



Sistemas Informáticos

UNIDAD DIDÁCTICA 3
INTRODUCCIÓN A LOS SISTEMAS OPERATIVOS Y
VIRTUALIZACIÓN

PROFESORA: VANESA ESPÍN
1º DAW BILINGÜE
IES Politécnico HLANZ



Índice de Contenidos

Contenido

1. Funciones y características de un SO	2
2. Tipos y Arquitecturas de los SO	5
2.1. Tipos de Sistemas Operativos.....	5
2.2. Versiones de los SO más usados	8
Sistemas Operativos de Microsoft.....	8
Sistemas operativos GNU/Linux.....	8
Sistemas operativos de Apple	9
2.3. Arquitectura de los Sistemas Operativos.....	9
Sistemas con Capas	10
Sistemas Monolíticos	10
Sistemas Microkernel	11
Sistemas Híbridos	12
3. Arranque de un SO. Gestores de arranque	13
3.1. Arranque Inicial. El POST	13
Secuencia del Post	13
Notificaciones de error POST.....	14
3.2. Elección y Arranque del Sistema Operativo	16
Concepto de partición	17
Estándar MBR.....	19
EFI/UEFI Y GPT.....	21
3.3. Dar formato a discos.	25
3.4. Gestores de arranque	25
Arranque de Windows (BOOT MANAGER).....	26
Editando entradas con software [Presentes en HBCD_PE_x64.ISO].....	27
3.5. Arranque de Windows 8.x / 10 / 11	29
Secure Boot	29
Trusted Boot	30
Fast Startup	30
3.6. Arranque de Linux: GRUB.....	31
3.7. Recuperación de errores en el arranque.....	32
LINUX	35
3.8. Modos de arranque a Prueba de Fallos	35

Introducción

Cuando trabajamos con un computador con sistema operativo, ya sea un móvil, una tableta o un supercomputador, el usuario no se tiene que preocupar de las direcciones de memoria RAM usadas, de la gestión de las interrupciones, de la interfaz gráfica o cómo trabajan internamente los dispositivos de almacenamiento no volátiles.

Los sistemas operativos actuales están compuestos por un conjunto de software muy avanzado que trata de facilitar el empleo del dispositivo al usuario lo máximo posible e intentan menoscabar lo menos posible los recursos hardware.

En este capítulo se abordarán las principales funciones, características y arquitecturas de los sistemas operativos. Además, estos se clasificarán partiendo de conceptos que se han venido desarrollando desde el origen de los sistemas operativos hasta la actualidad, y han determinando su arquitectura. Profundizaremos en los procedimientos de instalación de los sistemas operativos Microsoft Windows y Ubuntu Desktop sobre máquinas virtuales en Oracle VM VirtualBox, estudiando también los procesos de arranque y su actualización.

1. Funciones y características de un SO

Las *funciones básicas* de un sistema operativo son:

1. **Actuar de interfaz** entre el usuario y el hardware de manera transparente para el primero. Debe ofrecer soporte a los usuarios para que sus acciones se transmitan con facilidad. Los usuarios no tienen por qué ser especialistas de software o hardware para usarlo.
2. **Gestionar los recursos** software y hardware del equipo. El uso eficiente de los recursos es primordial puesto que son limitados. Dependiendo del fin y las tareas encomendadas al sistema informático, la eficiencia puede redirigirse a acciones diferentes. Por ejemplo, la eficiencia buscada en un equipo de sobremesa en nuestro hogar es diferente a la eficiencia de un sistema que gestione un conjunto de alarmas en tiempo real.

El sistema operativo es un software con características particulares, ya que debe administrar todos los recursos del sistema entre los usuarios y el resto de software. Por tanto, las *características fundamentales* que debe soportar cualquier sistema operativo genérico son:

- ◆ **Adaptabilidad:** se debe acomodar a dos situaciones que evolucionan en paralelo, nuevo software y nuevo hardware. El sistema operativo debe ser capaz de reacondicionarse (normalmente mediante actualizaciones) para hacer uso de nuevas características o mejoras, tanto en componentes físicos como software.
- ◆ **Facilidad de uso:** teniendo como referente el fin al que se empleará el sistema informático, la facilidad de manejo ha de ser primordial. Normalmente, una mayor comodidad implica mayor gasto de recursos (como por ejemplo un sistema gráfico de ventanas). Por ello, existen sistemas operativos que ganan en eficiencia a costa de restringir su manejabilidad.
- ◆ **Eficiencia:** los recursos (procesadores y núcleos, RAM, acceso a discos, red o cola de impresión) son limitados. El sistema operativo debe atender todas las peticiones de usuarios, programas y el propio sistema operativo para facilitar el acceso a los recursos. Ello debe hacerse barajando la importancia de cada solicitud y de quién desee hacer uso de los recursos. Esta tarea es muy compleja y crítica, ya que repercutirá en todo el sistema.

El sistema operativo debe administrar de forma eficiente los recursos, atendiendo al objetivo de dicho sistema operativo. Los recursos más solicitados son:

1. **Memoria RAM.** La parte del sistema operativo que siempre reside en memoria RAM se denomina **núcleo o kernel**. Es un subconjunto software del propio sistema operativo que por su importancia en la gestión del sistema no puede abandonar la memoria principal. El resto de módulos del sistema operativo se irá cargando y descargando desde los dispositivos de almacenamiento secundario a la memoria principal, dependiendo de la arquitectura del sistema operativo. El espacio restante de memoria RAM se debe gestionar eficientemente para albergar el resto de software y los datos que maneje este.
2. **Procesador.** Aunque disponga de varios núcleos y, por tanto, pueda ejecutar varios procesos a la vez, existe multitud de software que desea ejecutarse.
3. **Adaptadores de red.** Múltiples aplicaciones hacen uso de la red simultáneamente, debiendo administrar las conexiones de red entre aplicaciones, procesos y usuarios.
4. **Medios de almacenamiento.** El acceso a discos duros puede representar un cuello de botella importante.
5. **Colas de impresión.** Pueden existir más de una petición de impresión a una misma impresora, por lo que se debe gestionar la cola de trabajos de impresión adecuadamente.

La **Administración Del Sistema** por parte del sistema operativo se divide en:

- a) **Gestión de procesos.** El procesador, como recurso fundamental del sistema, ha de repartir su tiempo entre los diferentes procesos que deseen ejecutarse. El sistema operativo debe organizar el paso de estos procesos por el procesador (o procesadores) y sus núcleos, de tal manera que los tiempos de ejecución de las diferentes tareas sigan los objetivos del sistema operativo. Por tanto, el sistema operativo debe gestionar:
 - i. La asignación de procesos a varios procesadores (si dispone de varios).
 - ii. El uso de la multiprogramación sobre procesadores individuales y sus núcleos.
 - iii. La ejecución de una aplicación o proceso en cuanto a su sincronización con otros procesos o hilos.

Estos objetivos son definidos por políticas de planificación con orientaciones diferentes:

- *Planificación orientada a los usuarios* (orientada a las entradas y salidas): intenta agilizar las acciones de procesos como accesos a discos, señales de pantallas táctiles o accesos a Internet. Prima el tiempo de respuesta a los usuarios.
 - *Planificación orientada al sistema* (orientada a procesos de cálculo): su objetivo es la eficiencia y el rendimiento de procesamiento. Un ejemplo de ello es lo que ocurre cuando se intenta acaparar el procesador durante mucho tiempo para resolver cálculos aritméticos o lógicos intensos.
- b) **Gestión de Memoria.** Íntimamente ligado a la gestión de procesos se encuentra la de memoria. Por gestión de memoria se entiende la planificación y gestión global de la memoria principal con extensión a la memoria secundaria. Hoy en día los sistemas disponen de memoria RAM suficiente para albergar el sistema operativo y mucho más software. Pero también se debe planificar cómo actuar en caso de necesitar mayor espacio de memoria empleando el almacenamiento permanente. El sistema operativo amplía virtualmente la memoria RAM, tomando prestado del disco duro espacio como si fuese una extensión de la primera (a este concepto se denomina memoria virtual). Toda la transferencia de información

entre memorias requiere una planificación vital para ahorrar tiempo y no lastrar la eficiencia del sistema.

- c) **Gestión de Entradas y Salidas.** Acciones como tocar una pantalla táctil, imprimir un documento, acceder a un fichero del disco duro o navegar por Internet requieren que el sistema operativo necesite administrar dichos recursos, ofreciendo soluciones rápidas y de la forma menos costosa posible. Cada dispositivo de E/S tiene una forma peculiar de interactuar con el sistema operativo, y este ha de gestionarlo estableciendo un diálogo claro y fluido.
- d) **Gestión de Almacenamiento Secundario.** Los discos duros son dispositivos de E/S por sí mismos, pero la gestión de los archivos y directorios como elementos atómicos en ellos es fundamental. La estructura organizativa de los archivos y su gestión viene determinada por los sistemas de archivos.
- e) **Gestión de la Seguridad.** Se debe evitar actuaciones originadas por errores software, errores hardware o por actuaciones maliciosas de usuarios, ya sean intencionadas o no, dando lugar a inconsistencias en el sistema. Por ello, el sistema debe garantizar:
 - i. El servicio y la disponibilidad de sus recursos.
 - ii. La confidencialidad, protección e integridad del sistema y los datos.
 - iii. El control de accesos.
 - iv. La autenticidad en las acciones.
- f) **Gestión de los Errores.** Es un elemento fundamental en todo sistema operativo. El control de la totalidad de las acciones que puedan derivarse del software de terceros, el hardware y el propio sistema operativo es prácticamente imposible. Por ello, el sistema operativo debe gestionar todo tipo de errores de la manera más liviana posible, informando al usuario y salvaguardando de forma prioritaria la seguridad del sistema y los datos.
- g) **Gestión de la Interfaz de Usuario.** Todas las acciones encomendadas al sistema operativo tratadas hasta ahora no tendrían sentido sin una interfaz que permita una clara manejabilidad del sistema. Por tanto, los sistemas operativos con interfaz gráfica o textual deben ofrecer un soporte que permita una fluida comunicación, así como realizar todas las acciones necesarias para la gestión, administración o explotación del mismo.



Figura 1. Administración del Sistema realizada por el Sistema Operativo

2. Tipos y Arquitecturas de los SO

2.1. Tipos de Sistemas Operativos

Los objetivos de los sistemas operativos marcan la eficiencia en el uso al que se destine el sistema. Se pueden diferenciar tipologías de sistemas operativos con objetivos antagónicos entre sí, aunque en la práctica podamos encontrar versiones intermedias muy variadas.

Existen distintos puntos de vista para catalogar los sistemas operativos:

1. Atendiendo al **número de procesos** que se pueden ejecutar concurrentemente:
 - i. *Monotarea o monoprogramado*: un usuario solo puede estar ejecutando un programa, además del propio sistema operativo.
 - ii. *Multitarea o multiprogramado*: un usuario puede ejecutar varios procesos simultáneamente. De esta manera, pueden existir varios programas en memoria susceptibles de ser ejecutados.
2. Atendiendo al **número de usuarios** que pueden ser atendidos por el sistema operativo simultáneamente:
 - i. *Monousuario*: solo pueden atender a un usuario. El usuario goza de todos los recursos, a menos que el sistema operativo los acapare.
Multiusuario: pueden atender a más de un usuario concurrentemente. Por tanto, los recursos del sistema deben ser gestionados para todos ellos.

TOMA NOTA



Los sistemas operativos multiusuario son multitarea, puesto que tratan con diferentes procesos asociados a varios usuarios. Por tanto, un sistema operativo multiusuario y monotarea, puede tratar con varios usuarios simultáneamente, pero con un único proceso por usuario.

Es de reseñar que pueden existir sistemas multiusuario y monotarea, así como multitarea y monousuario.

3. Atendiendo al **tipo de procesamiento**: el SO ha de estar preparado para ejecutar procesos diferentes. Los SO intentan optimizar sus recursos, independientemente de los procesos que atiendan. Sin embargo, los **procesos**, según su forma de ejecutarse, pueden ser:
 - i. *De tiempo real*: requieren unos plazos en su ejecución o tiempos de respuesta.
 - ii. *Interactivos*: requieren de la participación del usuario.
 - iii. *Por lotes, batch o no interactivos*: se suministra un conjunto de tareas al sistema operativo con características similares, y este se encarga de ejecutarlas en serie y sin la intervención del usuario. En caso de producirse un error en una tarea del lote, el resto de tareas no se podrá ejecutar. Ejemplos: realización de facturas agrupadas, tareas de cómputo en investigación, envío de mensajes con informes o resúmenes en cadenas de producción, etc.

De esta manera, existen sistemas operativos más orientados a uno u otro tipo de proceso, puesto que la eficiencia de estos se planifica desde el diseño de los mismos:

- i. *Sistemas operativos en tiempo real*: donde se deben cumplir escrupulosamente los plazos de ejecución de los procesos y, además, deben tener un comportamiento

predecible. Ejemplos: en aviónica, instrumentación médica, sistemas de alertas en una central nuclear, etc.

- ii. *Sistemas operativos interactivos o de tiempo compartido*: orientados a la participación continua del usuario, los cuales hacen uso de los programas antes comentados, tales como un procesador de textos o un editor de imágenes. Son sistemas de propósito general en los que, a diferencia de los sistemas de tiempo real, no priman los tiempos de respuesta en la ejecución de procesos. En esta clasificación se encuentran los más conocidos por nosotros como las versiones de escritorio y de red de Microsoft Windows o de Apple (Mac OS), así como distribuciones Linux, como Ubuntu.

