U.T.5.- SISTEMAS OPERATIVOS LIBRES: UBUNTU

Objetivos:

- Conocer los requisitos para la instalación de un sistema operativo.
- Planificar la instalación, eligiendo las particiones a crear y el sistema de archivos a utilizar.
- Seleccionar los parámetros y componentes básicos del sistema operativo que se va a instalar.
- Realizar una instalación de un sistema operativo GNU/Linux.
- Conocer las características principales de un SO GNU/Linux.
- Conocer el proceso de arranque y parada de un SO GNU/Linux.
- Conocer como referencia unidades de un SO GNU/Linux.
- Conocer el usuario root.
- Conocer el concepto de variable de entorno.
- Conocer que son los enlaces en SSOO GNU/Linux.
- Conocer la sintaxis básica de los comandos de un SO GNU/Linux.
- Manejar los comandos principales de un SO GNU/Linux.

5.1.- CONSIDERACIONES PREVIAS Y REQUISITOS PARA LA INSTALACIÓN DEL SO.

Antes de proceder a la instalación de un sistema Linux hay que tener en cuenta una serie de consideraciones:

Particionado del disco. Generalmente, se necesitan al menos dos particiones. Una para la instalación de Linux y todas las aplicaciones y otra para las funciones de la memoria virtual del SO, llamada SWAP.

Se aconseja que el tamaño del SWAP sea el doble de la memoria del sistema, es decir si un equipo tiene 1 GB de memoria física debería usarse un SWAP de 2 GB.

Todas las distribuciones disponen de una herramienta de particionado automática que se ejecuta dentro del programa de instalación. Sin embargo, se debe tener claro como se quiere el particionado antes de comenzar.

Gestor de arranque. Cuando se instala en el equipo más de un SO es necesario disponer de un gestor de arranque que permita seleccionar el SO con el que se quiere iniciar la máquina.

Lo más normal es usar un gestor de arranque incluido en la distribución como son los dos más usados LILO y GRUB. No se puede usar un gestor de arranque de Windows, ya que no reconoce particiones de Linux. Pero si se pueden usar otros gestores comerciales.

El usuario administrador. Durante el proceso de instalación se pedirá que se suministre una contraseña para el usuario que se creará en la instalación. El superusuario **root**, en Ubuntu está deshabilitado y para tener derechos de administración hay que ejecutar el comando **sudo**. Cualquier tarea de administración debería realizarse con este usuario.

La mayoría de las distribuciones también permiten la creación de un usuario normal en la instalación, en Ubuntu el usuario que se crea es un usuario con derechos de administración. El resto de los usuarios tienen esta opción desactivada por defecto, aunque es fácil de activar.

Software instalado. Normalmente, el número de programas instalados en una distribución es muy grande por lo que la mayoría de ellas permite elegir que aplicaciones o paquetes de software se instalarán y cuáles no. También los programas de instalación del SO suelen ofrecer distintos perfiles de instalación, para elegir el perfil que los usuarios y la máquina vayan a desempeñar, es decir, si la máquina se usará, por ejemplo, para tareas de ofimática o como servidor de algún tipo.

5.2.- INSTALACIÓN DEL SISTEMA OPERATIVO SOBRE UNA MÁQUINA VIRTUAL.

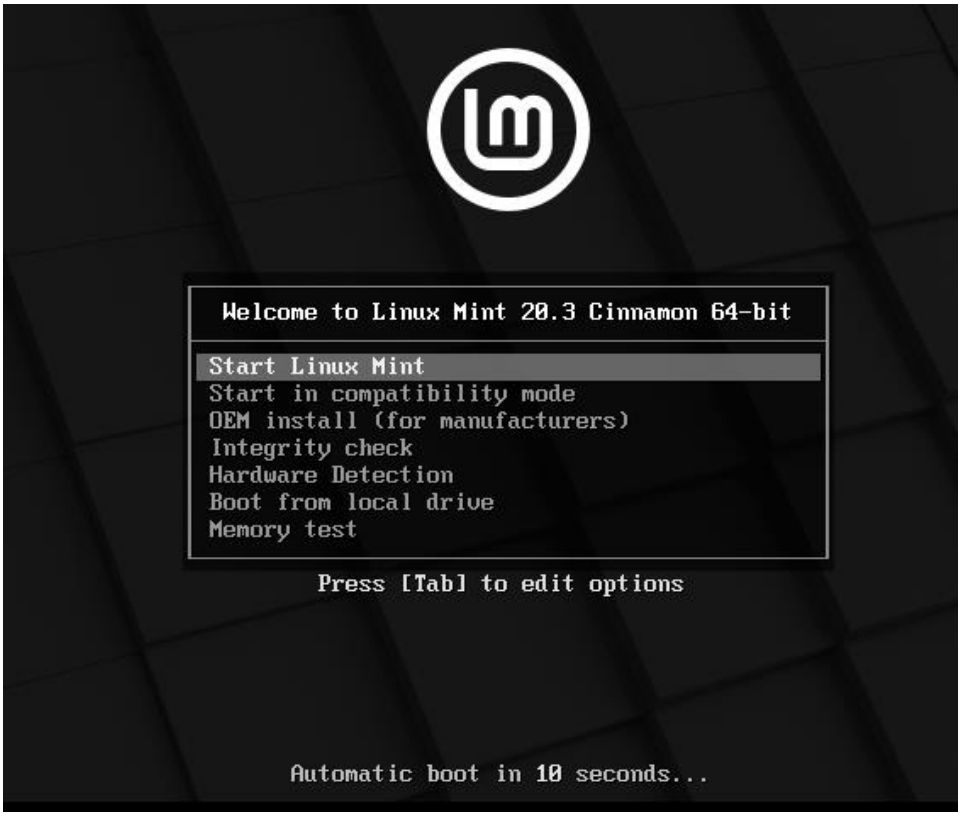
Instalaremos sobre una máquina virtual la versión de Linux Mint 20.03 para escritorio.

Antes de Instalar Linux Mint 20.03 tenemos que realizar las siguientes tareas:

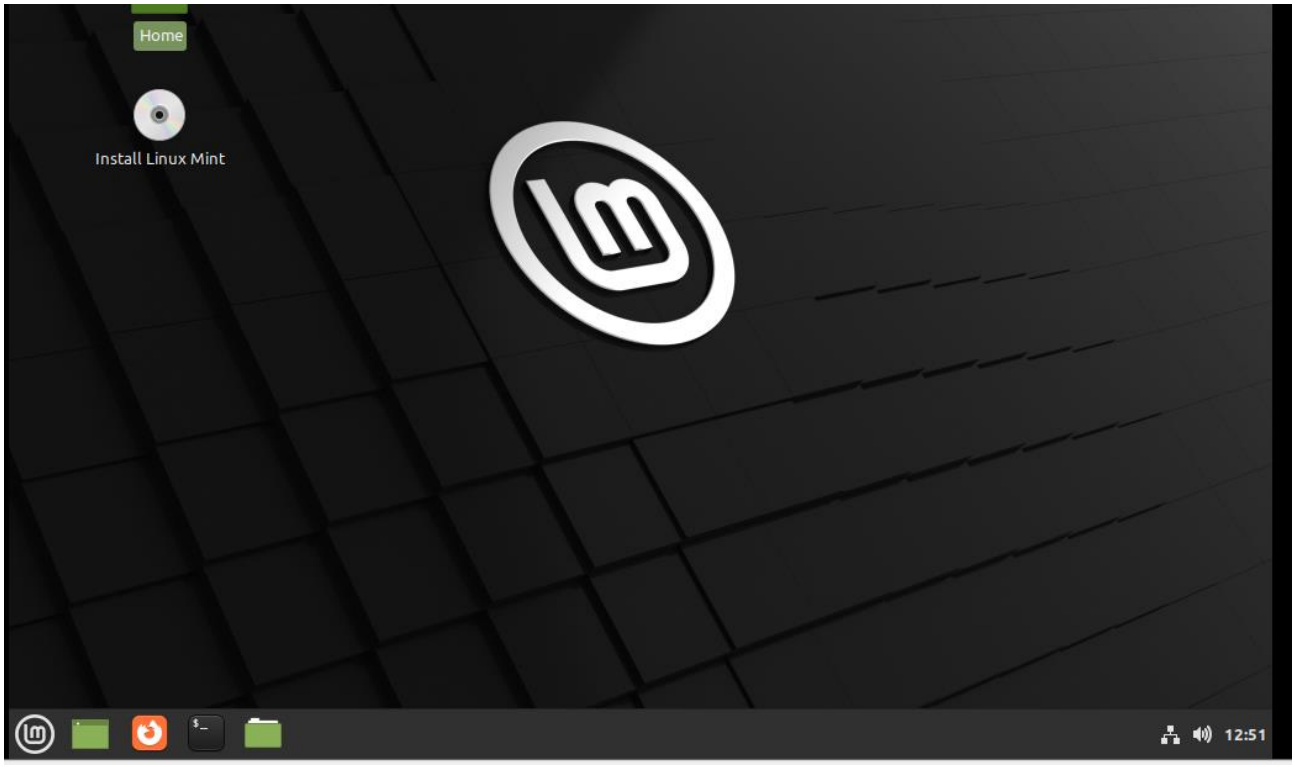
- Verificar que nuestro equipo cumple los requisitos mínimos de instalación del SO que vayamos a instalar.
- Planificar el particionado de los discos que vamos a usar.
- Si no disponemos del software de instalación podemos descargar una ISO de la versión que vamos a instalar, de la página oficial de Linux Mint.

