intuitivo y fácil de navegar, lo que reduce la curva de aprendizaje para aquellos que migran desde otros sistemas operativos.

• **Estabilidad y seguridad:** Ubuntu recibe actualizaciones regulares que mejoran su rendimiento y aseguran la protección contra vulnerabilidades.

• **Amplia comunidad de soporte:** Al ser una de las distribuciones más utilizadas, Ubuntu cuenta con una extensa comunidad en línea que ofrece recursos, tutoriales y soporte técnico.

• **Versatilidad en el uso:** Ubuntu se adapta a diversas necesidades, desde el uso cotidiano en ordenadores personales hasta su implementación en servidores y entornos de desarrollo.

• **Software de código abierto:** Ubuntu promueve el uso de software libre y de código abierto, permitiendo a los usuarios acceder a una gran variedad de herramientas y aplicaciones sin costo adicional.

3. Uso de la Terminal en Ubuntu

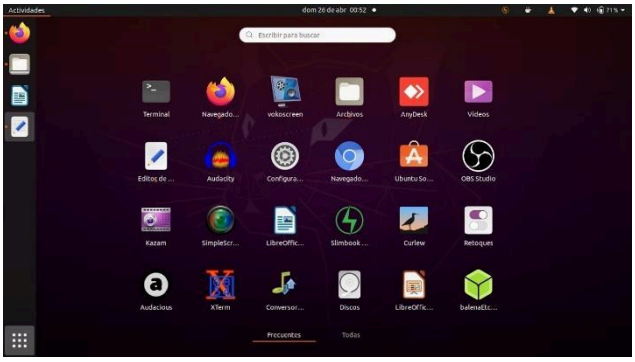
La terminal de Ubuntu es una herramienta poderosa que permite a los usuarios interactuar directamente con el sistema operativo utilizando comandos. A través de la terminal, es posible realizar tareas administrativas, gestionar archivos, ejecutar programas y mucho más, todo con gran eficiencia y control.

3.1. Acceso a la Terminal

Ubuntu ofrece varias formas de acceder a la terminal, lo que permite a los usuarios elegir la opción más conveniente según su preferencia o situación.

3.1.1. Acceso a través del Menú de Aplicaciones:

- **Método:** Haz clic en el ícono de la cuadrícula (Mostrar aplicaciones) en la esquina inferior izquierda del escritorio, escribe "Terminal" en la barra de búsqueda, y selecciona la aplicación Terminal.



3.1.2. Acceso por Atajo de Teclado:

- **Método:** Presiona Ctrl + Alt + T en tu teclado para abrir la terminal al instante.

4. Estructura del Sistema de Archivos en Linux

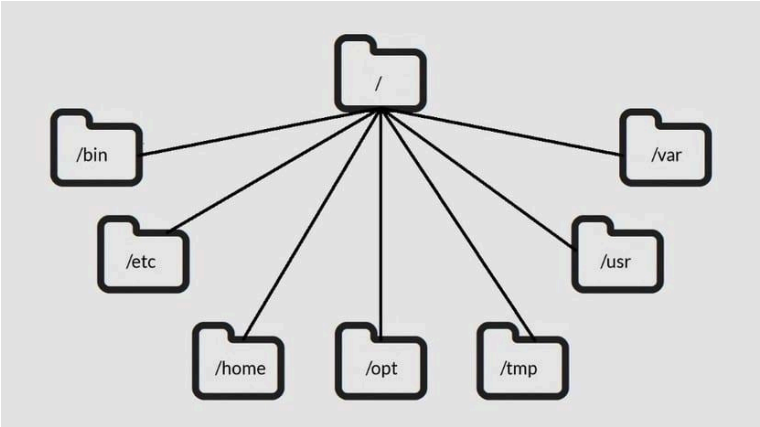
Se entiende como **directorio** es la estructura que organiza y almacena archivos en un sistema de archivos. Los directorios actúan como carpetas que contienen archivos y, en algunos casos, otros subdirectorios, permitiendo una jerarquía organizada de datos.

La raíz (`/`) es el punto de partida de toda la estructura de archivos en un sistema Linux. Es el directorio superior desde el cual se derivan todos los demás directorios. Dentro de la raíz, se encuentran los directorios principales como `/bin`, `/etc`, `/home`, `/var`, entre otros. Cada uno tiene un propósito específico, como almacenar binarios ejecutables, configuraciones del sistema, archivos de usuario, y datos de aplicaciones. La estructura jerárquica del directorio debe

mantenerse de esta manera para garantizar un correcto funcionamiento y organización del sistema operativo.

A continuación, se presenta una imagen que representa la estructura básica de los directorios en la raíz de un sistema Linux.