4. Atendiendo al **sistema de interfaz** empleado:

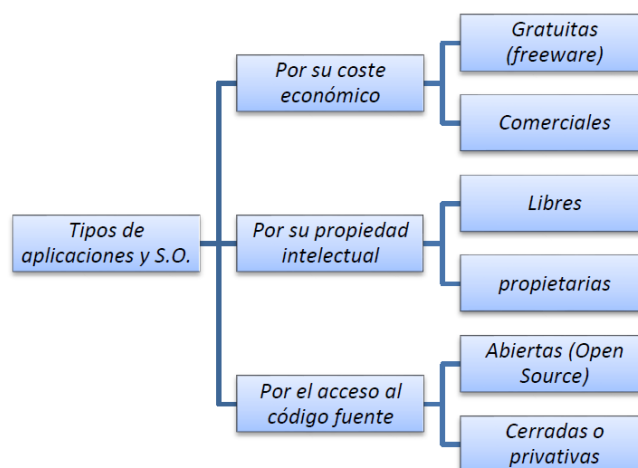
- i. *Textuales*: emplean un repertorio de comandos que se introducen en el sistema de forma escrita a través de un terminal de órdenes. Aunque, se necesitan mayores conocimientos de sintaxis y manejo del sistema operativo, las acciones pueden llegar a ser muy potentes desde un punto de vista de explotación del sistema operativo.
- ii. *Gráficos*: usan un conjunto de ventanas, botones y desplegables gráficos donde se representan los diferentes volúmenes, unidades y sistemas de ficheros de forma muy intuitiva. Además, los programas lanzados presentan una vista gráfica. El manejo se realiza con un dispositivo de entrada/salida, como un ratón, y destaca por su fácil utilización. Este sistema emplea muchos más recursos que el textual, por tanto, en sistemas operativos donde se busca ahorrar todo tipo de recursos en favor de atender a peticiones de usuarios y procesos, la interfaz gráfica se desprecia.

5. Atendiendo a la **forma de ofrecer los servicios**:

- i. *Sistemas operativos cliente o de escritorio*. Se encargan de realizar el procesamiento de la información, la gestión de los procesos, de la memoria, dispositivos de E/S de una sola computadora. Esta computadora suele estar conectada en red, pero el usuario es consciente de sus accesos externos. En un entorno corporativo, se pueden emplear prácticamente para compartir archivos en red. Por tanto, este tipo de sistema operativo es el normalmente empleado en un hogar o pequeña oficina, así como en entornos empresariales en el ámbito de un servicio de directorio en una red distribuida.
- ii. *Sistemas operativos en red*. Se encargan de gestionar la red, los usuarios y los recursos de una red de computadoras en general, de forma centralizada mediante un servidor o varios como réplicas o extensiones del primero. Es en el servidor donde se instala este sistema operativo. El resto de equipos de la red (con sistemas operativos cliente) se conectan al servidor (de forma consciente) formando parte del sistema e interactuando con él. Su principal objetivo es el intercambio de información centralizada. Sin embargo, el servidor puede resultar un cuello de botella si cae o si se deteriora la transferencia de información. Destacan por su seguridad y robustez en la administración general del sistema y la gestión de la información que gestionan frente a los sistemas operativos de escritorio.
- iii. *Sistemas operativos distribuidos*. A diferencia de los anteriores, actúan varios computadores de manera transparente al usuario, de forma que da la sensación que este interactúa solo con uno de ellos. Por tanto, permiten emplear los recursos de varias computadoras en paralelo. Existen pocos ejemplos en la actualidad, destacando *Plan 9* y *Amoeba*. En cualquier caso, se consideran herramientas de estudio e investigación.

6. Atendiendo a su **propiedad intelectual**:

- i. **Libres:** sistemas operativos en los que se ha renunciado a cualquier tipo de *propiedad intelectual*. Pueden usarse libremente, ser distribuidos, permiten que se acceda a su código fuente y permiten que esté sea modificado de la forma que queramos. No hay que confundir el hecho de que sean libres con el hecho de que sean gratuitos (problema de la palabra *free* en inglés, que significa las dos cosas).
- ii. **Propietarios:** la propiedad intelectual es de la empresa que los desarrolla, que no vende en realidad el sistema operativo, sino una licencia de uso del mismo. No se tiene acceso al código fuente del sistema, o por lo menos, no se tiene permiso para modificarlo libremente. También está prohibido distribuir estos sistemas, o usarlos de formas no autorizadas por la empresa desarrolladora. Toda la familia Windows es un claro ejemplo de sistema operativo propietario.
- iii. **Otras clasificaciones:** en general, tanto los sistemas operativos como las aplicaciones normales, pueden definirse según alguno de estos apartados



7. Atendiendo a su **tipo de licencia comercial**

Dentro de los sistemas operativos comerciales, propietarios y privativos, nos podemos encontrar con diversos tipos de licencia de uso:

- i. **OEM (Original Equipment Manufacturer):** licencias que otorga el desarrollador del sistema operativo al fabricante de hardware, tal que cuando compramos uno de sus productos, este viene con una licencia de uso del sistema operativo de tipo OEM. El SO viene preparado para ese hardware específicamente, por lo que no tenemos realmente una licencia de uso del sistema operativo, sino del sistema operativo únicamente para ese hardware en concreto. Son las más económicas, y suelen poseer restricciones especiales, aparte de venir sin manuales ni caja.
- ii. **RETAIL:** licencia que compramos directamente del desarrollador. Somos propietarios de la licencia, podemos instalarlo en cualquier tipo de hardware compatible, podemos revender la licencia o cederla, etc. Normalmente solo permiten su uso en una sola máquina a la vez. Vienen con su caja y manuales. En las licencias de tipo Retail, normalmente podemos elegir entre una licencia completa, o una licencia de actualización, que permite actualizar un sistema anterior al nuevo, por un coste algo más reducido.

- iii. **VLM (LICENCIAS POR VOLUMEN)**: para una empresa con cientos de ordenadores, es complicado controlar las licencias individuales de cada una de sus máquinas. Existe la posibilidad de contratar un tipo de licencia especial con el desarrollador, de modo que con una única clave de licencia, podemos utilizar varias máquinas a la vez. Son las licencias más caras evidentemente, aunque son bastante más económicas que comprar cada una de las licencias individualmente.
- iv. **MSDN (LICENCIAS DE EDUCACIÓN)**: son licencias especiales de Microsoft que permiten su uso únicamente para actividades educativas y de formación. Cualquier uso de estas licencias en equipos que desarrollen actividades fuera de este ámbito, es ilegal. Existen también para empresas de desarrollo, academias, etc.

2.2. Versiones de los SO más usados

Los sistemas operativos comerciales más utilizados disponen de versiones o distribuciones para las siguientes plataformas, principalmente: equipos de escritorio, servidores y dispositivos móviles.

Sistemas Operativos de Microsoft

En el caso de los sistemas operativos de la compañía Microsoft, nos encontramos con multitud de versiones cuyo mercado se centra principalmente en las siguientes plataformas

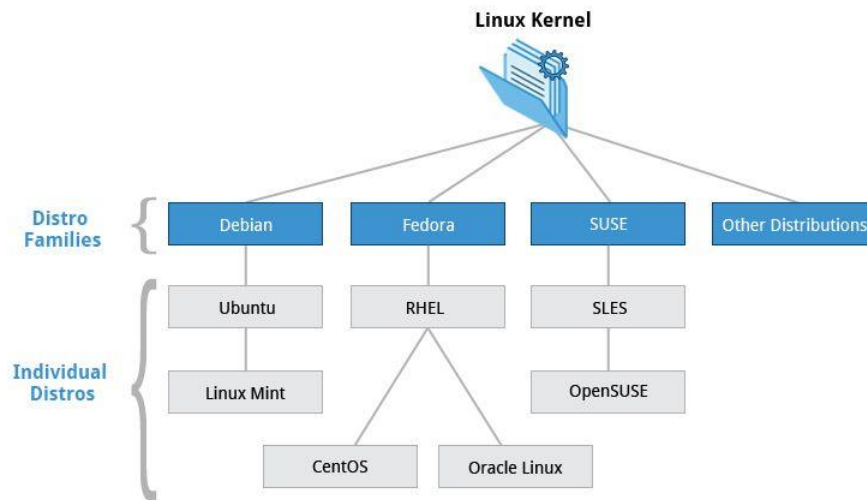
1. **Para equipos de escritorio: Microsoft Windows 10 / 11.** Incluye multitud de ediciones orientadas a diferentes ámbitos que, según las orientaciones, dispone de mayores prestaciones. Destacamos:
 - ◆ Home: para equipos de sobremesa, tabletas o portátiles de poca potencia, para un uso básico (multimedia y conectividad).
 - ◆ Pro: orientado a fines de negocio para empresas o profesionales.
 - ◆ Enterprise: también orientado a fines de negocio, pero de mayor volumen
 - ◆ IoT: ideada para dispositivos relacionados con el Internet de las Cosas
 - ◆ Education: dirigido a un entorno académico.
 - ◆ Pro for Workstations: para equipos muy potentes con grandes cargas de trabajo intensivo o tareas críticas.
6. **Para equipos de tipo servidor: Microsoft Windows Server 2019.** Entre las que destacamos las siguientes ediciones:
 - ◆ Datacenter: para entornos en la nube o centros de datos altamente virtualizados
 - ◆ Standard: para ambientes poco virtualizados.
 - ◆ Essentials: para pequeños negocios con un número limitado de usuarios y dispositivos.

Sistemas operativos GNU/Linux

Las distribuciones de sistemas operativos GNU/Linux son muy variadas, existiendo multitud de versiones en cada una de ellas. En el caso de distribuciones para equipos de **sobremesa** o portátiles, son enormes las variedades según el uso del equipo (genérico, seguridad, juegos, ligereza, orientado a la nube). Algunas de las distribuciones más empleadas son:

- ◆ Ubuntu Desktop y Mint: muy genéricos y versátiles, de gran facilidad de uso.
- ◆ Arch Linux: distribución personalizable para usuarios avanzados.

- ◆ Kali Linux y Tails: orientados a la seguridad y la privacidad.
- ◆ Chromium OS: versión liberada de Chrome OS (sistema operativo en la nube de Google) con licencia BSD.
- ◆ Android: opción archiconocida para smartphones.



Principales familias Linux

Algunas de las distribuciones más empleadas para **servidores** son:

- ◆ Red Hat Enterprise Linux
- ◆ Ubuntu Server
- ◆ CentOS
- ◆ SUSE Linux Enterprise Server
- ◆ FreeBSD

Sistemas operativos de Apple

Por otro lado, la empresa Apple Inc. desarrolla sistemas operativos para portátiles, equipos de sobremesa, servidores, móviles y otros dispositivos conectados para hardware específico. Sus versiones más utilizadas son:

- ◆ macOS: sistema operativo de escritorio y equipos portátiles. Versiones más actuales: Monterey (octubre de 2021) y Ventura (lanzada en octubre del 2022).
- ◆ iOS para sus smartphones.

2.3. Arquitectura de los Sistemas Operativos

La arquitectura de los sistemas operativos ha ido evolucionando de la mano del desarrollo hardware de los sistemas informáticos. Ambas partes no pueden funcionar de forma aislada y dependen la una de la otra.

A lo largo de los años se han sucedido varias tipologías de arquitecturas en el desarrollo de los sistemas operativos, cada una con sus ventajas e inconvenientes y estando orientadas a propósitos diferentes. Si bien es cierto que la evolución de los propios sistemas operativos ha tomado ideas de

arquitecturas o modelos anteriores para fusionarlos y hacerlos propios en beneficio de nuevos sistemas operativos.

Los componentes principales de la estructura de un Sistema Operativo son:

- Núcleo o kernel:** capa que interactúa directamente con el hardware y está formada por los componentes esenciales del sistema operativo debido a su relevancia y frecuencia de uso. Se encuentra cargado permanentemente en memoria principal. Una parte del núcleo se encarga de abstraer la parte hardware del sistema para que el sistema operativo trabaje independientemente de la máquina donde sea instalada. A esta parte se le llama HAL (Hardware Abstraction Layer).
- Servicios:** formada por un conjunto de funciones básicas que dan soporte a las aplicaciones de usuario para que interactúen con el núcleo. En esta capa se incluye de manera más o menos diferenciada las siguientes funciones:– Gestión de procesos.– Gestión de memoria.– Gestión de la E/S.– Gestión de almacenamiento secundario.
- Interfaz:** constituida principalmente por un intérprete de órdenes cuya función es traducir y trasladar las acciones deseadas por un usuario niveles inferiores. En este mismo nivel, aunque de manera diferenciada, se pueden catalogar los “Programas de usuario”, es decir, cualquier aplicación o software que instalamos en nuestro equipo y que nos permite realizar tareas concretas.

La **diferentes arquitecturas** estructuran dichos componentes de diferente modo. Las principales arquitecturas son:

Sistemas con Capas

Presentan una estructura interna llamada jerárquica, en niveles o en capas. Se puede decir que están formados por un conjunto de capas que representan servicios o funciones diferentes. Cada capa solo se puede comunicar con la capa inmediata inferior o superior para solicitar servicios o resolver peticiones, respectivamente. Su principal ventaja es el uso de una estructura bien definida que facilita la corrección de errores, pero resulta lento y complejo al definir las capas. Ejemplo de ello son los sistemas operativos THE y MULTICS, ambos en desuso. Estructura típica de un sistema de capas:

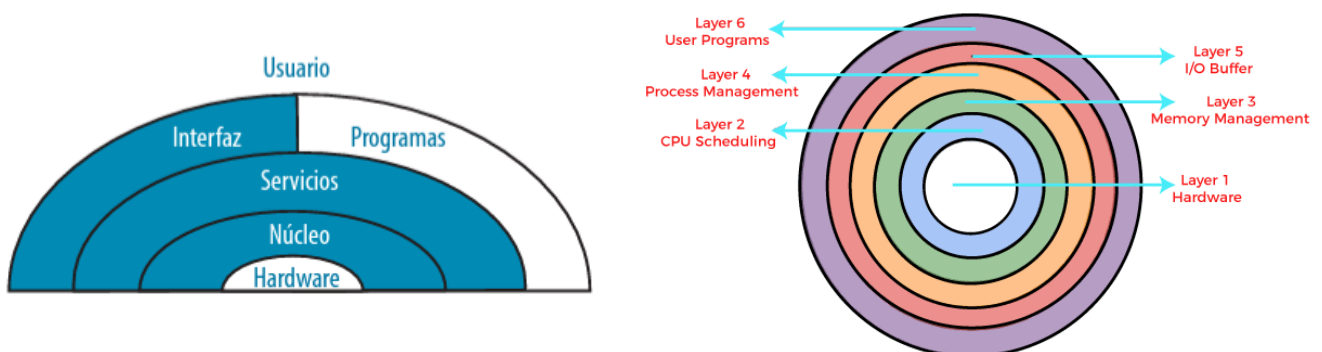


Figura 2. Estructura genérica de un SO en capas

Sistemas Monolíticos

Su nombre procede de los sistemas que tenían una única estructura, es decir, un gran programa dividido en rutinas (subprogramas), en la que todas ellas tenían los mismos privilegios (ejecutándose en modo *supervisor*, es decir, con altos privilegios) y se podían llamar unas a otras. Se ejecutaba en un espacio de direcciones de memoria principal único y compartido por las diferentes rutinas. Por ello,

es sencillo su diseño y, sobre todo, destacan por su rendimiento o velocidad. Como se ve en la Figura 3, en realidad, hay una estructura muy simple donde, el programa principal invoca al procedimiento de servicio que necesite, el cual realiza llamadas a los procedimientos auxiliares o utilitarios, según sean necesarios.