Una vez realizado lo anterior arrancamos el equipo desde la ISO, si se trata de una máquina virtual, o CD/DVD.

Elegimos iniciar Linux Mint:



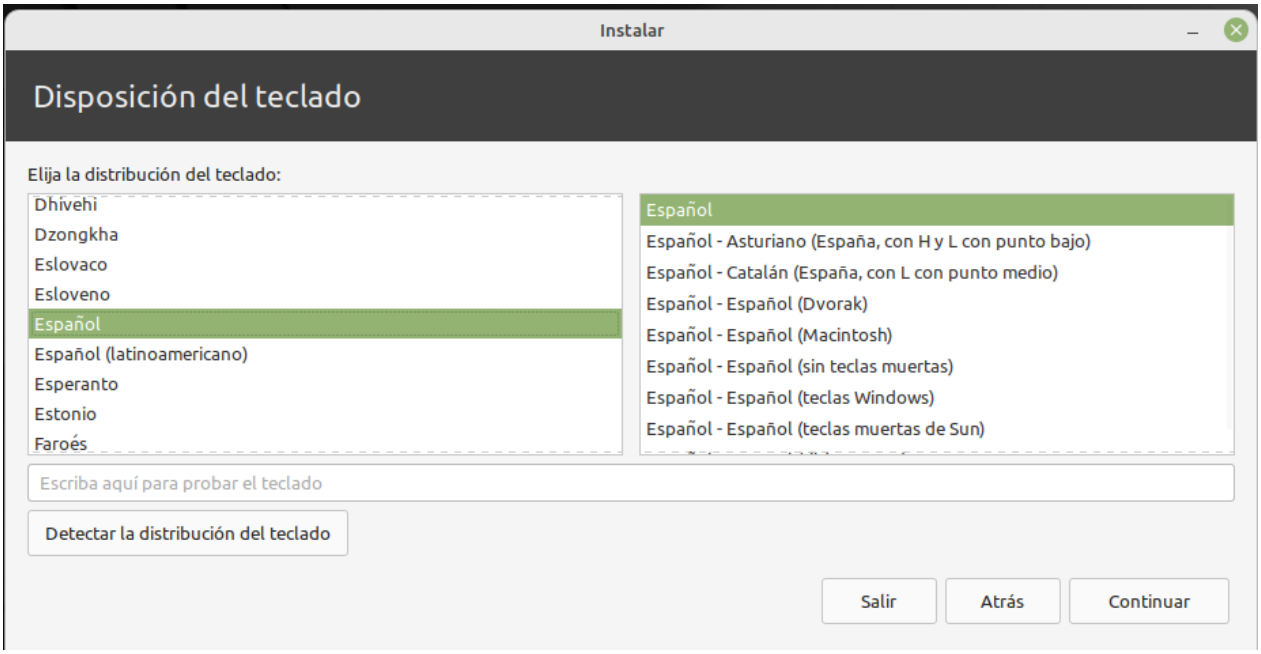
A continuación, nos mostrará el escritorio que estará funcionando en su modo "live", sin la necesidad de estar instalado en el equipo.



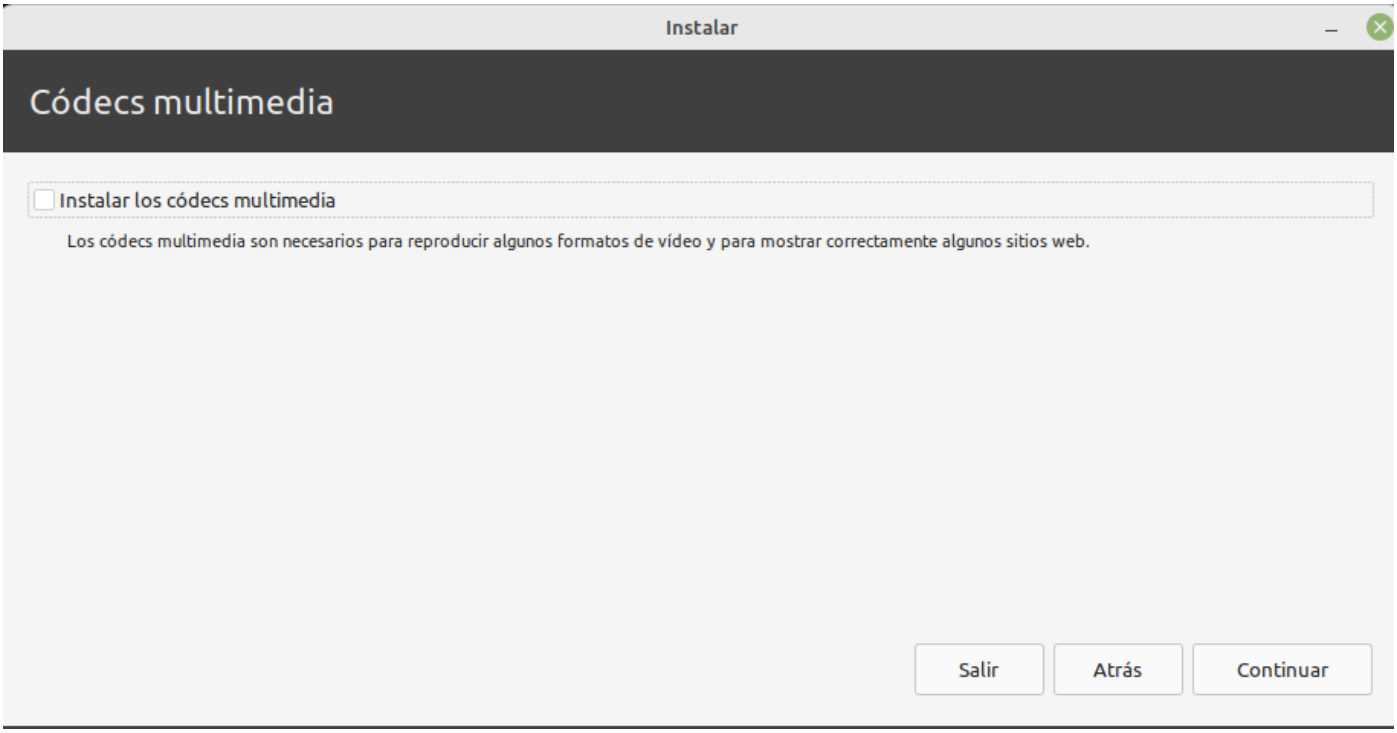
En el escritorio elegimos instalar Linux Mint y nos aparecerá este interfaz para elegir idioma.

Elegimos español y hacemos clic en Continuar.

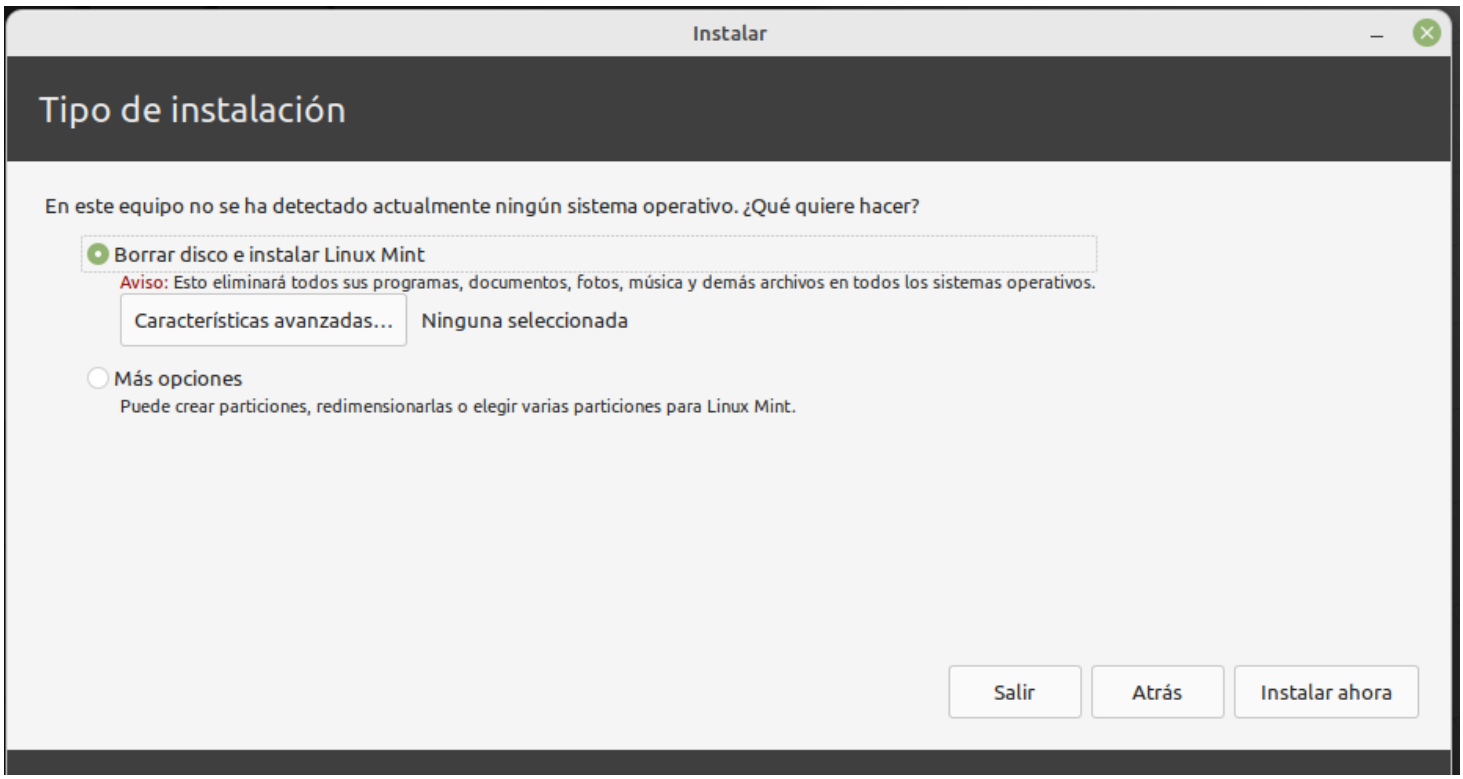
Elegimos el idioma del teclado. Podemos detectarlo por medio del programa de instalación.