4.1. Raíz (/) y Directorios Principales



- **/**: El directorio raíz del sistema, a partir del cual se derivan todos los demás directorios.
- **/bin**: Contiene comandos esenciales y binarios necesarios para el arranque y mantenimiento básico del sistema
- **/home**: Contiene los directorios personales de los usuarios. Cada usuario tiene su propio subdirectorio dentro de /home.
- **/etc**: Contiene archivos de configuración del sistema y de las aplicaciones instaladas.
- **/var**: Almacena datos variables como registros de sistema (logs), bases de datos, archivos temporales, etc.
- **/usr**: Contiene aplicaciones y archivos de uso común por los usuarios.
- **/lib y /lib64**: Contienen las librerías esenciales del sistema y los módulos del núcleo (kernel).
- **/tmp**: Es un espacio temporal donde las aplicaciones y el sistema almacenan archivos temporales que se eliminan automáticamente en cada reinicio, siendo ideal para datos que no necesitan persistir.
- **/mnt y /media**: Puntos de montaje temporales para dispositivos externos como discos duros, USBs, y CDs/DVDs.
- **/opt**: Es el directorio para instalar aplicaciones de terceros y software adicional.

5. Comandos Básicos de la Terminal

5.1. Navegación de Directorios

- ***cd:*** Cambiar de directorio.
  Ejemplo: cd /home/user
- ***ls:*** Listar contenido de un directorio.
  Ejemplo: ls -l /etc
- ***pwd:*** Mostrar el directorio actual.
  Ejemplo: pwd

5.2. Manipulación de Archivos y Directorios

- ***cp:*** Copiar archivos o directorios.
  Ejemplo: cp archivo.txt /home/user/
- ***mv:*** Mover o renombrar archivos o directorios.
  Ejemplo: mv archivo.txt nuevo_nombre.txt
- ***rm:*** Eliminar archivos.
  Ejemplo: rm archivo.txt
- ***mkdir:*** Crear un nuevo directorio.
  Ejemplo: mkdir nuevo_directorio
- ***rmdir:*** Eliminar un directorio vacío.
  Ejemplo: rmdir nuevo_directorio

5.3. Visualización de Archivos

- ***cat:*** Crea un archivo y permite escribir contenido en él directamente desde la terminal. Se termina la entrada con Ctrl+D.
  Ejemplo: cat archivo.txt
- ***less y more:*** Visualizar contenido de archivos de forma paginada.
  Ejemplo: less archivo.txt
- ***head y tail:*** Mostrar las primeras o últimas líneas de un archivo.
  Ejemplo: head -n 10 archivo.txt
  tail -n 10 archivo.txt

5.4. Búsqueda de Archivos:

- ***find:*** Buscar archivos y directorios.
  Ejemplo: find /home/user -name "*.txt"
- ***grep:*** Buscar patrones dentro de archivos.
  Ejemplo: grep "error" /var/log/syslog
- ***locate:*** Encontrar archivos por nombre.
  Ejemplo: locate archivo.txt

5.5. Gestión de Procesos:

- ***ps:*** Listar procesos en ejecución.
  Ejemplo: ps aux
- ***top:*** Monitorizar procesos y uso del sistema en tiempo real.
- ***kill:*** Terminar un proceso.
  Ejemplo: kill 1234
- ***jobs, bg, y fg:*** Controlar procesos en segundo plano.

5.6. Redirección y Tuberías:

- ***Redirección de salida (> y >>):*** Enviar la salida de un comando a un archivo.
  Ejemplo: ls -l > lista.txt
- ***Tuberías (|):*** Pasar la salida de un comando como entrada de otro.
  Ejemplo: ls -l | grep ".txt"

6. Gestión de Usuarios y Grupos

6.1 Permisos y Propietarios de Archivos:

6.1.1 Propietarios

- **Usuario (owner):** El propietario del archivo.
- **Grupo:** Grupo de usuarios que comparten permisos en el archivo.
- **Otros:** Todos los demás usuarios.

6.1.2 Tipos de Permisos:

- **Lectura (r)**: Permite ver el contenido del archivo o listar el contenido de un directorio.
- **Escritura (w)**: Permite modificar el contenido del archivo o añadir/eliminar archivos en un directorio.
- **Ejecución (x)**: Permite ejecutar el archivo como un programa o acceder a un directorio.

6.1.3 Cambiar Permisos de Archivos

Para cambiar los permisos de un archivo o directorio, se utiliza el comando chmod (change mode). Este comando permite establecer los permisos de lectura, escritura y ejecución para el propietario, el grupo y otros usuarios.

Sintaxis Básica de chmod:

chmod [opciones] permisos archivo

Ejemplos de Uso:

- **Asignar permisos específicos:**

  Para otorgar permisos de lectura, escritura y ejecución al propietario (u), de lectura y ejecución al grupo (g), y solo de lectura a otros (o):

  chmod u=rwx,g=rx,o=r archivo.txt

- **Utilizar permisos numéricos:**

  Los permisos también pueden representarse mediante números. Por ejemplo, para dar permisos de lectura, escritura y ejecución al propietario, y solo lectura al grupo y otros:

  chmod 744 archivo.txt

  Aquí, "*7*" representa rwx (4+2+1), "*4*" representa r (4), y "*4*" representa r (4).