Ejemplos de ello fueron los sistemas operativos DOS y las primeras versiones de UNIX. A día de hoy, los sistemas operativos basados en sistemas monolíticos han mejorado, dejando atrás sus mayores inconvenientes: difícil evolución y resolución de errores y baja estabilidad. Ejemplo: Ubuntu.

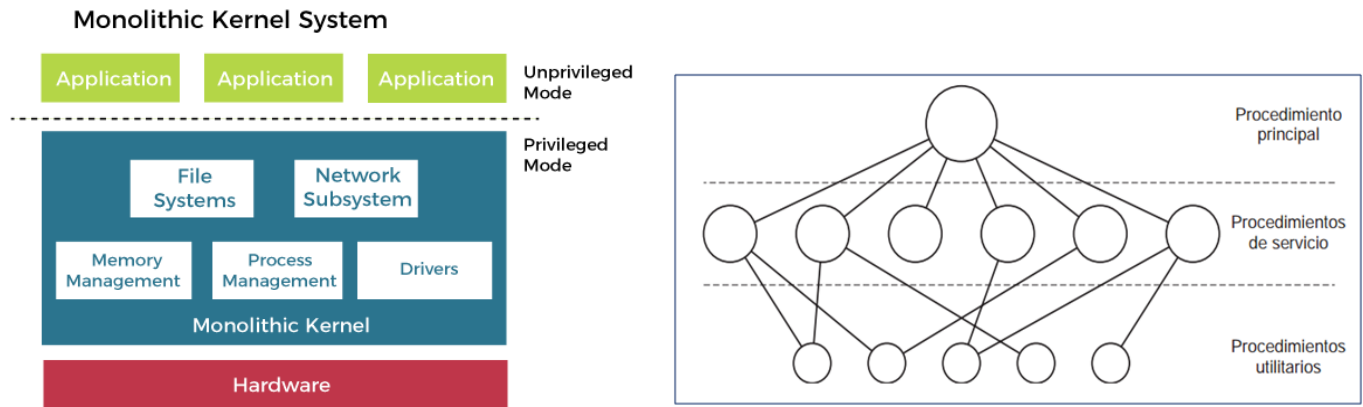


Figura 3. Estructura de un SO Monolítico

Sistemas Microkernel

Su principal propósito es el de liberar al núcleo del máximo de su funcionalidad. Se pretende restringir el uso del modo supervisor (o modo núcleo) y facilitar la evolución y el mantenimiento del sistema operativo. De esta manera, el kernel se encargaría básicamente de:

- ◆ La gestión de la memoria.
- ◆ Gestiones prioritarias de procesos e hilos.
- ◆ Control básico de la comunicación entre el resto de procesos o servicios.

El resto de servicios quedarían fuera del núcleo, así, se ejecutarían ahora en modo usuario por ejemplo, la gestión de archivos, los protocolos de comunicaciones o los drivers de dispositivos.

La idea es que un proceso *cliente*, como, una aplicación de usuario, desea obtener servicio de un proceso *servidor*. Para ello, el cliente envía un mensaje al servidor a través del microkernel, y este se encarga de la comunicación y gestión necesaria para que todos los clientes sean atendidos con eficiencia por los diferentes servidores. Así, tanto clientes como servidores se ejecutan en modo usuario, y una pequeña parte de todo el proceso (la más crítica), en modo núcleo. Con esto se mejora:

- La seguridad del sistema operativo, al ejecutarse la mayoría de los procesos en modo usuario.
- La estabilidad, debido a la modularidad.
- La actualización del sistema operativo.

Sin embargo, uno de los principales defectos de esta arquitectura es la posible sobrecarga en la gestión de procesos que ocasiona un deterioro en el rendimiento del sistema.

Un ejemplo de sistema operativo microkernel es MINIX (clon de Unix, desarrollado por Tanenbaum en 1987 para enseñarlo en clase cuando Bell prohibió el uso de Unix para la enseñanza).

Sistemas Híbridos

Se considera una evolución que aúna las arquitecturas monolítica y microkernel, persiguiendo las ventajas de ambas.

Consiste en un diseño microkernel, pero con una implementación monolítica, que consigue una gran estabilidad y un significativo rendimiento (como ventajas de ambos modelos, respectivamente). A diferencia de los sistemas microkernel, los sistemas híbridos añadirían en su espacio kernel los drivers de dispositivos y todo lo relativo a la comunicación entre procesos, como servicios fundamentales para ejecutar en modo supervisor. Windows de la familia NT y MacOs se consideran sistemas híbridos.

La siguiente figura muestra una comparativa entre las diferentes arquitecturas:

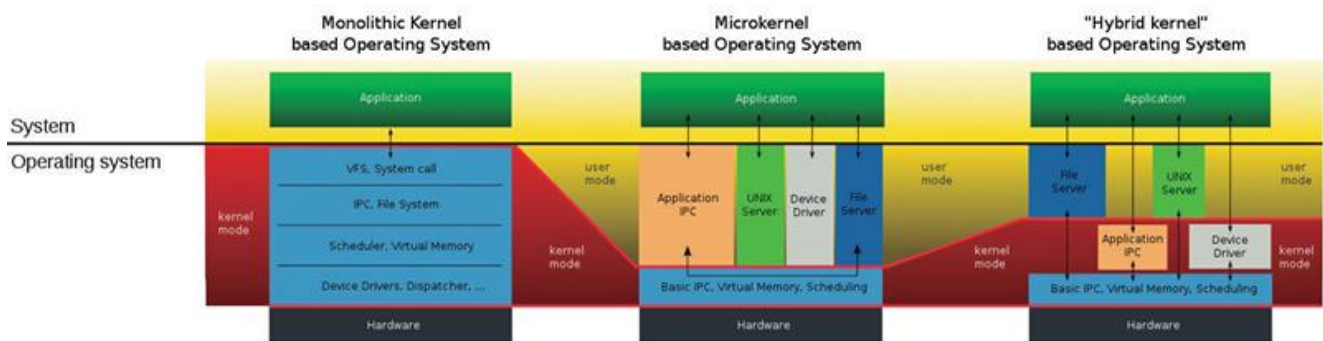


Figura 4. Comparativa entre arquitecturas del SO

IPC=Inter-Process Communications

VFS=Virtual File System

Recordatorio sobre los modos de ejecución:

- ◆ Modo supervisor o kernel: modo privilegiado, para ejecutar las instrucciones privilegiadas (aquellas que afectan directamente a los recursos de la máquina, asociadas a la protección del hardware, gestión de procesos y memoria, etc).
- ◆ Modo usuario: no tiene acceso a las instrucciones privilegiadas.

3. Arranque de un SO. Gestores de arranque

Ya hemos visto anteriormente que el **hardware**, por si solo es totalmente incapaz de realizar ninguna acción, necesita un software inicial que le indique que tiene que hacer. Cuando encendemos un sistema informático, estamos poniendo en marcha hardware, por lo que se necesitan medios especiales para hacer que se cargue un primer **software**.

3.1. Arranque Inicial. El POST

La BIOS (Basic Input/Output System) es una memoria especial alojada en la placa base que contiene una aplicación que se inicia cuando se enciende o resetea un equipo. Esa aplicación se llama BIOS y ha acabado dando nombre a la memoria que la contiene.

Las funciones que tiene la BIOS (como aplicación) son:

- Primero: chequear el hardware del sistema (POST).
- Segundo: buscar la unidad que cargará el sistema operativo (BOOT).

Estas dos acciones se ejecutan estrictamente por este orden, de forma que mientras exista un fallo en el test del sistema no se podrá ejecutar la carga del sistema operativo.

El **POST (Power-On-Self-Test)**, como hemos visto, es una parte de la aplicación BIOS que se encarga de verificar el hardware conectado. La variedad de modelos y versiones de BIOS hace que haya una gran gama de POST.

Todos comprueban las partes principales del equipo, aunque no siempre en el mismo orden, y se distinguen por las comprobaciones de determinadas partes

Secuencia del Post

La secuencia del POST para un mismo modelo de BIOS puede variar o no de una versión a otra de la aplicación si se han incorporado funcionalidades nuevas. Estas incorporaciones surgen de las prestaciones de las placas base para las que suelen fabricarse estos modelos de BIOS.

Secuencia del POST en una BIOS AWARD (AWARDBIOS)	
↓	Test general del estado del microprocesador y del reloj del sistema.
	Test al controlador del teclado para verificar que está disponible.
	Arranca la actividad del chipset para llevar a cabo los test de los dispositivos del equipo.
	Se comprueba y activa el controlador de la tarjeta gráfica para mostrar posteriormente la información por la pantalla.
	Test de estado del chip CMOS (BIOS).
	Test de la memoria DMA (de acceso directo).
	Test a la parte baja de la memoria RAM (los primeros 64 KB).
	Test a la memoria de la tarjeta gráfica.
	Test a la pila de la placa base.
	Test de capacidad de memoria del sistema.
	Test del estado de la memoria RAM.
	Test de la memoria extendida.
	Se inicializa la disquetera (si existe).
	Se detectan los puertos serie y paralelo.
↓	Se inicializan los discos duros (si existen).

En cualquier caso, el POST se adapta a las prestaciones de la placa en la que está funcionando y no realizará un test a un componente que no esté contenido en la placa.

En las tablas que mostramos a la izquierda, se puede observar la secuencia de comprobaciones típica del POST para los BIOS Phoenix Award y AMI (American Megatrends).

Secuencia del POST en una BIOS AMI (AMIBIOS)	
↓	Test al controlador del teclado para verificar que está disponible.
	Arranca la actividad del chipset para llevar a cabo los test de los dispositivos del equipo.
	Test de estado del chip CMOS (BIOS).
	Test de la memoria DMA (de acceso directo).
	Se comprueba y activa el controlador de la tarjeta gráfica para mostrar posteriormente la información por la pantalla.
	Test a la parte baja de la memoria RAM (los primeros 64 KB).
	Test a la memoria de la tarjeta gráfica.
	Se inicializa la disquetera (si existe).
	Se detectan los puertos serie y paralelo.
	Se inicializan los discos duros (si existen).

En ambos casos, una vez han finalizado las comprobaciones, si no ha fallado ningún test, pasa el testigo al BOOT para que cargue el sistema operativo en el equipo.

Notificaciones de error POST

En el caso de que se produzca algún fallo, el POST lo notifica mediante una secuencia de pitidos o a través de un mensaje de error en la pantalla. Según el modelo de BIOS instalado, la forma de comunicar el error varía. Suele venir en forma de pitidos, mensajes o códigos (recibidos en tarjetas de diagnostico POST).

AWARD	Pitidos cortos	Pitidos largos	Descripción del error	Componente implicado
	2	1	Error en la conexión del monitor	Monitor/Tarjeta gráfica
	1 continuo	0	Error en la memoria RAM	Memoria RAM
	0	1 continuo	Calentamiento excesivo del microprocesador	Microprocesador
AMI	3	1	Error en la tarjeta gráfica	Tarjeta gráfica
	Pitidos cortos	Pitidos largos	Descripción del error	Componente implicado
	1	0	Error en el refresco de la memoria RAM	Memoria RAM
	2	0	Error en la paridad de la memoria RAM	Memoria RAM
	3	0	Error en los primeros 64 KB de la RAM	Memoria RAM
	4	0	Error en el reloj del sistema	Placa base
	5	0	Error en el microprocesador	Microprocesador
	6	0	Error en el puerto del teclado	Placa base
	7	0	Error de excepción en el microprocesador	Microprocesador
	8	0	Error en la memoria de vídeo	Tarjeta gráfica
	9	0	Error en la BIOS	BIOS
	10	0	Error en el acceso a la CMOS	Placa base
	11	0	Error en la memoria cache	Memoria cache
	1	2	Error en la tarjeta gráfica	Tarjeta gráfica
	1	3	Error por encima de los 64 KB de la RAM	Memoria RAM
	1	8	Error en la comprobación de la tarjeta gráfica	Tarjeta gráfica

Además de los pitidos, los BIOS notifican los errores por mensajes en pantalla. Aunque cada BIOS tiene sus propios mensajes, al final prácticamente todas tienen los mismos. Ver la tabla siguiente: Mensajes del Post para conocer los mensajes más típicos.