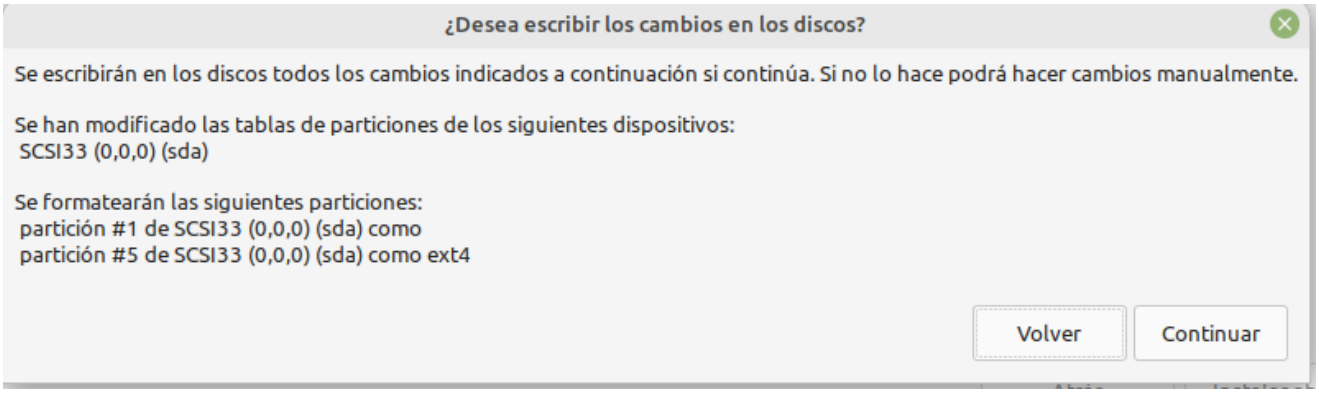
Podemos instalar los codecs multimedia, aunque no es necesario para nuestros propósitos.



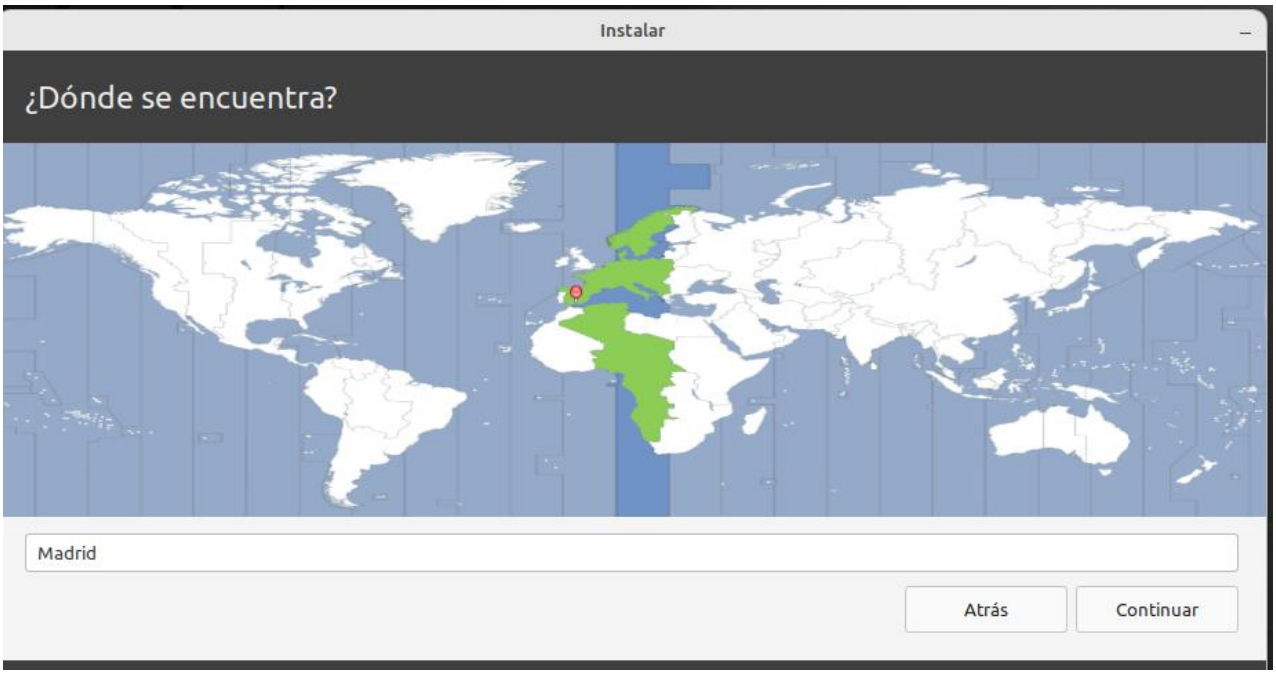
Nos informa de que no hay un SO instalado y nos da la opción de borrar el disco e instalar el SO o realizar el particionado manualmente. Elegimos borrar disco e instalar y hacemos clic en



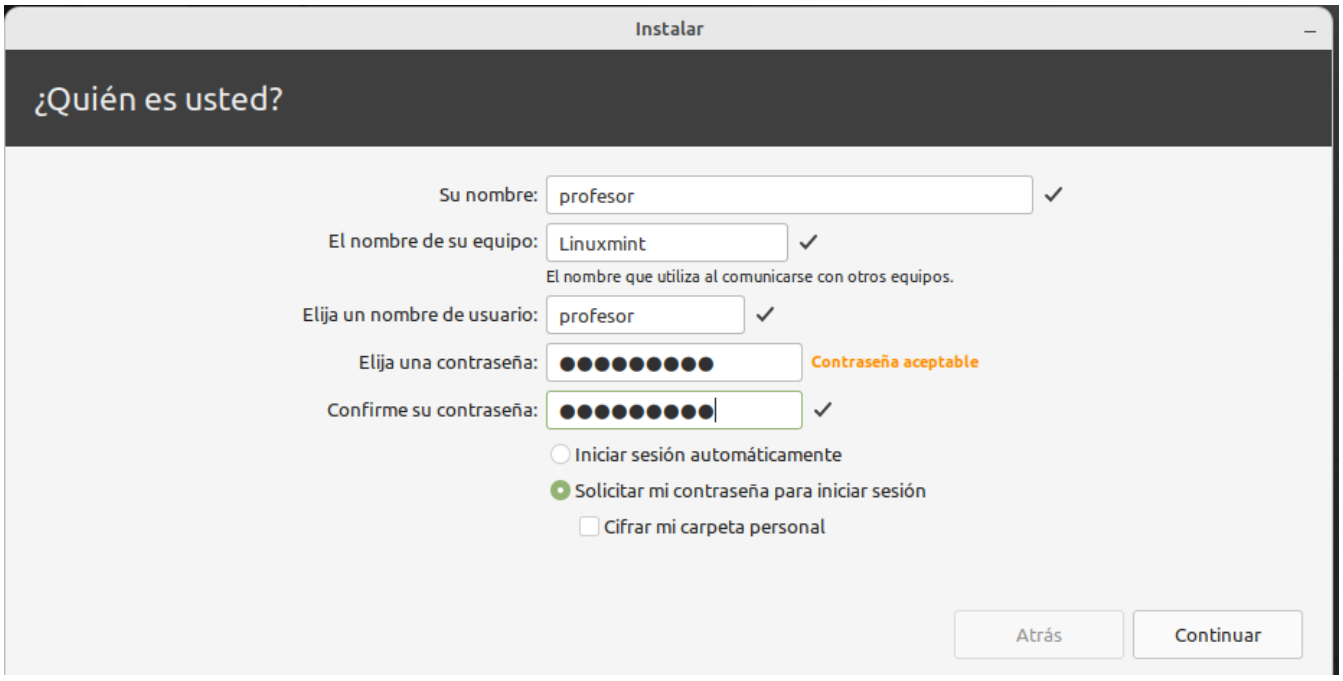
Nos avisa de que se escribirá en los discos para instalar y que formateará los discos.



