Unidad 2 – Sistemas Operativos

1. ¿Qué es un Sistema Operativo?

Un SO es el programa principal que hace funcionar un computador.

- Gestiona el hardware
- Permite que los programas se ejecuten de manera ordenada y segura
- Actúa como intermediario entre el usuario y la máquina

2. Libertades y licencias de los SO

- **Licencia**: Es un contrato entre el desarrollador y el usuario
- Patente: Protege innovaciones tecnológicas durante un tiempo limitado
- **Copyright**: Protege frente a copias no autorizadas
- **GNU** (GNU´s Not Unix) iniciado por Richard Stallman en 1985, que promovió el concepto de software libre.
- La licencia GNU GPL da cuatro libertades al usuario: usar, estudiar, compartir y mejorar el software.

Ej: En 1988, Netscape liberó su navegador como software libre, lo que impulsó la idea de código abierto como estrategia de innovación

3. Origen de los SO

Los primeros SO eran muy básicos, de tipo monolítico (todo el código en un bloque). Con el tiempo fueron evolucionando:

- **MS-DOS**: Sencillo, dependía de programas y controladores con interacción directa al software
- **UNIX**: Más estructurado, con capas claras entre kernel, llamadas al sistema y hardware
- **Windows NT**: Sistema modular, pensado para ser más robusto y compatible con diferentes arquitecturas
- **POSIX**: Un estándar creado por garantizar compatibilidad entre distintos sistemas tipo UNIX, de forma que un mismo programa pudiera ejecutarse en varios de ellos.

4. Funciones y componentes del SO

El SO puede verse como el administrador de un computador. Sus funciones principales son:

- **Gestión de procesos y recursos:** Convierte programas en procesos activos y reparte los recursos entre ellos.
- **Gestión de memoria:** asigna espacio en la RAM a cada proceso.
- **Sistema de entrada/salida (I/O):** facilita el uso de dispositivos sin que el usuario tenga que entender el hardware directamente.
- **Sistema de archivos:** permite crear, leer, modificar y borrar archivos
- **Redes:** gestiona la comunicación en sistemas conectados entre sí
- **Protección y seguridad**: controla el acceso a memoria y recursos para evitar errores y ataques.
- **Interfaces de usuario**: pueden ser gráficas o de comandos

5. Particionamiento lógico

Antes de guardar datos en un disco duro es necesario particionarlo, dividirlo en secciones que el SO pueda reconocer y utilizar

- **MBR (Master Boot Record, o “Legacy”)**
  - Usa posiciones de 32 bits
  - Bloques de 512 bytes
  - Capacidad máxima: 2TB
  - Herramienta típica: fdisk
  - Es el método tradicional, pero hoy en día es insuficiente para discos modernos de gran tamaño
- **GPT (GUID Partition Table, usado con UEFI)**
  - Usa posiciones de 64 bits
  - Soporta discos de hasta 9 ZB (zettabytes)
  - Herramienta típica: gdisk
  - Es el estándar actual y necesario en sistemas modernos

MBR se quedó pequeño, GPT es el futuro

6. LVSM – Logical Volume System Management

El gestor de volúmenes lógicos (LVM) es una forma más flexible de organizar el almacenamiento que las particiones tradicionales. Ventajas principales:

- Permite redimensionar volúmenes sin necesidad de formatear o perder datos
- Posibilita la creación de snapshots (copias exactas en un momento dado)
- Es muy útil en servidores y entornos con grandes cantidades de datos

7. Gestión de sistemas de archivos (File System Management)

Es el encargado de orgnizar cómo se guardan los datos

- **Bloque**: unidad mínima de almacenamiento
- **Inode**: estructura que contiene información de un archivo (nombre, permisos, etc.)
- **Superblock**: almacena los metadatos del sistema de archivos (estructura, estado, etc.)
- **Hard Link**: mecanismo que permite que varios nombres diferentes apunten al mismo archivo.