Mensajes de error	Error en pantalla	Descripción del error
	BIOS ROM checksum error - System halted	BIOS corrupta
	CMOS battery failed	Fallo en la pila de la placa (agotada)
	CMOS battery state low	Pila a punto de agotarse
	CMOS checksum error - Defaults loaded	Error al cargar configuración de la BIOS
	Floppy disk(s) fail	Fallo en la disquetera
	Hard disk install failure	Fallo en la conexión del disco duro
	Hard disk(s) diagnosis fail	Fallo en el diagnóstico del disco duro
	Keyboard error or no keyboard present	Error en el teclado o teclado no conectado
	Memory test fail	Fallo en el test de memoria
	Parity error	Fallo en la paridad de la memoria RAM
	Primary/Secondary master hard disk fail	Fallo en unidad de disco maestra de IDE1/2
	Primary/Secondary slave hard disk fail	Fallo en unidad de disco esclava de IDE1/2
	Insert bootable media	No se encuentra una unidad para arrancar
	Primary boot device not found	Unidad de arranque inicial no encontrada
	System halted	Sistema interrumpido por fallo desconocido

Dependiendo del tipo de error, el POST nos permitirá seguir adelante con la carga del sistema o nos obligará a solucionar el fallo antes de continuar.

A continuación se muestra un tarjeta de diagnóstico POST:



La última instrucción del programa POST se encarga de buscar otro programa (pasarle el testigo) que pueda ser cargado en el procesador del PC para que se encargue de seguir arrancando el sistema informático, normalmente cargando ya un sistema operativo.

¿Pero dónde buscará el POST el programa a cargar? Y en caso de que existan varios sistemas operativos en varios soportes, ¿cuál de ellos será el elegido? Esto es lo que vamos a ver a continuación.

3.2. Elección y Arranque del Sistema Operativo

Por ello, la última misión del POST es buscar otro programa, y cargarlo en la CPU antes de liberarla. En un sistema informático actual podemos tener:

- múltiples discos duros, cada uno de ellos con varias particiones (este concepto lo veremos más adelante) donde pueden estar almacenados varios sistemas operativos,
- un CD/DVD en la unidad lectora que también cuente con su propio sistema operativo,
- un pequeño sistema operativo en un dispositivo USB,
- un cable de red que arranque un sistema por PXE, etc.

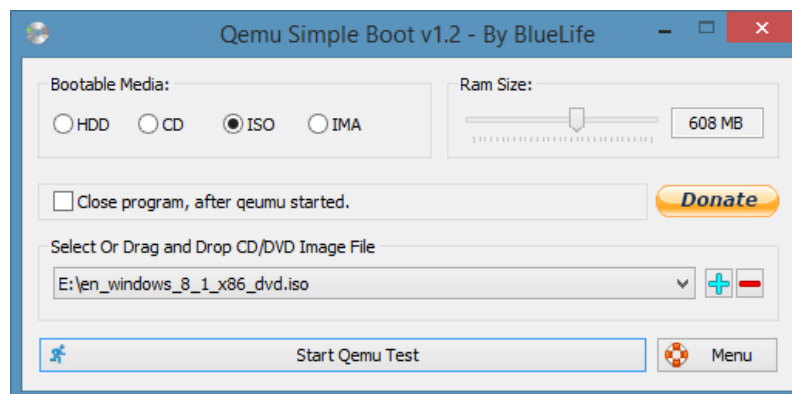
1. ¿Cómo puede saber el POST a cuál de todos estos programas cederle el control?

La respuesta está en la BIOS

Normalmente estas opciones se encuentran en la segunda opción que aparece en el menú de la BIOS (opciones avanzadas de la BIOS o Advanced BIOS Features).

En alguna opción de este menú, normalmente se nos permite indicar varios dispositivos ordenados que utilizaremos para el arranque. Desde la BIOS vemos cómo podemos indicar de qué dispositivo queremos arrancar. Podemos indicar normalmente si queremos arrancar desde el disco duro, desde el CD, USB, por la red vía PXE, etc.

Existe un programa muy útil llamado [QEmu Simple Boot](#) que sirve para probar si un dispositivo es arrancable sin necesidad de probarlo con software de virtualización: una ISO, una partición / Pendrive (HDD), etc...



2. Una vez que sabemos el dispositivo, ¿De qué forma realiza el arranque del sistema?

La respuesta depende del tipo de dispositivo de almacenamiento

- Si el sistema operativo se ejecuta **desde CD/DVD**, no hay demasiados problemas, dado que en un CD/DVD solo puede haber un único proceso de arranque para un único sistema operativo.
- Si el sistema operativo se ejecuta **desde USB** (o desde un disco duro externo), el gestor de arranque instalado en él será el encargado de decidir la partición desde la cual será cargado el sistema operativo (para hacer arrancable un dispositivo de almacenamiento de este tipo normalmente hay que formatearlo, y ejecutar un programa que lo haga booteable, es decir, que ponga activa la partición). Por ejemplo: *GRUB4DOS* es una variante de GRUB que se instala en

los USB que es capaz de arrancar cualquier tipo de arranque. Usa el programa *grldr* y una lista como *grub (menu.lst)*. Por tanto podré arrancar distintos SSOO desde USB.

- Si el sistema operativo se ejecuta **desde DISCO DURO**, es posible que en el disco duro tengamos varios sistemas operativos para arrancar en nuestra maquina en varias particiones. Además, podemos tener varios discos duros en nuestro sistema, y en cada disco podemos tener varios sistemas operativos instalados. Hay BIOS desde donde se puede indicar incluso desde cuál de los discos duros queremos arrancar (HDD-0, HDD-1, etc.). Si no existe esta opción, tendremos que desactivar los discos duros de los que no queremos que arranque. Con esto conseguimos indicar al sistema informático qué disco duro quiero utilizar para el arranque del sistema... pero resulta que en un solo disco duro puedo tener instalado más de un sistema operativo.

¿Cómo se le indica al sistema que quiero arrancar con Windows 10, o con Windows 7, o con Ubuntu si todos estos SO están instalados en el mismo disco duro? Para entender esto tenemos que comprender bien como está organizado un disco duro.

Concepto de partición

Los discos duros se deben particionar para organizar la información convenientemente, poder almacenar datos, programas y almacenar sistemas operativos. Cuando adquirimos un disco duro y lo instalamos en nuestro ordenador, normalmente se encuentra particionado. Además podemos realizar particiones libremente, con programas específicos del sistema operativo o externos a él.

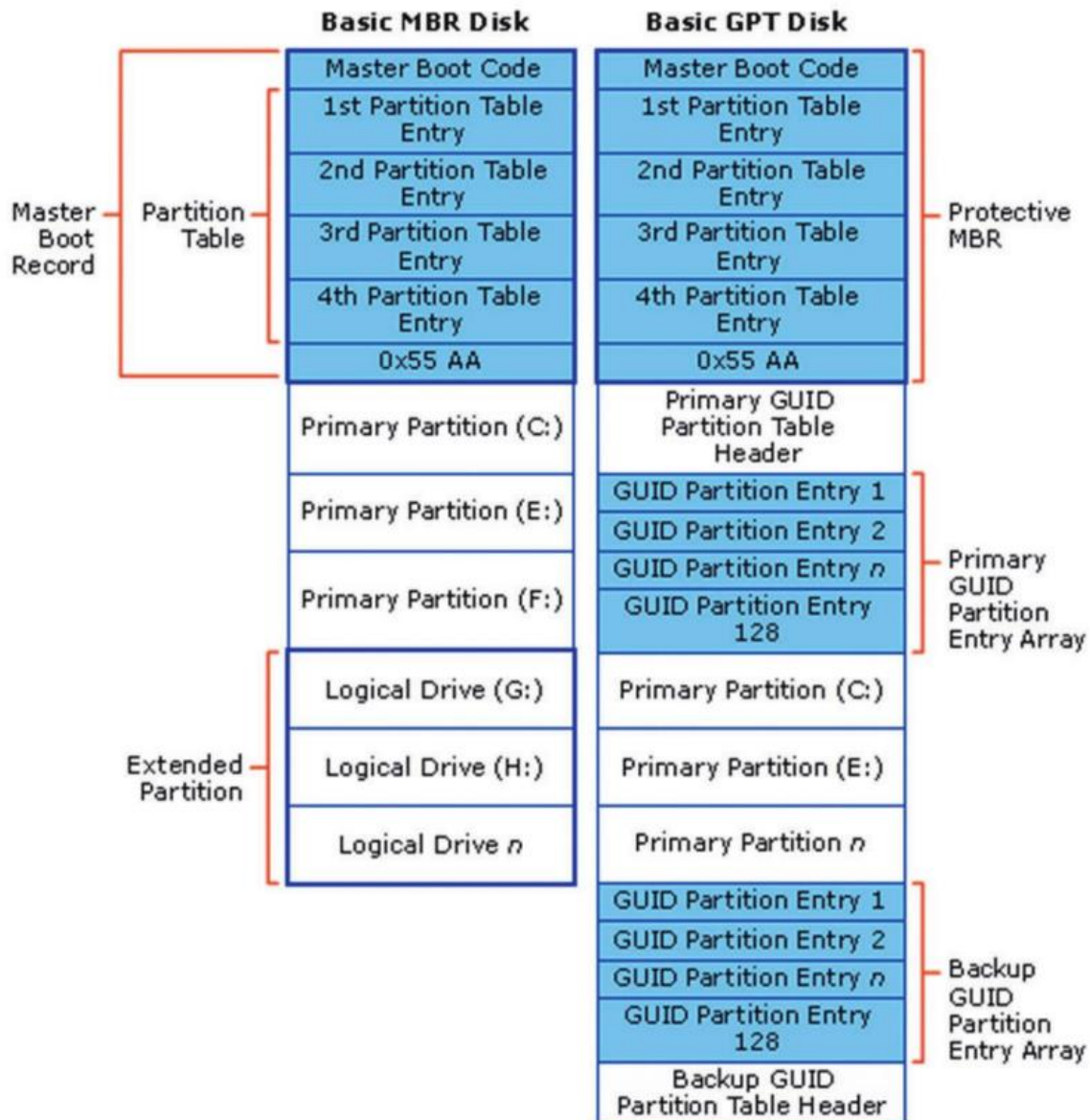
Para almacenar información en una partición, esta debe tener un sistema de archivos (sólo uno). Los sistemas de archivos confieren características al tratamiento de los datos contenidos en ellos. El proceso de asignar un sistema de archivos, por ejemplo: FAT o ext4, es conocido como **formatear**.

Por tanto, **llamamos partición a una división del espacio de almacenamiento de forma contigua en un disco duro**. Las particiones se usan para:

- ◆ Organizar la información: podemos estructurar la info contenida en las particiones de manera coherente.
- ◆ Eficiencia: en los discos mecánicos, el rendimiento de la cabeza lectora mejora al tener un recorrido inferior al total.
- ◆ Instalación de sistemas operativos: ya que deben ocupar una partición con sistema de archivos compatible.
- ◆ Seguridad: al ofrecer espacios físicos distintos, los problemas que afecten a una partición (malware, errores físicos o del sistema...) no tienen por qué afectar al resto de particiones.

Los discos duros de nuestro equipo pueden estructurar sus particiones atendiendo a dos estándares:

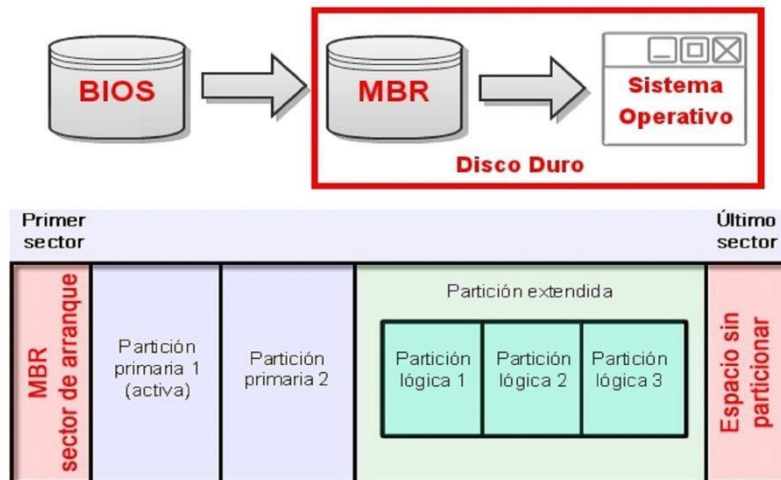
- a) **MBR (Master Boot Record)** → usado en los sistemas con estándar BIOS. Es más antiguo, pero se sigue empleando por su compatibilidad con los SO's.
- b) **GPT (GUID Partition Table)** → usado en los sistemas con estándar EFI / UEFI. A cada partición se asigna un identificador global GUID. Es más flexible, potente y fácil de usar.



Estándar MBR

Veamos como organiza el sistema operativo el disco duro en los sistemas con BIOS tradicional:

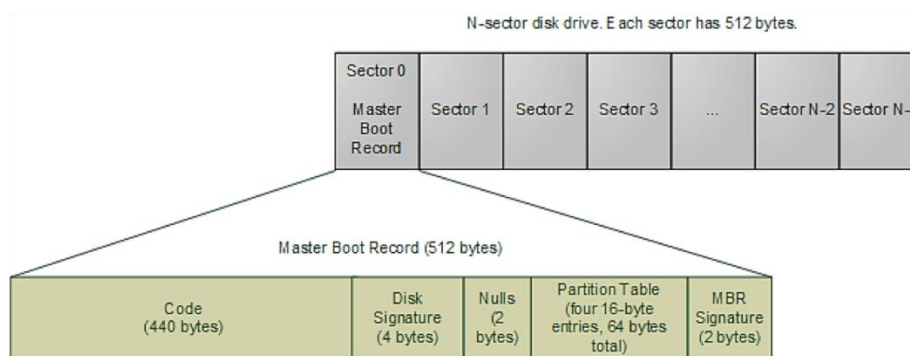
- 1) Tienen una **tabla de particiones** en el primer sector (MBR).
- 2) Pueden crearse **de una a cuatro particiones**.
- 3) Cada partición (*primaria*) tiene su propio **sector de arranque** (permite arrancar un sistema).