Elegimos la zona horaria.

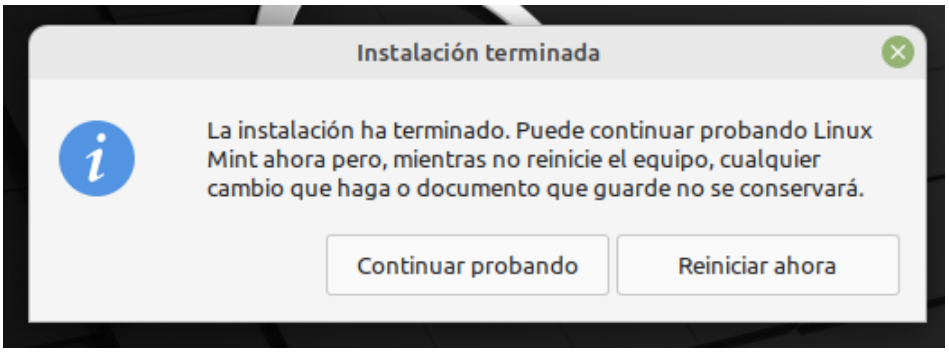


Nos solicita ahora el nombre del usuario con el que instalará el SO. Este usuario tendrá privilegios de administración. También nos solicita la contraseña del usuario y el nombre del equipo.

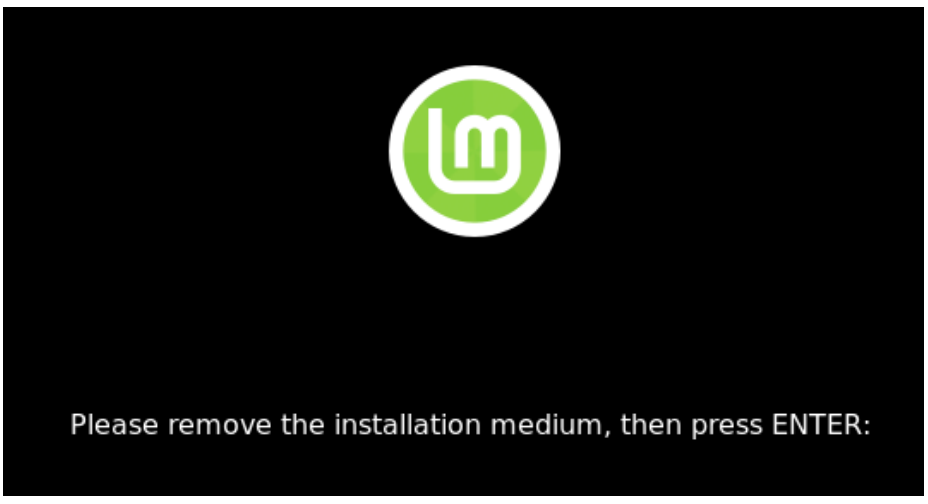


A continuación, el programa de instalación continua sin nuestra intervención.

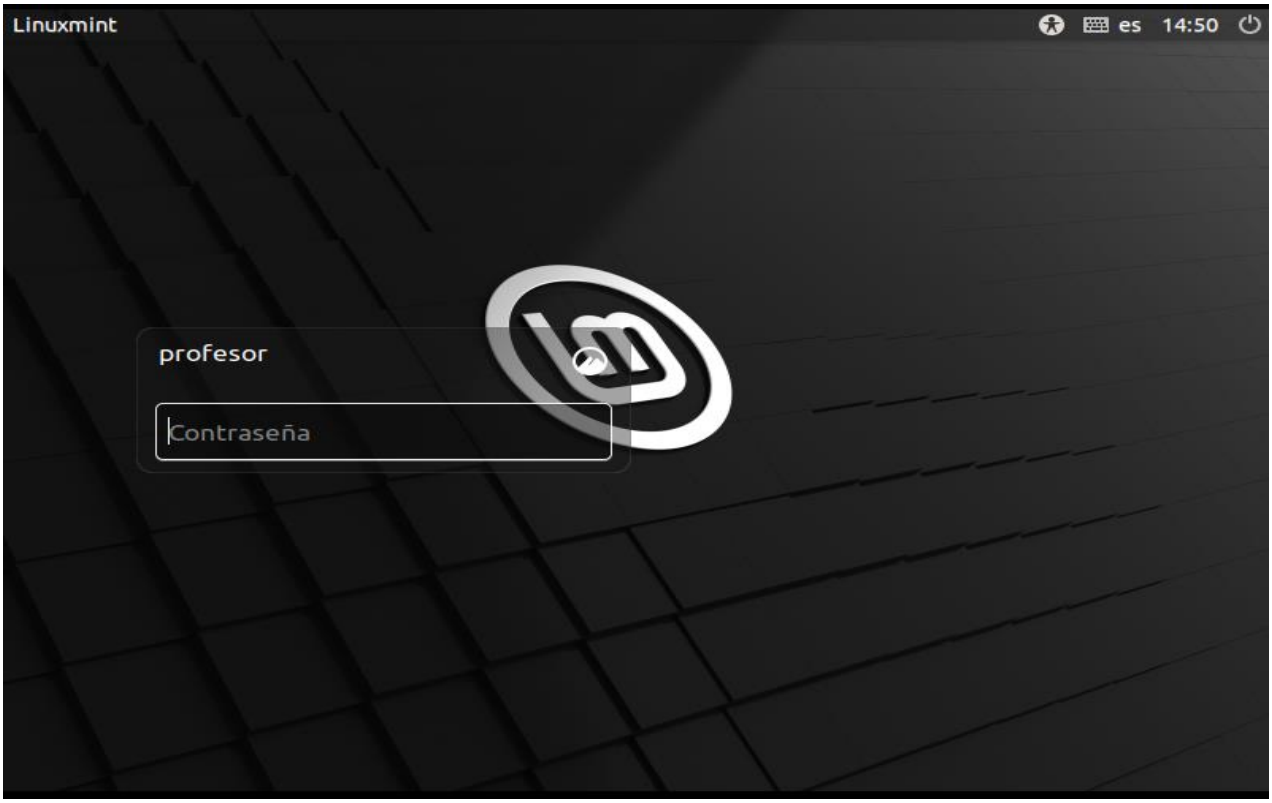
Finalmente acaba la instalación y nos pide reiniciar.



Nos solicita antes de arrancar de nuevo que quitemos la ISO de la unidad virtual de CD/DVD.



Tras el reinicio aparecerá la pantalla de login para iniciar sesión.



Finalmente, iniciamos sesión y se nos mostrará el escritorio.

5.3.- EL SISTEMA DE ARCHIVOS.

Cuándo se trabaja en Linux, debemos conocer la organización de la información en las distintas unidades de almacenamiento.

Al igual que en otros SSOO, la información se almacena en ficheros, organizados de forma lógica en directorios y subdirectorios, formando una estructura en árbol similar a los sistemas MS-DOS/Windows. La gran diferencia respecto a estos sistemas es que en Linux existe **un único árbol de directorios**, y no uno para cada partición o disco del sistema. Cualquier disco duro, partición, disquete, CD-ROM o dispositivo de almacenamiento, formará parte de esta estructura única.

A la operación por la cual un dispositivo de almacenamiento entra a formar parte del árbol de directorios del sistema se la denomina **montaje del dispositivo**.

Existe un directorio que contiene el resto de los archivos y directorios del sistema; se le llama **directorio raíz**. De él nace toda la estructura en árbol del sistema de directorios de Linux. El directorio raíz se representa siempre con el símbolo de barra inclinada "/".

Una característica que debemos conocer de Linux es que es sensitivo a las mayúsculas es decir que realiza **distinción entre mayúsculas y minúsculas**. Tanto para nombres de carpetas, archivos, nombres de programas, comandos y cualquier elemento, en Linux. Esto significa que no será el mismo fichero *Documentación*, que *documentación* o *DOCUMENTACIÓN*.

En los SSOO multiusuario, es fundamental una correcta gestión de los permisos de acceso sobre los elementos del sistema de archivos. En la mayoría de las distribuciones Linux, se distinguen dos tipos de usuarios: el superusuario, cuyo nombre de usuario, normalmente es **root,** con capacidad para administrar todo el sistema, y el resto de usuarios

El **directorio personal de un usuario** es el directorio asignado a cada usuario de un sistema Linux. En su directorio, un usuario puede crear archivos y directorios propios y puede establecer restricciones de acceso a otros usuarios. La mayoría de los usuarios trabajan regularmente con los archivos y directorios que crean bajo su directorio.