8. Tipos de sistemas de archivos (ejemplos en Linux)

- **ext2** → Primer sistema de archivos extendido
- **ext3** → Añade journaling (registro previo de los cambios, reduce el riesgo de corrupción en apagones)
- **ext4** → Soporta discos grandes, mejora la gestión de bloques con extents (bloques contiguos para evitar fragmentación)
- **BTRFS** → Más moderno, soporta snapshots, permite compresión de archivos y verifica la integridad de los archivos
- **Subvolumenes (en BTRFS)** → Actúan como “carpetas especiales” que funcionan casi como particiones internas, pero con mayor flexibilidad.

9. Sistema de archivos en MS-DOS

El SO MS-DOS usaba un sistema muy sencillo llamado FAT (File Allocation Table)

- Distintas versiones: FAT12, FAT16 y FAT32, según el número de bits usados para las direcciones
  - No soportaba enlaces (ni duros ni simbólicos)
  - El tamaño de bloque es siempre un múltiplo de 512 bytes, llamado cluster
- Aunque fue muy básico, se convirtió en el estándar de los primeros sistemas y todavía se usa en dispositivos portátiles como USB o tarjetas SD porque es ligero y universal.

10. FAT-32

Limitaciones:

- Tamaño máximo de archivo: 4 GB
- No maneja permisos ni opciones avanzadas de seguridad

11. NTFS (New Technology File System)

Surgió como reemplazo moderno de FAT en Windows. Características principales:

- Sectores de 512 bytes.
- Tamaño de clúster flexible (desde 512 B hasta 64 KB)
- Usa una MFT (Master File Table) que guarda la información de todos los archivos del disco
- Soporta permisos de seguridad avanzados
- Maneja mejor archivos grandes y evita problemas de fragmentación

NTFS es el sistema de archivos estándar de Windows: robusto, seguro y eficiente

12. Jerarquía de archivos en Linux

El sistema de archivos está organizado como un árbol jerárquico que siempre empieza en la raíz /. Existe un estándar llamado Filesystem Hierarchy Standard (FHS) que define las carpetas más importantes:

- **/bin** → comandos esenciales
- **/etc** → archivos de configuración del sistema
- /**home** → directorios personales de los usuarios
- **/var** → archivos variables, como registros (logs) o colas de impresión.
- **/boot** → archivos de arranque del sistema.

En Linux no existen letras de disco (C:, D:), sino que todos los dispositivos se montan en la raíz /.

13. Sistema de archivos en Windows (vista lógica)

Windows también organiza sus archivos en estructuras de árbol, pero con diferencias

- Los archivos más importantes se encuentran en:
  - C:\Windows\System32 (núcleo del sistema operativo)
- Además, Windows tiene un elemento particular: el Registro (Registry).
  - Es una base de datos interna que almacena configuraciones del sistema, parámetros de hardware y programas instalados.
  - Cuando instalas un programa, crea una clave en el registro con su información.
- Diferencia con Linux:
  - Windows guarda configuraciones en el Registro, mientras que Linux las mantiene en archivos de texto plano, normalmente en la carpeta /etc.

14. El registro en Windows

Registry es la base de datos interna de Windows donde se guarda información esencial:

- Configuraciones del sistema
- Opciones de programas instalados
- Parámetros de hardware

Se organiza en ramas (keys) que contienen valores y configuraciones.

Ejemplo: HKEY_CURRENT_USER almacena la información del usuario que ha iniciado sesión.

- Se puede abrir con Win+R → regedit
- Funciona como el cerebro oculto de Windows.

15. Iniciando la Shell

La shell es la interfaz de línea de comandos que permite comunicarse directamente con el sistema operativo.

- **En Linux**:
  - La más común es bash
  - Se abre con Ctrl+Alt+T
- **En Windows**:
  - Se utiliza PowerShell
  - Se abre con Win+X → Windows PowerShell.
- **Funcionamiento básico:**
  - **Stdin** (**entrada estándar**) → lo que escribes en la consola
  - **Stdout (salida estándar**) → la respuesta o resultado
  - **Stderr (errores estándar)** → mensajes de error cuando algo falla

16. Características de los comandos

- **En Linux (bash):**
  - Los comandos pueden ser funciones, programas o alias.
  - Son case sensitive (distinguen entre mayúsculas y minúsculas).
  - Para ejecutar tareas administrativas se usa sudo.
  - El prompt cambia de $ (usuario normal) a # (administrador/root).
- **En Windows (PowerShell):**
  - Los comandos se llaman cmdlets.
  - Tienen la forma verbo-nombre (ejemplo: Get-Process).
  - NO son case sensitive (da igual usar mayúsculas o minúsculas).
  - Algunas tareas requieren abrir la consola como administrador.