Las **particiones** son divisiones lógicas efectuadas en un disco duro. Cada una de ellas podría albergar un Sistema Operativo. Una partición tipo *DOS* contiene sector de arranque, FAT, directorio raíz y área de datos, una partición NTFS tiene su sector de arranque y MFT (Master File Table), etc. Los datos de una partición no se mezclan con los de otra. Ya veremos los tipos de sistemas de archivos en un tema posterior pero de forma general los que vamos a utilizar son:

- ◆ FAT16 y FAT32: el primero con limitación de 2GB (si usa *clusters* de 32k) o 4GB (si usa *clusters* de 64k)
- ◆ NTFS: en todas sus versiones
- ◆ EXT2, EXT3, EXT4: Sistema de ficheros nativo de Linux

En un disco duro con formato MBR podemos **tener hasta 4 particiones como máximo**. De las 4, solo una puede estar definida como **activa (bootable)** al mismo tiempo. Esta partición activa será la que cargue el sistema operativo. En el primer sector (sector 0) de todo disco duro NO se sitúa un sector de arranque (ya que se sitúan en las particiones, por lo que es posible que en un disco duro MBR existan 4 sectores de arranque), en su lugar se sitúa el Master Boot Record o **MBR** (que podría incluso llegar hasta el sector 63).

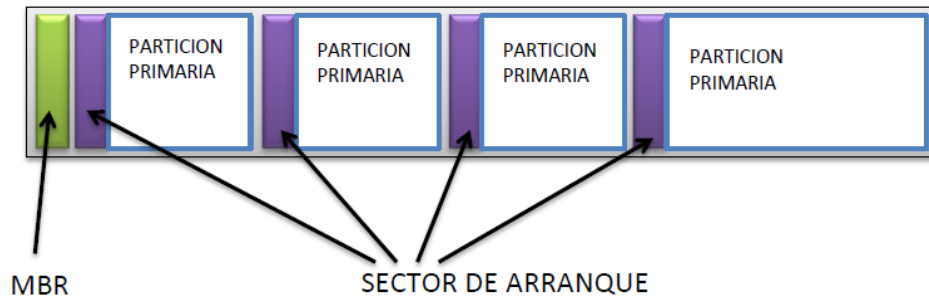


El MBR incluye una tabla (Tabla de Particiones) donde definimos las 4 particiones que pueden estar presentes en nuestro disco duro y un pequeño programa que permite localizar la partición activa, leer su sector de arranque y usarlo para arrancar nuestro sistema informático.

Un programa MBR estándar, leerá la tabla de particiones y escogerá de cuál de esas particiones va a arrancar el sistema operativo. No lo hará como podría parecer lógico de la primera partición, sino de la **partición primaria que está marcada como activa**. El MBR lee el primer sector de esa partición, y le cede el control de la CPU a ese programa (Boot Sector o **SECTOR DE ARRANQUE**).

Las particiones de un disco duro pueden ser de dos tipos: **PRIMARIAS** y **EXTENDIDAS**

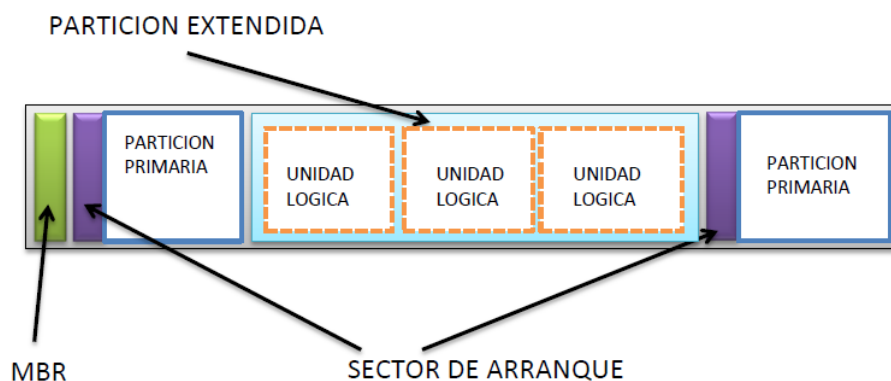
En un disco duro MBR puede haber 4 particiones como máximo, lo que implica que puede haber 4 particiones primarias como máximo. Sin embargo, no puede haber más de 1 partición extendida en un disco duro. Cada **PARTICIÓN PRIMARIA** forma un **volumen** (una letra de unidad, para entendernos) y tiene su **propio sector de arranque**.



Una **PARTICIÓN EXTENDIDA** sin embargo, no forma ningún volumen, ni tiene un sector de arranque como tal. Una partición extendida en realidad es un contenedor de unidades lógicas.

Cada **unidad lógica** que se crea dentro de una unidad extendida sí forma su propio volumen, aunque no tiene un sector de arranque real, sino que usa su sector de arranque para controlar su tamaño entre otras cosas.

De esta manera, si dividimos un disco duro en dos particiones primarias (dos volúmenes) y una partición extendida (donde creamos 3 unidades lógicas, cada una con su propio volumen) formaremos un total de 5 volúmenes (5 letras de unidad) pero solo tendremos dos sectores de arranque usables como tales, los de la partición primaria.



En principio, solo el sector de arranque de una partición primaria es válido para arrancar el sistema operativo. El sector de arranque de la partición extendida solo contiene información sobre las unidades lógicas que se encuentran dentro de ella, y los sectores de arranque de las unidades lógicas contienen información específica a cada unidad lógica.

La tabla del MBR identifica la localización y tamaño de la partición extendida, pero no contiene información sobre las unidades lógicas creadas dentro de esta partición extendida. Ninguna de estas unidades lógicas pueden ser marcadas como activas, por lo que es posible que instalemos un sistema operativo en alguna de estas particiones lógicas, pero nunca podrá ser cargado directamente, ya que

no podemos marcar esa partición como activa, y por lo tanto no podemos indicar que sea el disco de arranque. Sí que podemos cargar estos sistemas operativos instalados dentro de una unidad lógica, pero usando un programa especial que haga las funciones del sector de arranque del que no disponen.

El "**truco**" está en instalar un programa especial (si el programa es pequeño como en el caso del GRUB se instala directamente en el lugar donde está el MBR) o se instala en el sector de arranque de la partición activa en el caso de los Windows. Este programa, puede engañar a la máquina, buscando información sobre las particiones lógicas, y luego cargando el Boot sector deseado en lugar del que debería leerse. Estos programas, que permiten "hacer trampas" en el momento del arranque, suele ser conocidos como **GESTORES DE ARRANQUE** (Boot Manager) y suelen venir incluidos junto con los sistemas operativos actuales.

Estos gestores permiten indicar en el momento del arranque, de cual volumen vamos a cargar el **SECTOR DE ARRANQUE**, sin importarles si dicho volumen es una partición primaria o una unidad lógica.

La familia Windows NT (XP, Vista, Siete, 2000, 2003, 2008, 2010, Server 2012,...) cuenta con su propio gestor de arranque **NTLDR** o **BOOTMGR** que se instala automáticamente al instalar uno de estos sistemas operativos, pero solo activa su menú si detecta que en el disco duro existe más de un sistema. Asigna las letras de unidad según particiones primarias (aunque sean en discos distintos), luego asigna letras a unidades lógicas.

Por su parte, los sistemas basados en Linux utilizaban un gestor de arranque conocido como **LILLO** (Linux Loader) aunque cada vez más sistemas Linux han cambiado este gestor por otro mucho más potente **GRUB** (Grand Unified Bootloader). Además, en estos sistemas las particiones primarias reservan los números del 1 al 4, empezando siempre las lógicas a partir del 5.

Todos estos gestores de arranque funcionan en modo texto normalmente. Nos presentan una lista con todos los sistemas operativos instalados en nuestros discos duro, y escogemos aquel con que deseemos cargar. Hay gestores que trabajan de forma gráfica, pero debido a su mayor tamaño no son especialmente recomendables.

EFI/UEFI Y GPT

EFI/UEFI

Hemos visto en el punto anterior cómo se utiliza la BIOS. Sin embargo, desde hace un tiempo se está sustituyendo nuestras antiguas BIOS por un sistema más moderno conocido como **EFI** (Extensive Firmware Interface), también conocida como **UEFI**, la UEFI es la fundación (más de 140 empresa incluidas Microsoft) que promueve el uso de esta nueva "BIOS". EFI/UEFI permite una BIOS gráfica donde se puede utilizar el ratón y da mucha más información que sus predecesoras. Es incompatible con BIOS (no pueden estar funcionando las dos al mismo tiempo).



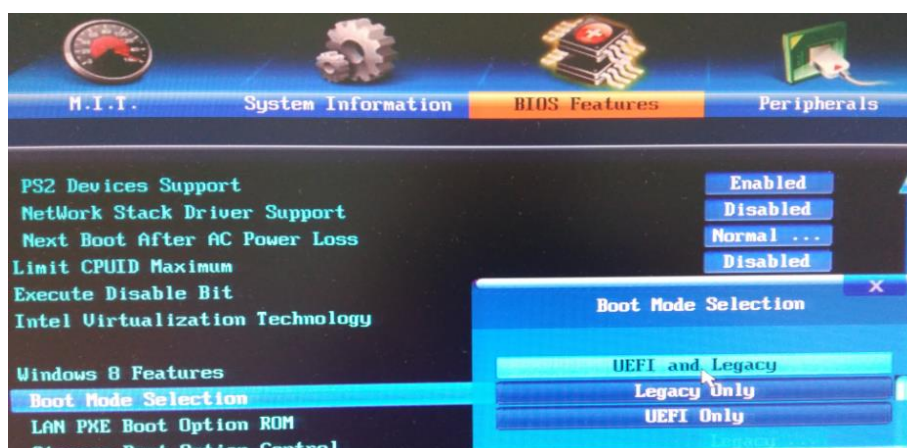
Ventajas más importantes:

- Las 3 nuevas tecnologías incluidas:
 - *Trusted Boot*: Ayudar a proteger el proceso previo al inicio frente a ataques de bootkit (virus de arranque: <https://www.incibe-cert.es/blog/bootkits>).
 - *Fast Startup*: Tiempo de inicio y reanudación desde la hibernación más rápidos. No quiere decir que arranque más rápido sino que despierta más rápido desde la hibernación. No confundir suspender (estado en RAM, quedando encendida) a hibernar (estado de RAM se pasa a disco, con lo que todo se apaga).
 - *Secure Boot* (comprobaciones sobre el sector de arranque).
- Compatibilidad con unidades de disco duro con particiones de arranque de más de 2 TB (límite que tenía BIOS).
- Solo permite instalar SSOO de 64bits.
- Define los nuevos discos GPT que evolucionan y mejoran los antiguos MBR. Es decir, una BIOS EFI pura solo es capaz de arrancar discos GPT (vamos a hacer ciertas puntualizaciones con respecto a este punto en el siguiente apartado).

DUAL BIOS Y BIOS LEGACY

Como siempre que conviven dos tecnologías (la estándar y la nueva) se intenta que puedan coexistir juntas, es decir, que se pueda elegir entre usar una u otra (pero nunca a la vez). La tendencia es utilizar BIOS duales donde se pueda elegir entre **BIOS UEFI** y la BIOS de toda la vida o **BIOS LEGACY** (legado de BIOS). Pero...¿qué ocurre si tenemos un disco MBR y contamos con una BIOS UEFI? Pues que podemos elegir entre el modo de funcionamiento:

- UEFI Only: Solamente será capaz de arrancar discos GPT.
- LEGACY Only (HEREDADO): Solamente será capaz de arrancar discos MBR.
- UEFI and LEGACY: La BIOS analizará el disco, si es MBR conmutará a BIOS LEGACY, si es GPT conmutará a UEFI.



Normalmente todas las placas nuevas cuentan con DUALBIOS y lo seguirán haciendo hasta que se dejen de utilizar los discos MBR.

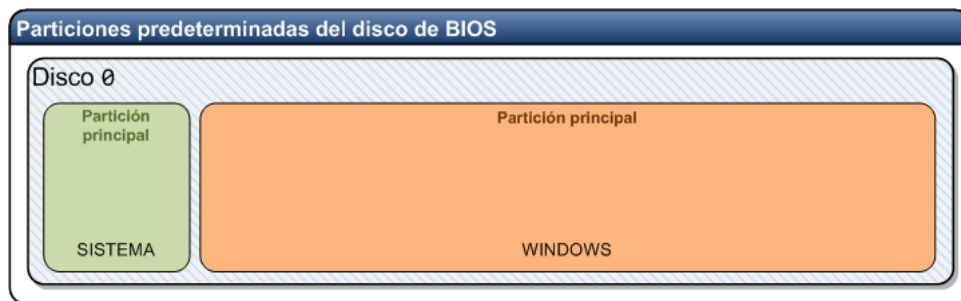
Cuando se realiza la instalación de Windows en un equipo con arranque UEFI, el esquema predeterminado de particiones es el siguiente:

- a) Partición del sistema (System): partición del sistema EFI 0 ESP. Se encuentra en el disco duro de arranque, siendo la primera que inicia.
- b) Partición reservada de Microsoft (MSR), partición reservada para la gestión de las particiones, que no puede almacenar datos de usuarios.
- c) Partición de Windows (Windows): partición que alojará el sistema operativo.
- d) Además: Partición de recuperación (Recovery): partición de herramientas de recuperación.



En las configuraciones con arranque heredado, el esquema ideal propuesto por Microsoft, es:

- a) Partición del sistema (System): ha de estar configurada como la partición actor disco de arranque
- b) Partición de Windows (Windows): partición que aloja el sistema operativo
- c) Partición de recuperación (Recovery): almacena las herramientas del entorno de recuperación de Windows.



RECUERDA....

Siempre es recomendable emplear un espacio de almacenamiento destinado a guardar **datos** por parte de los usuarios, diferente a las particiones definidas por defecto por Windows durante su instalación de manera predeterminada. Por tanto, es el administrador del sistema el encargado de crearla.