Habitualmente, la ubicación del directorio personal de un usuario es ***/home/usuario***, donde *usuario* es el nombre de acceso al sistema. El directorio *home* de un usuario es el directorio donde será ubicado el usuario al iniciar una sesión.

5.4.- LA JERARQUÍA ESTÁNDAR DE DIRECTORIOS.

En los sistemas Linux, las ubicaciones de los ficheros en el sistema siguen unas normas determinadas con el objeto de aumentar el nivel de organización. La mayoría de las distribuciones de Linux siguen el llamado estándar de jerarquía del sistema de archivos, **FHS** (***Filesystem Hierarchy Standard***). Algunos de los directorios importantes son los siguientes:

- **/bin y /usr/bin**: Contienen la mayoría de los ficheros ejecutables y los comandos más comunes del sistema Linux.
- **/boot**: Contiene los ficheros necesarios para el arranque del sistema. Por ejemplo, aquí se almacenan los ficheros del gestor de arranque si hubiera alguno instalado (LILO o GRUB). También se suelen almacenar las imágenes del kernel o núcleo del sistema.
- **/dev**: Almacena ficheros de dispositivos. Estos ficheros son la forma en la que Linux implementa los controladores de dispositivos.

- **/etc**: Contiene los ficheros de configuración de todo el sistema. Normalmente los ficheros que contiene son ficheros de texto. Además, suelen tener solo permisos de lectura para usuarios normales, es decir, solo un usuario con privilegios de administrador los puede modificar.
- **/home**: Contiene todos los directorios *personales* de los usuarios del sistema (salvo el de **root)**, por tanto, en función del número de usuarios y del uso que hagan del sistema, este directorio puede llegar a necesitar mucho espacio. En servidores se recomienda utilizar una partición separada para este directorio.
- **/lib** y **/usr/lib**: Contienen librerías compartidas del sistema.
- **/media**: cuando se monta un CD-ROM, una memoria USB o un disquete, se crea aquí automáticamente un subdirectorio:
- **/mnt**: Usado por defecto por el sistema para realizar el montaje de otros dispositivos de almacenamiento.
- **/proc**: Se corresponde con un sistema de archivos virtual creado por el kernel en memoria. Sirve de interfaz con los parámetros de configuración del kernel.
- **/root**: directorio *personal* de ***root.***
- **/sbin y /usr/sbin**: Contienen comandos y ficheros ejecutables normalmente ejecutados en tareas de administración, muchos de ellos sólo son ejecutables para ***root.***
- **/tmp**: Sirve para almacenar ficheros temporales.
- **/usr**: Destinado a almacenar las aplicaciones, con lo cual su tamaño puede ser elevado si existen muchos paquetes de software instalados. Puede ser una buena opción utilizar una partición separada para este directorio.
- **/var**: Directorio que contiene información variable en general, como colas de impresión, colas de envío y recepción de correos y news, archivos de registro y de eventos del sistema... En sistemas Linux utilizándose como servidores, este directorio puede necesitar mucho espacio, con lo cual es recomendable utilizar una partición propia.

5.5.- LOS ENLACES.

En los sistemas de ficheros usados en Linux, a cada archivo o directorio se le asigna un número identificador único llamado **inodo**.

Este número **inodo** representa una entrada en la tabla de inodos en donde se almacena toda la información importante del archivo: el propietario, grupo, permisos, tipo de archivo, fecha de última modificación, etc.

La única propiedad que no se almacena en la tabla de inodos es el propio nombre del archivo. El nombre del archivo se almacena en el directorio al que pertenece el archivo y es aquí donde se asocia el nombre del archivo a su inodo correspondiente. Esta estructura permite tener varios nombres, haciendo referencia al mismo archivo. Cuando se tienen varios nombres para un archivo, a cada nombre se le llama un **enlace duro** o **enlace hard**. Esto es útil cuando se quiere tener el mismo archivo en dos directorios diferentes.

La idea importante es que en el disco existirá un solo archivo (con un número de inodo único) con varios nombres y cualquier cambio que se haga utilizando cualquiera de los nombres, quedará reflejado en el archivo.

También se pueden crear los llamados **enlaces soft** o **simbólicos**. Este tipo de enlace tiene dos ventajas sobre el enlace *hard*: se puede crear un enlace *soft* a un directorio y se puede crear sobre un archivo o directorio de otro sistema de archivos.

El fichero o directorio se encuentra en un único punto del disco y los enlaces soft son un *puntero* contra él. **Cada enlace simbólico tiene su propio número de inodo** lo que permite hacer enlaces simbólicos entre distintos sistemas de ficheros.

Enlace duro*: diferentes nombres de archivos apuntando al mismo inodo*

Enlace soft*: diferentes inodos apuntando al mismo archivo.*

5.6.- LOS FICHEROS DE DISPOSITIVO.

Para Linux, cualquier dispositivo físico depende directamente de los llamados manejadores de dispositivos, que se integran en la estructura de ficheros del sistema dentro del directorio */dev*. De esta forma, el acceso a los diferentes dispositivos físicos se hace de una forma homogénea; la entrada/salida se realiza siempre como si se accediera a ficheros.

Para que un dispositivo pueda ser "manejado" a través de estos ficheros es necesario que haya un *driver* o controlador a bajo nivel instalado. Por tanto, todos los ficheros que contiene el directorio /dev, realmente no son ficheros de disco, sino que son manejadores de dispositivos creados por los *drivers*. Los nombres de estos "ficheros" siguen un estándar para identificarlos con los dispositivos físicos a los que están asociados.

Los más comunes son:

Unidad CD-ROM o DVD:

/dev/cdrom /dev/dvd

Dispositivos conectados a un bus IDE:

/dev/hda	Primer disco duro del canal IDE 1
/dev/hda1	Primera partición
/dev/hdb	Segundo disco duro del canal IDE 1
/dev/hdb1	Primera partición
/dev/hdc	Primer disco duro del canal IDE 2
/dev/hdd	Segundo disco duro del canal IDE 2

Dispositivos SCSI :

/dev/sda	Dispositivo 1
/dev/sda1	Primera partición
/dev/sda2	Segunda partición

Puertos serie:

/dev/ttyS0

Impresoras por puerto paralelo:

/dev/lp0

Ratón:

/dev/psaux

Dispositivos conectados al bus USB:

/dev/usb

Otros:

/dev/stdin	Entrada estándar (por defecto, el teclado)
/dev/stdout	Salida estándar (por defecto, la pantalla)
/dev/stderr	Salida estándar de error (por defecto, la pantalla)
/dev/null	Salida nula

5.7.- PROCESO DE ARRANQUE Y DETENCIÓN DEL SISTEMA.

Tradicionalmente el proceso de arranque de Linux estaba dirigido por el programa **init**. Desde Ubuntu 18.04 no es init el programa que inicia el sistema, si no que se trata de systemd, un reemplazo de init, que mantiene compatibilidad con los scripts de SysV.

Systemd es un conjunto de demonios de administración de sistema, bibliotecas y herramientas diseñados como una plataforma de administración y configuración central para interactuar con el kernel del SO. Se usa en Ubuntu para el inicio del sistema.

5.7.1.- PROCESO DE ARRANQUE CON SYSV

En el arranque tradicional que se ejecuta con **init**, una vez que se ha encontrado el kernel y se ha iniciado, el sistema comienza a cargarse, se inicia el HW, los discos están preparados, se asignan direcciones IP, se inician los servicios y se realizan otras muchas tareas, para todo ello Linux ejecuta el programa ***init***. las tareas más importantes que ejecuta ***init*** son:

- Comprueba los sistemas de ficheros.
- Monta los sistemas de ficheros permanentes.
- Activa la zona de memoria de intercambio o swap.
- Activa los demonios o servicios del sistema
- Activa la red
- Inicia los demonios o servicios de red
- Limpia los ficheros temporales
- Finalmente, habilita el *login* a los usuarios del sistema.

El proceso *init* es el proceso estándar que utilizan para apagar y arrancar equipos basados en SysV. SysV es un modo de definir que estado debe tener el equipo en un momento determinado. Para ello se emplea un concepto llamado modo de ejecución (o *runlevel)*

SysV utiliza 7 modos de ejecución que van del 0 al 6 y cada distribución usa los modos de ejecución para diferentes fines, aunque hay varios niveles que son comunes. Los niveles comunes son el 0 que se usa para apagar el equipo; el 1 que es el modo monousuario y el 6 que se usa para reiniciar.

- Nivel 0: Apagado.
- Nivel 1: Monousuario (sólo usuario root; no es necesaria la contraseña). Se suele usar para analizar y reparar problemas.
- Nivel 2: Multiusuario sin soporte de red.
- Nivel 3: Multiusuario con soporte de red.
- Nivel 4: Como el runlevel 3, pero no se suele usar
- Nivel 5: Multiusuario en modo gráfico (X Windows).
- Nivel 6: Reinicio.