17. Variables de entorno y PATH

- **Linux**:
  - Los directorios en el PATH se separan con dos puntos :
  - Se consulta con echo $PATH
  - Para hacerlo persistente, se edita un archivo de configuración como .bashrc.
- **Windows**:
  - Los directorios en el PATH se separan con punto y coma ;
  - Se consulta con $env:Path en PowerShell
  - Para hacerlo persistente, se puede usar el Registro o el comando setx.

18. Rutas relativas vs absolutas

- **Rutas absolutas:**
  - Empiezan siempre desde la raíz del sistema
  - Ej Linux: /home/usuario/
  - Ej Windows: C:\Users\Usuario\
- **Rutas relativas:**
  - Se interpretan en función de la carpeta en la que estamos actualmente
  - Símbolos clave:
    - . → Directorio actual
    - .. → Directorio padre
  - **Comandos útiles en Linux**
    - pwd → Muestra la carpeta actual
    - cd → Permite cambiar de directorio

19. Usuarios y grupos

- En Linux:
  - El usuario root es el superusuario con control total
  - Los usuarios normales tienen permisos limitados
  - Los usuarios pueden pertenecer a grupos, que sirven para compartir permisos
  - Para ejecutar tareas administrativas, se utiliza el comando sudo
- En Windows:
  - Existen dos perfiles principales:
    - Administradores → Tienen control total
    - Usuarios estándar → Acceso limitados

Ejemplo: con el comando net user administrator /active:yes se puede activar la cuenta de administrador.

20. Reglas básicas para usuarios

- En Linux:
  - El nombre de usuario suele tener máximo de 8 caracteres y no debe comenzar por un número
  - Cada usuario pertenece a un grupo primario, y puede estar en grupos secundarios
  - La información de las cuentas se guardan en:
    - /etc/passwd → Datos básicos del usuario
    - /etc/shadow → Contraseñas encriptadas
- En Windows:
  - Los usuarios y grupos se administran desde la interfaz gráfica o desde la consola con comandos como net user y net localgroup

21. Añadir usuarios

- **En Linux:**
  - Se usa el comando adduser nombre para crear un usuario nuevo
  - Al crearlo, se le asignan:
  - Directorio personal
  - Shell por defecto
  - Grupo primario y grupos secundarios
  - Contraseña.
- **En Windows:**
  - Desde CMD: net user juan contraseña /add
  - Desde PowerShell: New-LocalUser -Name "juan" -Password (ConvertTo-SecureString "contraseña" -AsPlainText -Force)

22. Modificación de usuarios y grupos en Linux

- **usermod** → modificar usuarios existentes
  - **usermod -L usuario** → bloquear una cuenta
  - **usermod -e 2025-12-31 usuario** → establecer fecha de caducidad
  - **usermod -d /nuevo/home usuario** → cambiar el directorio personal
- **groupmod** → renombrar grupos
- **groupadd / groupdel** → crear o eliminar grupos

- **gpasswd** → añadir o quitar usuarios dentro de un grupo.

23. Gestión de usuarios y grupos en Windows

Set-LocalUser → modificar usuarios (nombre, contraseña, descripción)
New-LocalGroup → crear un nuevo grupo
Add-LocalGroupMember → añadir usuarios a un grupo existente
Ejemplo práctico: New-LocalGroup -Name "desarrolladores" Add-LocalGroupMember -Group "desarrolladores" -Member "Juan"

24. Permisos en Linux

Tipos de usuarios:
Dueño (User) **Grupo** (group) **Otros** (others)
Tipos de permisos:
r → Lectura **w** → Escritura **x** → Ejecución
Comandos principales:
chmod → Cambiar permisos **chown** → Cambiar propietario **chgrp** → Cambiar grupo
Ejemplo:
rw- r-- r-- El dueño puede leer y escribir, el grupo solo puede leer y los demás usuarios solo pueden leer también.