Además, la partición de **recuperación** no es imprescindible o se puede emplear junto con la partición de Windows, dependiendo del modo de arranque seleccionado y de la planificación de las particiones que el administrador del sistema prevea.

Para saber en qué modo está nuestro PC podemos, o bien entrar en la BIOS o mediante el comando `msinfo32`, a la derecha aparecerá si tenemos el Modo Heredado, UEFI, etc..

GPT

Dentro de la especificación EFI se encuentran los formatos GPT. Hasta ahora hemos visto cómo la forma tradicional de gestionar un disco es mediante el MBR. Un disco **GPT** (GUID Partition Table, siendo **GUID** acrónimo de Globally Unique Identifiers) es una estructura de particiones más nueva y flexible en comparación con MBR. Es importante indicar que GPT es el **FORMATO** del disco, es decir, un disco GPT podría ser transformado a MBR.

El gran fallo de GPT, es que **solo es compatible con los nuevos sistemas operativos** y los nuevos programas. Por poner un ejemplo, si instalamos Windows 7 64 bits configurando el disco como GPT, si luego queremos instalar un **sistema operativo anterior** en ese mismo disco duro, el propio sistema nos indicará en la instalación que no puede trabajar con el disco duro ya que está corrupto, y no podremos instalar el sistema (aunque hay formas de transformar un disco GPT a MBR utilizando la herramienta de bajo nivel DISKPART).

Hay versiones que pueden leer GPT pero no todas pueden arrancar desde GPT. Por ejemplo, todas las versiones de Windows 7 pueden leer/escribir en discos GPT, pero solo las versiones de 64 bits de Windows 7 pueden arrancar usando un disco de este tipo. (esto es por la limitación que se comentaba anteriormente, si se quiere arrancar desde un disco GPT usando EFI el SSOO tendrá que ser de 64 bits).

En Windows hay **128 entradas de partición** reservadas, cada una de 128 bytes de longitud (es decir, 128 letras de unidad). Así, se pueden crear hasta 128 particiones en un mismo disco si usamos un sistema tipo Windows+GPT.

DIFERENCIAS RESPECTO A LAS PARTICIONES EN MBR Y GPT

1. En MBR sólo pueden ser definidas 4 particiones primarias o 3 primarias + 1 partición extendida (con un número arbitrario de particiones lógicas dentro de la partición extendida).
2. MBR Utiliza 32 bits - el tamaño máximo manejable del disco hasta ($2^{32} \times 512 = 2 \text{ TB}$).
3. GPT número arbitrario de particiones (depende del espacio asignado por la tabla de particiones). No hay necesidad de particiones extendidas y lógicas. Por defecto, la tabla GPT contiene espacio para la definición de 128 particiones. Sin embargo, si el usuario desea definir más particiones, se puede asignar más espacio (de momento solo en Linux).
4. GPT utiliza 64-bit - tamaño máximo del disco manejable es de $2^{64} \times 512 = 9.4 \text{ ZB}$ (Zeta Bytes).

3.3. Dar formato a discos.

Hoy en día, tanto los discos internos como los externos ya vienen formateados de fábrica y disponen de un sistema de archivos como, por ejemplo, NTFS. Sin embargo, aún es posible toparse con situaciones que hacen conveniente, o incluso necesario, formatear un disco duro.

En principio, se requiere un formateo siempre que se quiera utilizar un **disco duro que aún no disponga de un sistema de archivos** o si quieres **cambiar el sistema de archivos instalado**. Si, por ejemplo, se sustituye el sistema FAT32 de un disco por un sistema NTFS, el soporte será capaz de albergar archivos más grandes, de más de 4 GB, y los archivos almacenados gozarán también de mayor seguridad, entre otras ventajas.

Por el contrario, cambiar de NTFS a FAT32 puede ser buena idea cuando se trata de alcanzar mayor movilidad y **compatibilidad entre plataformas para el intercambio de datos**. Un disco duro con FAT32, por ejemplo, puede leerse generalmente desde dispositivos domésticos como los lectores multimedia y las smart TV. Este sistema evita cualquier problema al transferir datos entre plataformas, ya que incluso los ordenadores Mac son compatibles, sin necesidad de herramientas externas, con discos duros formateados con el sistema FAT32.

Un disco duro recién formateado es una buena base en la que **reinstalar un sistema operativo** para asegurarse de que esté libre de virus y funcione sin problemas. Las actualizaciones, de Windows 7 a Windows 10, por ejemplo, también son una buena ocasión para instalar desde cero el nuevo sistema operativo en un disco que acabe de ser formateado.

Recordar que se puede distinguir entre dos tipos de formateo:

- **Formateo de alto nivel o lógico** → sus funciones son: comprobar la integridad de los sectores marcando los defectuosos para que no sean usados y reescribir la tabla de particiones. Elimina los archivos del disco, pero no de forma física, ya que los datos siguen ahí pero no son accesibles.
- **Formateo de bajo nivel o físico** → define el tamaño de los sectores y su ubicación en el disco. Los datos previos, si los hubieran, serán prácticamente imposibles de recuperar. Es un proceso lento. No suele ser necesario que los usuarios realicen este formateo ya que es el que viene de fábrica. Generalmente requiere programas especializados para realizarse.

→ Windows introduce otro tipo de formateo conocido como **rápido**: se limita a reescribir la tabla de particiones marcando como disponibles todos los clusters de la partición, es decir, sin comprobar si los sectores que los conforman están defectuosos o no.

Como hemos visto, en muchas ocasiones, un formateo de disco va acompañado de la creación de particiones. Con las **herramientas estándar de Windows** se pueden formatear discos duros internos y externos y realizar la gestión básica de particiones.

3.4. Gestores de arranque

Los pasos de arranque son distintos para cada familia de SSOO, ¿para qué podría servirnos conocer dichos pasos? La respuesta es sencilla, si hay algún problema en el arranque (virus, mala configuración, ficheros que faltan) el conocimiento de los pasos que debe llevar a cabo de forma normal nos permitirá identificar la posible fuente del problema. Conforme van evolucionando los procesos de arranque estos se van volviendo más herméticos y podemos toquetearlos menos.

Podemos considerar 3 familias de arranques de SSOO Microsoft:

- 1) La familia de **MSDOS** (MSDOS, Windows 3.11, W95, W98, WMe, etc...)

- 2) La familia de **NTLDR** (Windows 2000, XP, 2003 Server, etc..)
- 3) La familia de **BOOTMGR** (Vista, 7, 2008 Server, 2012 Server, 8, 8.1, 10, etc..)

La primera (MSDOS) necesitaba estar siempre instalada en C: y lo que hacía era leer una serie de ficheros instalados en el sector de arranque. La segunda lo que hacía era utilizar el gestor de arranque NTLDR para leer un fichero de arranque con la lista de los sistemas operativos instalados (**BOOT.INI**). En la tercera, se pasó a un gestor de arranque más avanzado (**BOOTMGR**), el cual no lee de un fichero de texto sino de una base de datos (**BCD**), aunque sí existe un fichero BCD. Las dos primeras familias, aunque didácticamente sería bueno entender su funcionamiento y carencias, no las vamos a ver porque ya están totalmente en desuso.

Arranque de Windows (BOOT MANAGER)

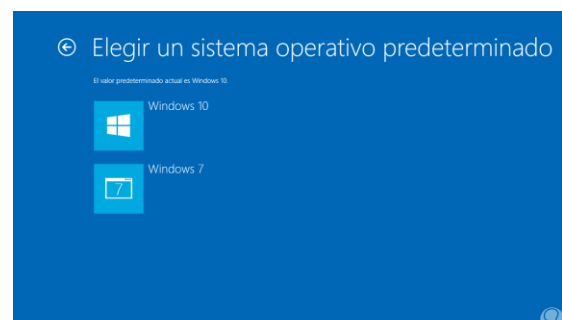
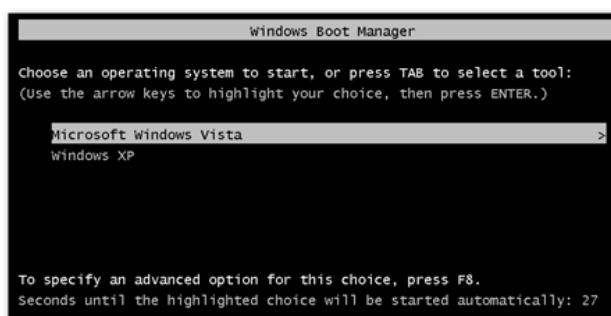
La principal diferencia con versiones anteriores estriba en que se ha cambiado el gestor de arranque, ya no se usa el NTLDR sino que se usa el Windows Boot Manager (**BOOTMGR**).

POST → BIOS → MBR/GPT → SECT.ARR. (BOOTMGR) → BCD.

Mientras que el gestor NTLDR usaba un fichero de texto denominado boot.ini para configurar sus opciones, BOOTMGR utiliza una base de datos conocida como **Boot Configuration Data (BCD)** que no puede ser editada directamente como lo era el boot.ini ya que no es un fichero de texto.

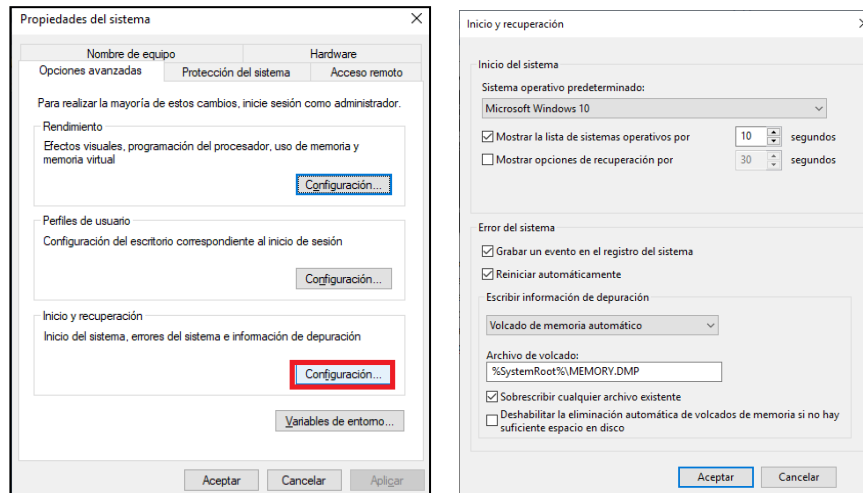
1. Se carga y ejecuta el **POST**
2. Se carga el **MBR** (o GPT) del disco duro (si es la opción elegida en la BIOS)
3. Se carga el **sector de arranque** de la partición primaria activa
4. Se carga el programa **BOOTMGR**.
5. **BOOTMGR** ajusta el procesador para trabajar a 32 bits o 64 bits.
6. **BOOTMGR** lee la base de datos **BCD** y muestra un menú si es necesario. El fichero BCD que guarda la base de datos suele estar en una partición oculta de 100/300/500MB según versión y cuando se instala en un disco duro vacío, en caso contrario se almacena en un archivo en la carpeta "\Boot". La ruta completa de este archivo es "[partición activa]\Boot\BCD". Para el arranque UEFI, el archivo BCD se encuentra en /EFI/Microsoft/Boot/BCD en la partición del sistema EFI. Para el arranque de BIOS tradicional, el archivo BCD se encuentra en /boot/ BCD en la partición activa.
7. El usuario selecciona un sistema operativo del menú, o se carga por defecto uno de ellos.
8. **BOOTMGR** carga **WINLOAD.exe** (Fichero de carga de Windows).
9. Winload.exe carga **NTOSKRNL.EXE** (Núcleo del sistema operativo o Kernel).
10. NTOSKRNL.EXE lee el **registro** de Windows, y procede a ir cargando el sistema completo.

A izquierda el aspecto de BOOTMGR en versiones como Vista/7, a la derecha 8.X/10:



Hay varias formas de intentar editar la configuración de BOOTMGR:

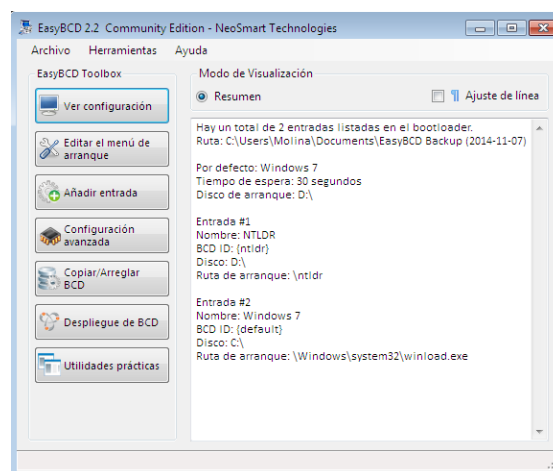
- Mediante las herramientas del propio sistema operativo, es decir, desde la pantalla anterior desde el comando **msconfig** o desde: Panel de Control -> Sistema y Seguridad -> Sistema -> Conf.Avanzada.Sistema -> Inicio y Recuperación -> Configuración.



- Mediante MSDOS editando entradas a mano mediante **bcdedit**: desde el modo a prueba de fallos o la consola de recuperación. Este método es el más complejo.
- Usando herramientas de terceros: editores (como Bellavista) que no nos dan demasiadas opciones o gestores más potentes (como **EasyBCD** o **BootICE**) que sí nos dejan tocar más la configuración.