Cada nivel de ejecución tiene asociado un directorio donde se especifican los servicios que se deben ejecutar o parar. Por ejemplo, el directorio */etc/rc0.d* o el directorio */etc/rc1.d*

Si mostramos, por ejemplo, el contenido de */etc/rc3.d con $ ls -l* tendremos la siguiente salida:

```
lrwxrwxrwx 1 root root 14 ene 27  2022 S01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 14 ene 27  2022 S01cups -> ../init.d/cups
lrwxrwxrwx 1 root root 22 ene 27  2022 S01cups-browsed -> ../init.d/cups-browsed
lrwxrwxrwx 1 root root 14 ene 27  2022 S01dbus -> ../init.d/dbus
lrwxrwxrwx 1 root root 21 ene 27  2022 S01grub-common -> ../init.d/grub-common
lrwxrwxrwx 1 root root 17 ene 27  2022 S01hddtemp -> ../init.d/hddtemp
lrwxrwxrwx 1 root root 20 ene 27  2022 S01irqbalance -> ../init.d/irqbalance
lrwxrwxrwx 1 root root 20 ene 27  2022 S01kerneloops -> ../init.d/kerneloops
lrwxrwxrwx 1 root root 17 ene 27  2022 S01lightdm -> ../init.d/lightdm
lrwxrwxrwx 1 root root 23 ene 27  2022 S01lvm2-lvmpolld -> ../init.d/lvm2-lvmpol
```

Como se puede ver en esta salida, el directorio contiene enlaces simbólicos a scripts del directorio */etc/init.d* . Cada enlace lleva una letra (K o S) y un número al principio. El número establece el orden en el que se van a ejecutar los servicios mientras que la letra "S" significa iniciar y la letra "K" parar el servicio correspondiente. Cada uno de estos scripts puede aceptar un argumento que suele ser: start, stop, status, restart o reload. Se pueden ejecutar manualmente estos scripts. Por ejemplo, para parar el servidor de páginas web podemos ejecutar:

 # /etc/init.d/httpd stop

O también podemos usar el comando *service* para administrar los servicios:

 # service httpd stop

Una vez que se lanzan todos los scripts que establece el nivel de ejecución se procesa el fichero ***/etc/rc.local*** en el que se pueden escribir todos los comandos o scripts que se quieran lanzar al iniciar el sistema. Con el actual ***systemd*** */etc/*rc.local no funciona, aunque se puede hacer funcionar construyendo un servicio para él.

5.7.2.- EL DEMONIO SYSTEMD

En systemd se usa en lugar de los niveles de ejecución lo que se conoce como targets u objetivos, que mantienen una correspondencia con los runlevel de SysV:

- Runlevel 0 coincide con poweroff.target y runlevel0.target es un enlace simbólico a poweroff.target. Su misión es apagar y encender el sistema
- Runlevel 1 coincide con rescue.target y runlevel1.target es un enlace simbólico a rescue.target. Su misión es establecer una shell de rescate
- Runlevel 2 coincide con multi-user.target y runlevel2.target es un enlace simbólico a multiuser.target. Su misión es establecer un entorno no gráfico de multiusuario.
- Runlevel 3 es emulado por multi-user.target y runlevel3.target es un enlace simbólico a multiuser.target. Su misión es establecer un entorno no gráfico de multiusuario.
- Runlevel 4 es emulado por multi-user.target y runlevel4.target es un enlace simbólico a multiuser.target. Su misión es establecer un entorno no gráfico de multiusuario.
- Runlevel 5 es emulado por graphical.target y runlevel5.target es un enlace simbólico a graphical.target. Su misión es establecer un entorno gráfico de multiusuario.
- Runlevel 6 es emulado por reboot.target y runlevel6.target es un enlace simbólico a reboot.target. Su misión es apagar y encender el sistema.

Para ver el target actual se usa: **systemctl get-default.**

Para establecer el target por defecto usamos: **systemctl set-default multi-user.target.**

Para cambiar a otro nivel de ejecución, por ejemplo al runlevel 3 usamos**: systemctl isolate multiuser.target.**

Usando el comando **systemctl** también podemos ver el estado de un servicio, pararlo, reiniciarlo o ver la lista de servicios que están funcionando en el sistema.

5.8.- ADMINISTRAR LINUX

Existen diferentes formas de administrar el sistema. Básicamente se puede administrar de tres maneras distintas:

- **Interfaces gráficas**: Hay diferentes interfaces gráficas que permiten administrar el sistema de forma fácil y sencilla.
- **Terminal del sistema**: Se puede administrar totalmente el sistema desde un terminal usando la línea de comandos. Esta forma tiene una gran flexibilidad y la ventaja del uso de scripts para facilitar y automatizar la administración.
- **Ficheros de configuración**: La modificación directa de los ficheros de configuración es la forma en la que se tiene mayor control del sistema. Como desventaja se necesita un alto conocimiento del sistema y su funcionamiento.

Cualquiera de estos tres métodos es adecuado y no se puede decir que ninguno sea mejor que otro. Lo mejor es conocer los 3 y usar el adecuado en cada ocasión.

5.8.1.- CONSOLAS VIRTUALES.

Para ofrecer varios terminales al usuario desde un mismo conjunto de pantalla/teclado/ratón, Linux proporciona un cierto número de consolas virtuales.

Se dispone tanto de varias terminales de "texto" para las tareas que deben efectuarse desde una línea comandos, como de una interfaz gráfica

Estas consolas virtuales, que generalmente son seis o siete en las distribuciones como Ubuntu, se componen de cinco a seis terminales de texto y una consola gráfica.

Se invocan con las teclas de función **[F1]** a **[F7]** del teclado, normalmente o **[F1]** o **[F7]** es la interfaz gráfica. Para pasar de una a otra, hay que pulsar simultáneamente las teclas **[Ctrl]**+**[Alt]**+**[Fn]** donde **n** es el número de la consola virtual.

5.9.- EL ESCRITORIO.

El escritorio suele ser la primera pantalla que aparece tras iniciar sesión en cualquier distribución GNU/Linux. Dependiendo de la distribución que utilicemos el escritorio tendrá un aspecto u otro, pero en la mayoría de las distros, podremos configurar y personalizar el comportamiento y el aspecto del escritorio fácilmente. Podemos personalizar el fondo de escritorio, aplicar distintos temas o personalizar el salvapantallas y su funcionamiento.

5.10.- ACTUALIZACIÓN DEL SISTEMA Y LAS APLICACIONES.

En las distintas distribuciones de sistemas GNU/Linux existen formas de actualizar el sistema bien sea desde la interfaz gráfica o desde la consola de comandos.

En la mayoría de las distribuciones de Linux existen dos tipos de versiones; las versiones LTS (Long term support) que aparecen cada dos años y que nos dan soporte para 5 años y las actualizaciones que no dan soporte a largo plazo y que suelen aparecer cada 6 meses aproximadamente, estas últimas suelen tener soporte para 9 meses.

La mayoría de las distros nos permitirán elegir entre instalar:

- Actualizaciones importantes de seguridad.
- Actualizaciones recomendadas.
- Actualizaciones sin servicio técnico.

Además, tendremos normalmente las opciones de:

- Comprobar ecualizaciones automáticamente.
- Cuando hay actualizaciones de seguridad.
- Cuando hay otras actualizaciones.
- Notificarme una versión nueva del sistema.

Normalmente, podremos elegir también:

- Elegir la frecuencia con la que se buscarán actualizaciones.
- Descargar e instalar automáticamente las actualizaciones.
- Descargar las actualizaciones en segundo plano.
- Elegir cuando descargar las actualizaciones de forma manual.
- Notificar cuando haya nuevas actualizaciones del sistema versiones LTS.
- Notificar cuando haya versiones normales sin soporte a largo plazo.

Todas estas opciones se suelen encontrar en el interfaz gráfico en las preferencias o en el menú de sistema, el lugar donde se encuentren dependerá de la distribución que usemos.

Desde el terminal de Linux también se pueden hay varios comandos que puedes usar desde Terminal para actualizaciones de Linux, por ejemplo, la aplicación apt-get nos permitirá.

- **Apt-get update** actualiza los repositorios que definimos en el archivo **/etc/apt/sources.list**. Se actualizan los listados de paquetes disponibles en las fuentes definidas en el archivo sources.list, pero no instala nada. Lo que hace es ver si hay nuevas versiones de los paquetes que tenemos en las fuentes definidas, pero sin modificar nada en nuestro sistema.
- **Apt-get upgrade:** Instala, previa confirmación todos los paquetes instalados que tengan alguna nueva versión. Lo que hace es una actualización del sistema con todas las posibles actualizaciones que pudiera haber. No sólo actualiza nuestro sistema operativo, sino que también las aplicaciones que están contenidas en los repositorios y que tengan una nueva versión para instalar.