25. Permisos en Windows

Windows maneja permisos a través de **ACLs (Access Control Lists)**, que permiten un control más detallado. Se administran con el comando **icacls**:
icacls archivo /grant usuario:(F) → Dar control total a un usuario
/remove → Quitar permisos
/inheritance → Aplicar herencia de permisos del directorio padre

26. Permisos especiales en Linux

Linux cuenta con bits especiales que permiten un control más avanzado que los permisos básicos:

- **SUID (Set User ID):**
  - Se aplica a archivos ejecutables
  - Cuando un usuario lo ejecuta, el programa corre con los permisos del dueño del archivo
  - **Ejemplo**: chmod u+s archivo
  - **Caso real:** el comando passwd usa SUID para poder modificar /etc/shadow, aunque lo ejecute un usuario normal.

- **SGID (Set Group ID):**
  - Similar al SUID, pero aplicado a grupos
  - En directorios, hace que todos los archivos nuevos creados dentro pertenezcan al mismo grupo.
  - **Ejemplo**: chmod g+s directorio.
- **Sticky bit:**
  - Usado en directorios compartidos como /tmp
  - Permite que solo el dueño del archivo (o root) pueda borrarlo, aunque otros usuarios tengan permisos de escritura en la carpeta
  - **Ejemplo**: chmod +t directorio

27. Instalación de aplicaciones en Linux

En distribuciones basadas en Debian/Ubuntu, el gestor de paquetes más usado es APT (Advanced Package Tool). Comandos básicos:

- **sudo apt-get update** → actualiza la lista de paquetes disponibles
- **sudo apt-get upgrade** → actualiza los programas ya instalados
- **sudo apt-get install nombre** → instala un programa
- **sudo apt-get remove nombre** → desinstala un programa.

Ventaja: los paquetes se descargan de repositorios oficiales, lo que garantiza seguridad y estabilidad.

28. Instalación de aplicaciones en Windows

- En **PowerShell** se usan cmdlets como:
  - **Install-PackageProvider**
  - **Find-Package**
  - **Install-Package**
  - **Get-Package**
- Además, Windows incluye la **Microsoft Store**, que facilita la instalación desde una interfaz gráfica.
- **Seguridad** en la ejecución de scripts (ExecutionPolicy):
  - **Restricted** → no se pueden ejecutar scripts
  - **AllSigned** → solo se permiten scripts firmados digitalmente
  - **RemoteSigned** → scripts locales permitidos, los descargados deben estar firmados
  - **Unrestricted** → se pueden ejecutar todos los scripts.

Resumen Conclusión

Los sistemas operativos son la base que permite que hardware y software trabajen juntos, evolucionando desde MS-DOS y UNIX hasta Windows, Linux y macOS, con mejoras en seguridad y comunicación. Aprendimos sobre **gestión de almacenamiento**, desde particiones MBR a GPT y LVM, y sistemas de archivos como FAT, FAT32 y NTFS en Windows, y la jerarquía FHS en Linux, que ofrecen distintos niveles de control y seguridad.

La **consola** permite automatizar tareas y administrar recursos (bash en Linux, PowerShell en Windows). La **gestión de usuarios y permisos** protege el sistema: Linux usa root, grupos y permisos r/w/x, mientras Windows usa administradores, ACLs y herramientas gráficas. Los permisos especiales en Linux y la gestión de paquetes en ambos sistemas (APT en Linux, Microsoft Store/PowerShell en Windows) garantizan seguridad y control.

En resumen, los sistemas operativos han pasado de simples a complejos y seguros, capaces de manejar equipos, redes y servicios en la nube. Windows prioriza facilidad de uso y seguridad centralizada, Linux flexibilidad y control del administrador, pero ambos buscan eficiencia, seguridad y fiabilidad para el usuario.