Editando entradas con software [Presentes en HBCD PE x64.ISO]

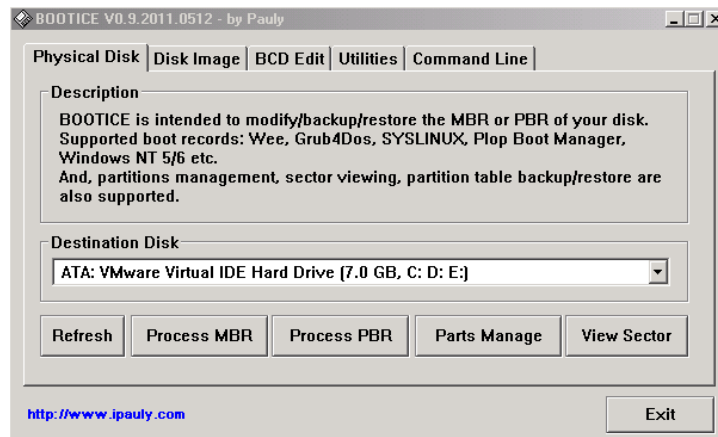
EASYBCD



Uno de los programas más útiles para hacer esto es EasyBCD. Lo primero que tenemos que hacer es “cargar el almacén” (la base de datos BCD) que queremos tratar. En distintas particiones podemos tener distintos BOOTMGR con distintas versiones de BCD, y dependiendo de cual se cargue en la partición activa me llevará a un sitio u otro. Esto es muy importante a la hora de diagnosticar problemas en el arranque y poder solucionarlos. Este programa me permite:

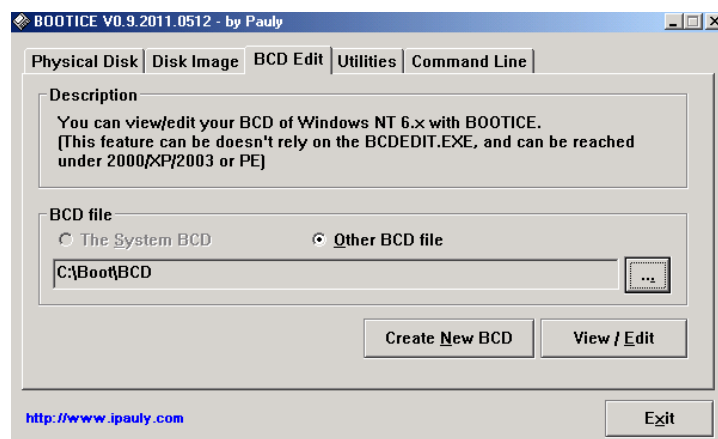
- Editar el menú: cambiando orden, nombres y tiempo de espera
- Añadir entrada: de casi cualquier tipo de sistema operativo
- Copiar/Arreglar BCD: la más interesante, puedo hacer copias de seguridad o incluso intentar arreglar el BCD.
- Despliegue de BCD: **permite crear una copia de BOOTMGR** en la partición indicada, es decir, copia el gestor de arranque en el disco.

BOOTICE



Este programa me permite:

- Gestionar el MBR (Process MBR) e instalar distintas tablas de arranque en el primer sector del disco (MBR de Windows, GRUB, PLOP, etc...). También puedo hacer copias de seguridad (guardar/cargar) del MBR (para guardarlo se ha de almacenar el sector 1) y para restaurarlo habrá que desmarcar el checkbox relacionado con la firma del disco.
- Gestionar el PBR (Partition Boot Record): más conocido como **gestor de arranque**, podemos instalar otro tipo de gestor con Install (pero ojo, NO INSTALA, lo único que hace es que apunte a un fichero u otro), manejar particiones... También puedo hacer copias de seguridad (guardar/cargar).
- Gestionar el BCD: podemos cargar el fichero BCD y editar las entradas, añadir nuevas, cambiar orden, etc... (lógicamente es más limitado que EASYBCD)
- Ver Sectores: podemos comprobar cómo el sector 0 está ocupado por "algo" (MBR)



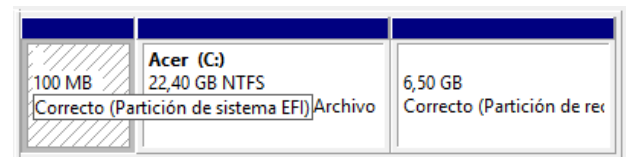
3.5. Arranque de Windows 8.x / 10 / 11

Aunque el arranque de Windows 8, 8.1, 10 y 11 es muy similar al de Windows 7 incorpora varias novedades, muchas de ellas basadas en el uso de **EFI** en lugar de BIOS. Las mas importantes son *Secure Boot*, *Trusted Boot* y *Fast Startup*. La partición EFI (de mínimo 100MB) almacena datos del arranque, drivers y todo lo necesario para asegurar estas 3 tecnologías. También contiene la BCD que luego veremos.



Secure Boot

Los ordenadores cuando encontraban el sector de arranque del SO que querían cargar, se limitaban a ejecutar dicho código, sin comprobar de ningún modo qué es lo que se está ejecutando.

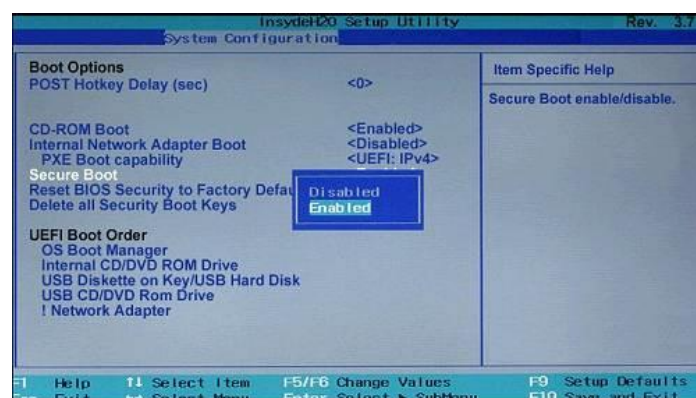





Sin embargo, si contamos en el sistema con UEFI en lugar de BIOS y esta activada una característica conocida como **Secure Boot**, el firmware del sistema comprueba la firma digital del sector de arranque, para comprobar si es de un sistema reconocido, y si se ha producido algún tipo de modificación sobre el mismo. Como consecuencia directa de esto: no se puede instalar un sistema desde un Pendrive booteable en modo UEFI.

Para permitir el arranque del sistema operativo, se debe dar una de las siguientes situaciones:

- ◆ El código de carga fue firmado utilizando un certificado “de confianza”. Por ejemplo, un certificado de Microsoft.
- ◆ El usuario aprueba la firma digital del código de carga (realiza alguna pregunta al respecto).
- ◆ El usuario deshabilita Secure Boot en la configuración de UEFI.
- ◆ El usuario deshabilita totalmente UEFI, y en su lugar utiliza BIOS. Aquí puede pasar:
 - i) que aparezca BIOS Legacy,
 - ii) que sigamos viendo el menú gráfico de EFI pero realmente la configuración se limite a las opciones de BIOS Legacy (y por tanto solamente pueda cargar discos MBR).

La función de *Secure Boot* es impedir la ejecución de cualquier software no firmado y certificado por el fabricante, por lo que cualquier amenaza que intentara atacar durante el inicio se vería frustrada, pues se detendría el arranque del sistema. Claro, esto por ejemplo deja fuera de juego la posibilidad de instalar distribuciones Linux. Por suerte, se puede desactivar desde la BIOS o desde la configuración del sistema operativo:



1. Selecciona **Inicio**  > **Configuración**  > **Actualizar & recuperación** > **de seguridad** .
2. En **Inicio avanzado**, selecciona **Reiniciar ahora**.
3. En **Elegir una opción**, selecciona **Solucionar problemas** > **Opciones avanzadas** > **Configuración del firmware de la UEFI** y, a continuación, selecciona **Reiniciar**.

Esto hará que se reinicie y se entre directamente en la BIOS UEFI.



Trusted Boot

Una vez que **Secure Boot** ha terminado su cometido, el código de carga (bootloader) verifica el firmado del kernel de Windows antes de cargarlo. A su vez, el kernel verifica todos los componentes de Windows que se van cargando, incluyendo los drivers de dispositivo de la propia Microsoft que se cargan en el arranque. Si un fichero ha sido modificado, el bootloader detectará el problema y se negará a seguir cargando el componente. También, en el caso de componentes corruptos en el arranque Windows intentará reparar el componente corrupto automáticamente.

Fast Startup

Windows **Fast Startup** (Inicio rápido) es la opción por defecto a utilizar en Windows 8.X/10/11 y Windows Server 201X siempre que se utilice UEFI.

En un sistema Windows en cada momento se encuentran ejecutándose dos sesiones en realidad, la del usuario actual y la del kernel del sistema. Cuando por ejemplo en Windows 7 se apaga el sistema, se cierran ambas sesiones y hay que volver a cargarlas desde cero cuando el sistema se inicia.

Windows cierra totalmente la sesión del usuario y la vuelve a cargar en cada inicio, sin embargo la sesión del kernel la hiberna, leyendo todo su estado en la RAM y grabándolo directamente en el disco duro. Esto permite que cuando el sistema se inicie, no se vea obligado a volver a leer todos los archivos del kernel, sino que directamente recupera el estado desde el disco duro hasta la RAM. Esto permite que se inicie Windows ahora mucho más rápido que con Windows 7.

Esta hibernación se realiza solo con la sesión de kernel porque es pequeña y predecible, mientras que no se realiza con la sesión de usuario porque suele ser mucho más grande, y es impredecible (igual ocupa muy poco que muchísimo).

3.6. Arranque de Linux: GRUB

Linux no cuenta con un gestor de arranque propio, sino que permite usar cualquier gestor de arranque que deseemos. El que se suele incluir actualmente en todas las versiones de Linux es el GRUB.

Es **MUY IMPORTANTE** indicar que los gestores de arranque de Linux no usan el esquema de partición **ACTIVA** (hasta ahora hemos visto que todos los Windows lo hacen). Dicho de otra forma, cuando se instala un gestor de arranque de Linux, da igual cual sea la partición activa ¿por qué? porque en la zona MBR (primeros bytes de los primeros 512B) se instala el GRUB que se ejecuta en varias fases para finalmente buscar la lista de sistemas a cargar (y esa la lista estará siempre en la partición de Linux). La tabla MBR no se toca en ningún momento, pero no se le hace caso.

El GRand Unified Bootloader (**GRUB**) es un gestor de arranque múltiple que se usa comúnmente para iniciar dos o más sistemas operativos instalados en un mismo ordenador. Otros gestores de arranque usados anteriormente en Linux son el *Syslinux* y el *Lilo*.

En la actualidad nos podemos encontrar con GRUB en sus versiones v1 y v2, que son algo distintas.

El proceso de inicio de GRUB v1 es el siguiente. Como en la mayoría de arquitecturas el gestor de arranque se encuentra en el MBR, el cual es de 512 bytes, no es suficiente para cargar en totalidad un sistema operativo. Por eso, el cargador de arranque consta de varias etapas:

1. La BIOS busca un dispositivo de inicio (como el disco duro) y pasa el control al registro maestro de inicio (Máster Boot Record, MBR, los primeros 512 bytes del disco duro).
2. El MBR contiene la FASE 1 de GRUB. Como el MBR es pequeño (512 bytes), la FASE 1 sólo se encarga de buscar y cargar la siguiente fase del GRUB (ubicado físicamente en cualquier parte del disco duro). La FASE 1 puede cargar ya sea la FASE 1.5 (que suele estar entre el sector 1 y el 63) o directamente la 2 (normalmente ya dentro de la partición)
3. GRUB FASE 1.5 está ubicada en los siguientes 30 kilobytes del disco duro. La FASE 1.5 carga la fase 2. Esta fase es optativa y normalmente no se usa (sobre todo porque no hay espacio para mucho).
4. GRUB FASE 2 (cargada por las FASES 1 o 1.5) recibe el control, y presenta al usuario el menú de inicio de GRUB. Este menú se configura mediante un fichero de texto con nombre **menu.lst**.
5. GRUB carga el kernel (núcleo) seleccionado por el usuario en la memoria y le pasa el control para que cargue el resto del sistema operativo.

GRUB 2 tiene un inicio similar a v1 pero sustituye el fichero **menu.lst** (que editamos manualmente) por un proceso modular, de modo que automáticamente se añaden los sistemas operativos y las opciones de los mismos. Ya veremos en profundidad estos gestores en los temas dedicados a GNU/Linux.



GRUB no es en realidad un gestor de arranque para Linux, sino un gestor de arranque para cualquier sistema operativo. De hecho, GRUB es perfectamente capaz de arrancar cualquier sistema operativo de la familia Windows sin ningún tipo de problemas.

En este enlace se explica cómo usar GRUB con discos tipo GPT:

[https://wiki.archlinux.org/title/GRUB_\(Español\)#Instrucciones_específicas_para_GUID_Partition_Table_\(GPT\)](https://wiki.archlinux.org/title/GRUB_(Español)#Instrucciones_específicas_para_GUID_Partition_Table_(GPT))

3.7. Recuperación de errores en el arranque

El proceso de arranque es un concepto al que cualquier informático debe prestarle mucha atención, dado que el más mínimo problema que se origine en dicho proceso, hará imposible que el sistema operativo arranque, y por lo tanto dejara inservible el sistema informático.

Las zonas que hay que vigilar y conocer cómo recuperar si es necesario, son el MBR, el sector de arranque de la partición primaria activa y el programa gestor de arranque que este situado en dichas zonas.

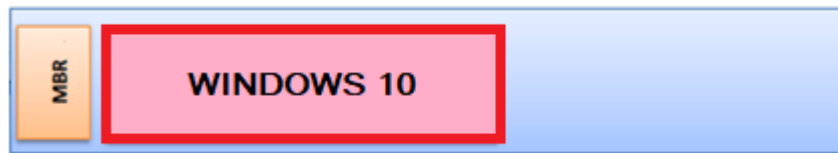
¿PERO, QUÉ ERRORES SE PUEDEN PRODUCIR EN EL ARRANQUE?