Para la actualización de las aplicaciones existen gestores de paquetes que o bien de forma gráfica, como **Synaptic**, o bien desde el terminal nos permite actualizar e instalar las aplicaciones disponibles para nuestro sistema operativo. Por ejemplo, Ubuntu dispone del **Centro de Software** que nos permite instalar, actualizar o buscar el software disponible para nuestra distribución.

5.10.1.- DRIVERS EN LINUX.

Los drivers son programas que se instalan en el SO, que permiten comunicar a los periféricos y otros componentes del equipo con el SO. Existen multitud de controladores, por ejemplo, controlador de la impresora, de la tarjeta gráfica, de la placa base etc.

La mayoría de ellos vienen instalados en Linux por defecto, aunque normalmente se instalan por defecto los controladores libres y no los privativos. Así, la mayor parte de los dispositivos que usemos serán compatibles con nuestro sistema, ya sean teclados, ratones, auriculares, impresoras y los componentes del ordenador.

Normalmente desde el interface gráfico, dependiendo de la distribución, podremos instalar nuevos controladores para los dispositivos. Por ejemplo, en Ubuntu podemos hacerlo desde ***orígenes del Software.***

Desde el terminal también podremos instalar cualquier driver conociendo el nombre de los paquetes en los que se distribuye el driver que queremos y usando por ejemplo **apt.**

Normalmente es suficiente con los drivers libres instalados en Linux por defecto, pero usando, en algunos casos, drivers propietarios podemos usar de forma más eficiente algunos dispositivos.

5.11.- USUARIOS EN LINUX

Linux es un SO multiusuario lo que significa que permite a varios usuarios utilizar el sistema simultáneamente a través de la línea de comandos o por conexiones remotas. Linux controla el acceso al equipo y sus recursos mediante las cuentas de usuarios y grupos.

En GNU/Linux existen tres tipos de usuarios:

- **Root:** Es el usuario más importante ya que es el administrador del sistema. Se aconseja usar esta cuenta para tareas específicas de administración y no usarla para el resto de las tareas.

- **Usuarios normales:** Son aquellos que pueden iniciar sesión en el equipo y tienen una funcionalidad limitada tanto en los comandos que pueden usar como en los ficheros a los que tienen acceso.

- **Usuarios asociados a servicios:** Este tipo de usuarios no pueden iniciar sesión en el sistema. Permiten establecer los privilegios de un determinado servicio.

5.11.1.- EL USUARIO ROOT O SUPERUSUARIO

El usuario root en Linux es el usuario que posee mayor nivel de privilegios. Es el único que tiene privilegios sobre todo el sistema en su globalidad, así como el responsable de las tareas administrativas.

Cuando se quiere llevar a cabo una acción que requiera permisos de superusuario hay que conceder estos privilegios.

No todas las distribuciones hacen el mismo uso de la cuenta de superusuario, en algunas el usuario root viene deshabilitado y para usarlo hay que habilitarlo o en otras root se puede usar directamente.

Existen algunos comandos para realizar tareas usando al superusuario o con los privilegios del superusuario.

Básicamente es importante conocer dos comandos que nos permiten ejecutar tareas como si fuésemos otro usuario: ***sudo*** y ***su.***

- ***sudo:*** Permite al usuario actual ejecutar aplicaciones o procesos bajo los privilegios de otro usuario distinto.

  El uso más general que se le suele dar es el de permitir ejecutar procesos bajo los permisos de root. Su nombre viene justamente de "superuser do".

  Se utiliza acompañada de comandos que requieran permisos elevados. Su uso consiste únicamente en ser introducido delante del comando en cuestión, para indicar que ese comando se va a ejecutar bajo los permisos de otra cuenta, por defecto, root.

  Sudo requiere únicamente las credenciales del propio usuario con el que estás logueado.

  Para que un usuario pueda usar ***sudo*** debe aparecer en el fichero de configuración ***/etc/sudoers***.

  No se recomienda el uso de sudo para ejecutar comandos que puedan ser ejecutados por un usuario estándar.

- ***su:*** Se usa para cambiar el usuario que está usando la línea de comandos, de forma que se puedan ejecutar acciones en nombre de otro usuario. Inicia una Shell del usuario al que se quiere cambiar. Requiere conocer la contraseña del usuario al que se quiere cambiar. Su nombre viene de "Switch user".

5.12.- SHELL, EL INTERPRETE DE COMANDOS

El Shell o intérprete de comandos es la herramienta que permite al administrador ejecutar tareas en modo comando.

El intérprete de comandos es el programa que recibe lo que se escribe en la terminal y lo convierte en instrucciones para el sistema operativo. En otras palabras, el objetivo de cualquier intérprete de comandos es ejecutar los programas que el usuario teclea en el prompt del mismo.

El prompt es una indicación que muestra el intérprete para anunciar que espera una orden del usuario. Cuando el usuario escribe una orden, el intérprete ejecuta dicha orden.

En el mundo Linux/Unix existen diferentes familias de Shells. Estas se diferencian entre sí básicamente en la sintaxis de sus comandos y en la interacción con el usuario.

Nosotros usaremos y hablaremos de la shell Bash.

5.13.- SINTAXIS DE LOS COMANDOS.

Los comandos en general en GNU/linux tienen la siguiente sintaxis:

```
# comando arg1 arg2 ... argn
```

En la ``línea de comandos", se introduce el *comando* seguido de uno o varios *argumentos*. Así, el intérprete ejecutará el *programa* con las opciones que se hayan pasado por medio de los argumentos.

En los comandos, también se puede utilizar los comodines:

El *asterisco "*"* es equivalente a uno o más caracteres en el nombre de un archivo.

Ejemplo: *ls *.c* lista todos los archivos con extensión c.

El *signo de interrogación (?)* es equivalente a un único carácter.

Ejemplo: *ls curso.te?* lista el archivo curso.tex completando el último carácter.

Un *conjunto de caracteres entre corchetes* es equivalente a cualquier carácter del conjunto.

Ejemplo: l*s clinux.t[aeiou]x* lista clinux.tex seleccionando la *e* del conjunto.

5.14.- VARIABLES Y VARIABLES DE ENTORNO

Una ***variable*** es un nombre asociado a una cadena de caracteres, que puede contener un valor. Dependiendo de la variable, su utilidad puede ser distinta.

Las **variables de entorno** son utilizadas por el SO para configurar el funcionamiento del mismo. Además, los programadores de scripts también definen sus propias variables y usan las variables de entorno para el desarrollo de scripts.

Algunas de las variables de entorno más importantes son

HOME Directorio personal.

HOSTNAME Nombre de la máquina.

PATH Lista de directorios donde buscar los programas.

PS1 Prompt.

SHELL Intérprete de comandos por defecto.

USER Nombre del usuario.

La forma de definir una variable de entorno cambia con el intérprete de comandos, nosotros usaremos bash para el desarrollo de este tema. En esta shell:

~ $ export VARIABLE=Valor

Por ejemplo, para definir el valor de la variable PRUEBA:

~ $ export PRUEBA=’esto es una prueba’

Con este comando habríamos creado una variable llamada PRUEBA con el valor “esto es una prueba”. El **alcance o scope** de esta variable es global a la sesión de la shell, es decir cuando cerremos el terminal la variable desaparecerá.

Para hacer que esta variable permanezca en el entorno una vez se ha cerrado el terminal que la inició, debemos modificar el fichero **/home/usuario/.bashrc** (es un archivo oculto cuyo nombre empieza por .) y añadirla como una nueva línea:

export PRUEBA=’esto es una prueba’

También se pueden crear variables de alcance global en el fichero **/etc/environment** añadiendo la línea:

PRUEBA=”esto es una prueba”

Si declaramos la variable en **/home/usuario/.bashrc** esta variable solo se verá para el usuario propietario de /home/usuario.

Si las declaramos en **/etc/environment** será visible para todos los usuarios del sistema.

Las variables se usan para guardar información y para leer la información que hemos guardado. Para poder ver el valor de una variable se coloca delante de su nombre el símbolo “$”.

Ejemplo: ~$ echo $PRUEBA

5.15.- LAS PROPIEDADES DEL SISTEMA DE ARCHIVOS.

Los elementos que se encuentran en el sistema de archivos, es decir, tanto archivos como directorios, poseen una serie de características o propiedades que pueden visualizarse con el comando *ls –l*:

Los campos que aparecen en este listado son los siguientes:

Los archivos tienen asociado un nombre de usuario, que es su propietario y un nombre de grupo.

Campo	Descripción
Permisos	Define los permisos sobre el archivo o directorio (descrito a continuación).
NL	Número de enlaces del archivo. Si es un directorio, indica el número de subdirectorios.
Prop	Nombre del propietario o dueño del archivo o directorio.
Grupo	Nombre del grupo al que pertenece el archivo o directorio.
Tam	Tamaño del archivo, en bytes.
Fecha y Hora	Indica la fecha y la hora de creación o modificación del archivo.
Nombre	Nombre del archivo.