- **Añadir o quitar permisos:**

  Para añadir permisos de ejecución al propietario:

  chmod u+x archivo.txt

  Para eliminar permisos de escritura para el grupo:

  chmod g-w archivo.txt

6.2 ¿Qué es sudo?

Sudo (superuser do) es un comando en sistemas Linux/Unix que permite a un usuario autorizado ejecutar comandos con los privilegios de superusuario o de otro usuario, según se configure. Esto es útil para realizar tareas administrativas o de mantenimiento que requieren permisos elevados.

6.2.1 Principales funciones

- **Elevación de Privilegios**: Permite ejecutar comandos que requieren permisos de superusuario (root), como instalar o actualizar software, modificar configuraciones del sistema, y gestionar usuarios.
- **Seguridad**: Ofrece una forma más segura de obtener privilegios elevados sin tener que iniciar sesión como superusuario (root). Los usuarios deben proporcionar su propia contraseña para confirmar la acción.
- **Control de Acceso**: Permite configurar qué comandos específicos pueden ser ejecutados por usuarios normales mediante el archivo de configuración /etc/sudoers, proporcionando un control granular sobre el acceso.

Ejemplo de uso:

Para actualizar el sistema, se usa:

sudo apt update

Aquí, `sudo` eleva los permisos del comando apt update, que requiere privilegios de administrador para modificar el sistema.

6.3 Creación y Gestión de Usuarios:

- ***useradd:*** Crear un nuevo usuario.
  Ejemplo: sudo useradd -m nuevo_usuario
- ***usermod:*** Modificar un usuario existente.
  Ejemplo: sudo usermod -aG sudo nuevo_usuario
- ***passwd:*** Cambiar la contraseña de un usuario.
  Ejemplo: sudo passwd nuevo_usuario

6.4 Gestión de Grupos:

- ***groupadd:*** Crear un nuevo grupo.
  Ejemplo: sudo groupadd nuevo_grupo
- ***groupmod:*** Modificar un grupo existente.
  Ejemplo: sudo groupmod -n nuevo_nombre_grupo antiguo_grupo
- ***gpasswd:*** Administrar los miembros de un grupo.
  Ejemplo: sudo gpasswd -a nuevo_usuario nuevo_grupo

6.5 Permisos de Archivos y Directorios:

- ***chmod:*** Cambiar permisos de archivos/directorios.
  Ejemplo: chmod 755 script.sh

- ***chown:*** Cambiar propietario de archivos/directorios.
  Ejemplo: sudo chown user:group archivo.txt
- ***chgrp:*** Cambiar grupo de archivos/directorios.
  Ejemplo: sudo chgrp nuevo_grupo archivo.txt

6.6 Gestión de Software

6.6.1 Sistema de Gestión de Paquetes:

- **Instalar un paquete:**
  sudo apt install nombre_paquete
- **Actualizar la lista de paquetes:**
  sudo apt update
- **Actualizar el sistema:**
  sudo apt upgrade

6.6.2 Instalación, Actualización y Eliminación de Software:

- **Instalación**
  Ejemplo:
  sudo apt install git
- **Actualización**
  Ejemplo: sudo apt upgrade
- **Eliminación**
  Ejemplo: sudo apt remove git

7. Creación de archivo en lenguaje de programación C

Para la creación de un programa ejecutable en lenguaje C se dispone de los comandos 'nano', 'vi', o 'gedit'.

Paso 1: **Abrir un editor de texto**:

nano programa.c

Paso 2: **Escribir el código en C**:

Se puede usar el siguiente código de prueba:

```
#include <stdio.h>
 int main() {
          printf("Hola, Mundo!\n");
          return 0;
}
```

Paso 3: **Guardar y salir**:

- En nano, guarda presionando CTRL + O y luego ENTER.
- Luego, sal del editor con CTRL + X.

Paso 4: **Compilar el archivo C**:

Para compilar el archivo y generar un ejecutable, utiliza el compilador gcc:

gcc programa.c -o programa, donde:

- programa.c: Es el archivo fuente que acabas de crear.
- -o programa: Especifica el nombre del archivo ejecutable que se generará (en este caso, programa).

Paso 5: **Ejecutar el archivo compilado:**

./programa, esto mostrará la salida por consola con el siguiente mensaje:

Hola, Mundo!

8. Actividad: Instalación de Ubuntu

Como parte de la actividad del laboratorio se requiere la instalación de Ubuntu 18.04 por medio de los enlaces de referencia:

https://www.youtube.com/watch?v=5HlvreMFFn0 – Enlace donde se explica el procedimiento de instalación desde Windows hasta las opciones de dual boot.

https://releases.ubuntu.com/18.04/ - Enlace de la página donde se encuentran las versiones actualizadas para esa versión en específico.