1. En primer lugar debemos hablar de los fallos de hardware. Al usar un disco duro siempre existe la posibilidad de que se corrompan clústeres del mismo. Normalmente estos errores no suelen tener demasiada importancia, pero si se da la casualidad de que se corrompe el primer clúster del disco duro, que es donde se sitúa el sector del MBR y el primer sector de arranque de la primera partición, nos vamos a encontrar en serios problemas. Normalmente en estos casos lo mejor es cambiar el disco duro completo, e intentar recuperar la información que existía en el disco duro con algún programa de recuperación de datos profesional.
2. En segundo lugar nos encontramos la acción del malware (virus, gusanos, troyanos, etc.). Estas amenazas pueden borrar el MBR y los sectores de arranque, y antiguamente existían bastantes virus que se dedicaban a realizar estas acciones
3. La tercera causa, y la que suele ser culpable en el 99% de los casos, es que directamente el usuario estropee el arranque de un sistema operativo, simplemente instalando cualquier programa, driver o sistema operativo. Veamos con detalle esta situación:

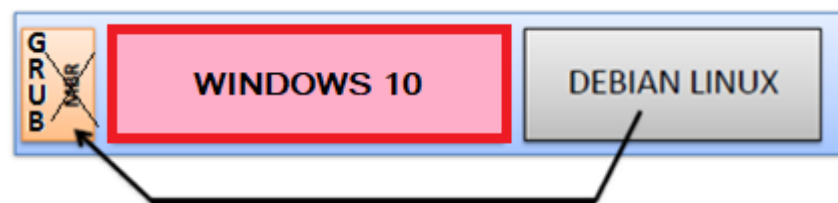
Hemos visto como cada sistema operativo cuenta con su propio modo de arranque (y formas de leer MBR, sector y gestor de arranque). Está claro que si instalamos en un mismo disco duro tres sistemas operativos distintos (por ejemplo), cada uno de ellos habrá ido instalando su propio proceso de arranque, pero como solo puede existir un proceso de arranque en un disco duro (sólo existe un MBR), el proceso de arranque que se quede al final será el del último sistema operativo instalado. Aquí tenemos dos posibilidades: si el SSOO no entiende lo que hay antes, machacará el proceso de arranque del sistema operativo anteriormente instalado, y así sucesivamente.

- ◆ Si es un Windows, mira la partición activa y si entiende lo que hay ahí, lo tendrá en cuenta (y añadirá una línea al gestor para que se pueda arrancar), si no machaca el sector de arranque con su propio gestor de arranque.
- ◆ Si es un Linux, no mira el MBR porque ya no existe, lo que hace es mirar los sectores de arranque de las particiones para tener en cuenta las instalaciones anteriores (de Windows). Si hay algún Linux instalado examinará el gestor de arranque que hay instalado en el primer sector del disco.

Imaginemos el caso siguiente: En un disco duro tenemos instalado una partición con Windows 10



En la partición de Windows 10 tendremos instalado los archivos que necesita el gestor de arranque de 10 para funcionar. Decidimos instalar en dicho disco duro una distribución de Linux como Debian, para lo cual le creamos una partición y procedemos a instalar dicho sistema operativo. Durante este proceso de instalación, DEBIAN instala en el MBR el gestor de arranque de DEBIAN (en este caso GRUB) y NO pone activa la partición (dado que en GRUB no existe este concepto). La tabla de MBR queda intacta, no así los primeros 512B (donde iba el MBR de Windows) donde GRUB ha instalado su primera fase.



La próxima vez que iniciemos la máquina, se cargará el gestor de arranque de GRUB, no el anterior que teníamos de 10. ¿Reconocerá el gestor de arranque de GRUB que en el disco duro existe un Windows y nos permitirá arrancar desde el, aparte de arrancar desde Debian? Pues en este caso sí, en el mundillo de los gestores de arranque, es conveniente recordar siempre estas pequeñas reglas:

REGLAS GESTORES DE ARRANQUE

1. GRUB es capaz de arrancar cualquier sistema operativo, por lo que respetará siempre (o al menos lo intentará) cualquier sistema operativo que hubiera en disco duro antes de que se instalara dicho gestor de arranque.
2. Los gestores de arranque de Windows nunca respetarán a Linux. De hecho, el gestor de arranque de Windows solo es capaz de arrancar automáticamente a sistemas operativos Windows, siendo muy complicado conseguir arrancar otros sistemas operativos no de Microsoft.
3. Los gestores de arranque de Windows reconocen solo a los del gestor de arranque que hay instalados anteriormente pero solamente a los inmediatamente anteriores, es decir, NTLDR entiende y es capaz de detectar los arranques anteriores, BOOTMGR entiende a NTLDR, pero BOOTMGR no es capaz de iniciar directamente un arranque anterior a NTLDR (Familia MSDOS). Habría que utilizar un programa como EasyBCD.
4. Todos los SSOO reescriben el MBR (reescriben el programa MBR para que encuentre el arranque gestor de arranque en cuestión, los Microsoft lo hacen para que encuentre la partición activa), lo que pasa es que en el caso de los Linux, además instalan su propio gestor de arranque donde suele estar el MBR (primer sector del disco) y no en la partición Linux (aunque esto depende de la distribución y sus posibilidades de instalación).
5. Cuando se dice que un gestor de arranque "machaca" a otro no quiere decir que elimine ningún fichero asociado al anterior gestor, lo único que hace es "instalarse" en el sector de arranque, es decir: el sector de arranque apuntará a la dirección donde está el ejecutable del gestor de arranque.

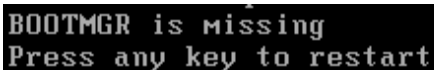
6. Lógicamente, si instalamos algo en una partición ocupada por otro sistema operativo, al formatear eliminará todo lo que ahí hay, incluidos posibles gestores de arranque y ficheros de configuración de los mismos.

EJERCICIO PRÁCTICO: Hacemos 3 particiones y lógicamente instalamos cada uno en su propia partición. ¿Darían algún problema? ¿Cuáles de estos sistemas operativos cargarían correctamente según el orden seguido? *Instalamos Windows 7 (incluye archivos de sistema), luego Linux y por último Windows 10.*

Nota: Windows puede instalar una o dos particiones. En el segundo caso crea una pequeña llamada reservada para el Sistema o de Sistema (con etiqueta Sistema en el gestor de discos) y otra más grande o llamada de Arranque (donde están todos los ficheros del sistema operativo y los de carga del sistema: winload, nt detect, etc...). Si se instala en una única partición, parte de los ficheros de Sistema van a una carpeta oculta llamada Boot y el resto van a la raíz del mismo.

Vamos a trabajar en intentar resolver estos problemas en los ejercicios de orden de instalación.

Cada sistema operativo cuenta con herramientas que permiten reconstruir el programa gestor de arranque en el MBR, y arreglar los sectores de arranque. Nos centramos en la familia BOOTMGR.



```
BOOTMGR is missing
Press any key to restart
```

Este error puede ser porque: No esté instalado el gestor de arranque en el sector de arranque, también porque falten o estén dañados la base de datos BCD, el BOOTMGR o el fichero BOOTSECT.BAK (Copia del sector de arranque donde se encuentra BOOTMGR).

Igualmente que en el punto anterior, tenemos que iniciar el sistema desde el CD original de instalación. (**O UNO DE LA FAMILIA BOOTMGR**). Llegará un momento en que el propio programa de instalación nos dará la opción de realizar una reparación automática del inicio de Windows. Escogemos esta opción y comprobamos si el sistema es capaz de repararse automáticamente. Si comprobamos que dicho automatismo falla (cosa bastante probable) volvemos a iniciar el sistema desde el CD, pero esta vez desde el menú avanzado escogemos la opción de **consola de recuperación o línea de comandos**. Desde allí podemos REPONER los ficheros necesarios borrados:

1. Si no podemos ejecutar ni *map* ni *mountvol* para identificar la unidad que tiene asignada el lector de CD, tendremos que hacerlo probando desde C: y listando con dir (ej: F). Otra forma es utilizar un programa gráfico que acceda al explorador de Windows, como es NOTEPAD, lo lanzamos y le damos a guardar para que podamos ver las unidades disponibles.
2. *F:\copy bootmgr c:*

Otra opción para crear el fichero bootmgr es usar comandos como **bcdboot** (además crea una copia de la base de datos BCD en la partición que se quiera).

Posteriormente podríamos ejecutar las siguientes órdenes (dependiendo de la naturaleza del problema en concreto):

BOOTREC.EXE /fixmbr : reconstruye el MBR.

BOOTREC.EXE /fixboot : Escribe un nuevo sector de arranque de la familia BOOTMGR en la partición activa (ojo, no copia BOOTMGR).

BOOTREC.EXE /scanos: Busca instalaciones previas de otros SSOO e informa de las mismas.

BOOTREC.EXE /rebuildbcd : Busca instalaciones en las particiones para añadirlas al BOOTMGR.

Las instrucciones que hemos visto anteriormente para Windows 7 funcionan exactamente igual en Windows 8.x/10.

Indicar por último que desde aquí también se podría intentar arreglar BOOT.INI (aunque no siempre funciona) mediante el comando: BOOTCFG (la lista de opciones es distinta desde esta consola de recuperación, por ejemplo, para listar las entradas se utiliza BOOTCFG /query). Siempre podemos editarlo a mano e indicar mediante las rutas ARC los sistemas a cargar.

LINUX

En este caso iniciamos el sistema desde un CD especial para recuperación del grub, como puede ser por ejemplo el “SUPER GRUB DISK” o “SUPER GRUB DISK2” (**La última versión es capaz de arrancar cualquier sistema operativo aunque su sector y/o gestor de arranque esté inservible**). También podemos recuperar el sistema arrancando desde un cd de una distribución “live”. Este tema lo dejamos para cuando nos hayamos familiarizado con Linux.

Muchos sistemas como Ubuntu añaden una línea en GRUB (modo de recuperación) desde donde se puede reinstalar GRUB.

3.8. Modos de arranque a Prueba de Fallos

En punto anterior hemos visto cómo podemos solucionar los fallos del arranque más importantes, que conllevan sobrescribir el MBR o bien el sector de arranque. Sin embargo existen otros muchos tipos de errores que se pueden producir en el inicio del sistema operativo, y que no se pueden solucionar con esas técnicas. Errores típicos de este tipo pueden ser la instalación de un driver corrupto, el borrado accidental de un fichero del sistema, etc.

Cuando un sistema no puede iniciarse debido a un error de este tipo, siempre podemos intentar iniciar por modos alternativos:

- ◆ A partir de los CD/DVD de instalación: **modos y consolas de recuperación** (para errores más graves)
- ◆ Utilizando **disquetes/CD de recuperación** propios de cada sistema operativo
- ◆ Modos a **prueba de fallos**: Es un modo especial conocido como modo a **prueba de fallos**, donde se cargarán las funciones básicas del sistema, intentando saltarse las partes que pueden estar provocando fallos. Para ingresar en el modo a prueba de fallos en un Windows, basta con pulsar la tecla F8 justo cuando el sistema inicia su carga o incluso se puede cargar desde el menú del gestor de arranque. Ejemplos de W98, XP y W7:


```

Menú Inicio de Microsoft Windows 98

1. Norma
2. Sesión iniciada (\BOOTLOG.TXT)
3. Modo a prueba de fallos
4. Confirmación paso a paso
5. Sólo símbolo de sistema
6. Sólo símbolo de sistema en Modo a prueba de fallos
7. Versión anterior de MS-DOS

Elija una opción: 1

F5=Modo a prueba de fallos Mayús+F5=MS-DOS Mayús+F8=confirmar paso a paso [N]

```

```

Menú de opciones avanzadas de Windows
Seleccione una opción:

Modo seguro
Modo seguro con funciones de red
Modo seguro con símbolo del sistema

Habilitar el registro de inicio
Habilitar modo VGA
La última configuración buena conocida (config. más reciente que funcionó)
Modo de restauración de SD (sólo contr. de dominio de Windows)
Modo de depuración
Deshabilitar el reinicio automático si hay un error en el sistema

Iniciar Windows normalmente
Reiniciar

Use las teclas de dirección Arriba y abajo para resaltar la opción.

```

```

Recuperación de errores de Windows

Windows no puede iniciar. Esto se puede deber a un cambio reciente en el
hardware o software. Para resolver el problema:

1. Inserte el disco de instalación de Windows y reinicie el equipo.
2. Elija la configuración de idioma y después haga clic en "Siguiente".
3. Haga clic en "Reparar el equipo".

Otras opciones:
Si se interrumpió la alimentación durante el inicio, elija Iniciar Windows
normalmente.
(Use las teclas de dirección para resaltar la opción que desee.)

Modo seguro
Modo seguro con funciones de red
Modo seguro con símbolo del sistema
La última configuración válida conocida (avanzada)
Iniciar Windows normalmente

Descripción: Iniciar Windows con su configuración normal.

ENTRAR=Elegir

```

Además, en W10, una vez que se pulsa F8 nos aparecerán las ventanas azules de configuración de arranque, tendremos que pulsar en “Opciones Avanzadas”→“Solucionar Problemas”→“Opciones Avanzadas”→“Configuración de inicio”.

Finalmente me aparecerá esta ventana:



Desde la ventana anterior se podrían cambiar opciones de inicio (como por ejemplo, no verificar la firma de drivers, etc..)

Al pulsar en Reinicio iremos directamente a las opciones de Recuperación (tercera imagen anterior en blanco y negro).

Vemos como además de los modos seguros, permite arrancar el sistema con otras configuraciones establecidas, como pueden ser con gráficos de baja resolución. Este menú aparece **de una forma u otra en todas las versiones de Windows**, aunque en Windows 8.X/10 hay que activarlo antes de poder usarlo. Hay varias formas de hacerlo, la más fácil es mediante bcdedit (como administrador. Para hacer esto ejecutamos CMD como administrador y luego ejecutamos el comando bcdedit):

- **bcdedit /set {default} bootmenupolicy legacy** (ACTIVA modo anterior de arranque, y por tanto permite usar F8)
- **bcdedit /set {default} bootmenupolicy standard** (ACTIVA modo normal, es decir, el de W8.X/10)

En Linux no tenemos un modo seguro como tal, pero podemos pasarle parámetros al kernel indicando como queremos lanzar nuestro Linux, desactivando por ejemplo los gráficos en alta resolución, el multiusuario, los puertos USB, etc.