5.16.- EL SISTEMA DE ARCHIVOS EN MODO ORDEN

Para trabajar en modo orden, selecciona **Terminal** de **Accesorios** de **Aplicaciones** y verás una pantalla en la que el sistema presenta el **prompt** indicando que ya está preparado para aceptar los comandos que desees.

A continuación, se presentarán los comandos básicos para trabajar con la estructura de directorios del sistema:

- **pwd:** Como ya sabéis al directorio donde se encuentra el usuario se le llama **directorio actual** o **directorio activo**.

El comando *pwd* muestra en pantalla la ruta completa del directorio actual o activo. En muchas ocasiones, el *prompt* del sistema se configura para mostrar el directorio activo, pero si no es así, se puede utilizar el comando *pwd* para visualizarlo.

- **cd [nombre_directorio]:** Cambia el directorio actual por el especificado como parámetro. Utilizado sin parámetros, cambia al directorio *home* del usuario.

A la hora de escribir rutas para el comando *cd* o para cualquier otro comando, recuerda la utilidad del tabulador que proporciona la *shell* de comandos.

- **ls [-opciones] [nombre_directorio]:** Visualiza el contenido del directorio especificado como parámetro. Este comando tiene multitud de opciones para alterar el tipo de información mostrada. Por ejemplo, la opción *–l* muestra información extendida sobre cada archivo o directorio. Ejemplos:

ls –l Lista extendida de archivos y directorios del directorio actual.

ls –l/var/log Lista extendida del directorio /var/log.

ls –a Lista de todos los archivos y directorios, incluidos los ocultos, que en Linux empiezan por el carácter punto ".".

ls –R Lista de archivos y directorios y de todos los directorios dentro de él.

ls –C Lista de archivos en columnas.

- **mkdir *directorio:*** Crea un nuevo directorio.
- **rmdir *directorio:*** Borra un directorio vacío.

Los comandos anteriores se utilizan para trabajar con las estructuras de directorios en un sistema Linux. Otros comandos generales son:

- **logout:** El sistema finaliza la sesión y vuelve a mostrar la petición de login para una nueva sesión.
- **exit** :El comando *exit* es equivalente y se utiliza para el cierre del terminal.
- **man:** Uno de los comandos más útiles del modo texto es el comando *man*. La forma de ejecutarlo es teclear *man* seguido del nombre del comando sobre el que queremos obtener ayuda. Por ejemplo:

$ man ls

La ayuda que ofrece el comando *man* se presenta paginada y formateada con algún programa editor de textos que esté instalado en nuestro sistema. Para salir de nuevo al prompt del sistema, pulsar la tecla [**q**].

5.17.- COMANDOS PARA EL TRABAJO CON EL SISTEMA DE ARCHIVOS.

A continuación tenéis algunos de los comandos más importantes para el manejo de archivos en un sistema Linux. Los parámetros opcionales se especifican entre corchetes []:

- **cp archivo1 [archivo2 ... archivoN] directorio**

Copia archivos y directorios. Si el último argumento es un directorio, *cp* copia cada archivo fuente a ese directorio. Si el último argumento es un archivo, *cp* copia el primer argumento, que deberá ser un solo archivo, en el fichero destino.

- **mv archivo1 [archivo2 ... archivoN] destino**

Mueve o renombra archivos o directorios. Si el último argumento es un directorio, *mv* mueve cada uno de los ficheros anteriores a ese directorio. Si se pasan como argumento dos nombres de archivo (o dos nombres de directorios), renombra el primero al segundo.

- **rm archivo1 [archivo2 ... archivoN]**

Borra archivos. Con la opción –r se borra la estructura entera de un directorio, incluidos los subdirectorios.

- **cmp archivo1 archivo2**

Compara dos archivos de cualquier tipo y escribe el resultado en la salida estándar.

5.18.- LOS COMODINES.

La mayoría de los comandos para el trabajo con archivos se pueden ejecutar sobre más de un fichero. Para ello, se utilizan como nombres de archivos palabras patrón. Una palabra patrón o comodín, es cualquier palabra que contenga alguno de los caracteres comodín de la siguiente tabla:

Carácter	Descripción
*****	Sustituye varios caracteres o ninguno.
?	Sustituye a un único carácter.
[...]	Sustituye cualquiera de las alternativas entre corchetes.
[!...]	Sustituye cualquier carácter menos los indicados en la lista entre corchetes.

Ejemplos:

$ rm prueba*	Elimina todos los archivos que comienzan por la palabra *prueba*.
$ rm *prueba	Elimina todos los archivos que acaban en la palabra *prueba*.
$ rm *prueba*	Elimina todos los archivos que contengan la palabra *prueba*.
$ rm prueba??	Elimina todos los archivos que empiecen por *prueba* y tengan 8 caracteres.

5.19.- REDIRECCIÓN DE LA ENTRADA/SALIDA.

Todos los comandos tienen por defecto una entrada estándar (teclado) y dos salidas: la salida estándar (pantalla) y la salida de error (pantalla).

En un comando podemos sustituir la entrada y salida estándar por otro dispositivo utilizando los caracteres `>`"y `<`". Por ejemplo podemos hacer que un comando lea su entrada de un archivo en vez de la línea de comandos o que un comando mande su salida a un archivo de salida, respectivamente. Por ejemplo:

Entrada: La orden ls toma su entrada del archivo directorio.txt

ls -l < directorio.txt

Salida: La orden ls manda su salida al archivo listado.txt

ls -l > listado.txt

Es importante resaltar que el carácter de redirección de salida `>`"destruirá el archivo al cual apunta, si este existe, para ser reemplazado por uno nuevo con los resultados del proceso. Si se desea anexar la información a uno ya existente debe usarse doble carácter `>>`":

5.20.- TUBERÍAS O PIPES.

También podemos hacer que un comando lance su salida para que esta sea la entrada de otro comando, esto es lo que se conoce como una tubería o pipe. Para ello se usa el carácter " | ".

Por ejemplo: la siguiente tubería mandaría la salida del comando more como entrada del comando grep y su efecto sería que listaría todas las líneas del archivo /etc/passwd que contengan la palabra root.

more /etc/passwd | grep root

De esta forma podemos encadenar los resultados de los comandos, aumentando así su utilidad.

Por supuesto se pueden agrupar más comandos construyendo la misma tubería:

comando1|comando2|comando3|....

5.21.- FILTRADO DE TEXTOS

Para conocer **el** contenido de los ficheros desde la línea de comandos, existen algunos comandos básicos muy utilizados, como ***less*** y ***more*** que permiten visualizar en pantalla el contenido paginado de un archivo.

El filtrado de textos consiste en seleccionar la salida de un comando como less o more, eligiendo para mostrar solamente las líneas que cumplen unas características concretas. Por ejemplo podemos usar el comando grep para mostrar solo las líneas del fichero /etc/passwd que contengan la palabra profe:

less /etc/passwd | grep profe

Dependiendo del comando que usemos para filtrar podremos filtrar unas u otras condiciones. Usando, por ejemplo, ***head*** y ***tail*** que muestran respectivamente las n primeras o las n últimas líneas de un fichero:

less /etc/passwd | head -n 1 mostraría la 1ª línea de /etc/passwd

less /etc/passwd | tail -n 1 mostraría la última línea de /etc/passwd

Un comando muy útil para filtrar es el comando ***cut*** que nos permite obtener partes de una línea de una salida que cumplen algún criterio, en lugar de toda la línea.

more /etc/passwd | cut -d ":" -f1

Esta tubería devolvería la parte de las líneas de /etc/passwd que está delimitada por ":" quedándose solo con la parte correspondiente hasta que se encuentra por 1ª vez con el delimitador. Es decir seleccionaría solo el primer campo o columna de cada línea de /etc/passwd.

Podemos también ordenar la salida de datos usando el comando ***sort*** que ordenará la salida.

more /etc/passwd | sort

Esta tubería devolverá ordenadas alfabéticamente las líneas de /etc/passwd

5.22.- PROCESOS

En los SSOO GNU/Linux se ejecutan múltiples servicios o demonios que permiten realizar determinadas actividades en el sistema. Cada uno de estos demonios consiste en uno o más procesos que se ejecutan en el equipo.

Además de los procesos vinculados a demonios, en el sistema se ejecutan procesos de usuario. Estos procesos también se pueden gestionar.

5.22.1.- COMANDO PS

El comando *ps* permite ver los procesos que se están ejecutando en el sistema. Para cada proceso se muestra diversa información tal como su identificador (PID), terminal donde se ejecuta (TTY), tiempo de uso de la CPU (TIME) y el comando que ejecuta (CMD).

Parar ver todos los procesos que se ejecutan en el sistema su usa la opción *-A*

$ ps -A

Parar eliminar un proceso que se está ejecutando se puede usar el comando *kill* y el identificador del proceso.

$ kill -9 <PID del proceso>

5.22.2.- COMANDO TOP

El comando *top* es una aplicación en tiempo real que informa sobre la actividad del sistema. Proporciona información sobre la carga del SO, grado de uso de la CPU, memoria, swap y los procesos que están en ejecución.