TIPOS DE KERNEL Y SUS DIFERENCIAS

Generalmente se dividen en dos tipos, esenciales y no esenciales. Esto quiere decir que algunos tienen un uso mas común que otros. Veamos a continuación estas dos categorías.

ESENCIALES:

* Kernel monolítico: Es un núcleo grande que maneja todas las tareas del sistema operativo, incluyendo gestión de memoria, procesos y drivers. Su alta integración permite un rendimiento eficiente, pero esta misma complejidad puede hacer que sea más difícil de mantener y depurar. Ejemplos incluyen Linux, macOS y Windows.
* Microkernel: Se centra en ofrecer solo las funciones básicas necesarias, delegando otras tareas a módulos externos. Esto mejora la estabilidad del sistema al evitar que un fallo en un módulo afecte al núcleo, aunque puede incrementar la sobrecarga de comunicación entre el núcleo y los módulos. Mach en macOS y QNX son ejemplos representativos.
* Kernel híbrido: Combina características del kernel monolítico y del microkernel, ofreciendo una estructura grande pero modular que permite la carga dinámica de componentes. Esto proporciona un equilibrio entre rendimiento y flexibilidad, aunque puede ser más complejo y menos eficiente que un microkernel puro. Se encuentra en Windows NT y algunos sistemas basados en Linux.

NO ESENCIALES:

* Nanokernel: Es extremadamente reducido y se centra en proporcionar alta fiabilidad en sistemas embebidos, manejando solo las funciones mínimas necesarias. Esto garantiza una alta estabilidad en entornos de tiempo real, pero su uso está limitado a aplicaciones específicas debido a su funcionalidad reducida.
* Exokernel: Utiliza una estructura innovadora en la que el núcleo solo gestiona permisos y evita conflictos de hardware, permitiendo que los programas controlen directamente los recursos. Esto ofrece una gran flexibilidad y control para los desarrolladores, aunque puede aumentar la complejidad en la gestión de permisos y configuración. Se usa principalmente en investigación.
* Unikernel: Integra el kernel y las aplicaciones en una sola imagen, eliminando capas intermedias y simplificando el proceso. Esto resulta en alta eficiencia y bajo consumo de recursos, ideal para dispositivos IoT, aunque la falta de flexibilidad puede ser una desventaja en aplicaciones variadas.
* Anykernel: Busca mantener las características del kernel monolítico mientras facilita el desarrollo de controladores y mejora la seguridad del sistema. Aunque combina robustez y características modernas, aún está en fase conceptual e investigación, por lo que su implementación práctica es limitada.

USER VS KERNEL MODE

El procesador cambia entre estos modos en función del tipo de código que se está ejecutando. Las aplicaciones funcionan en modo de usuario, mientras que los componentes principales del sistema operativo funcionan en modo kernel.

MODO USUARIO:

En el modo de usuario, las aplicaciones se ejecutan en un entorno aislado con su propio espacio de direcciones virtuales, lo que impide que una aplicación modifique o interfiera con los datos de otras aplicaciones o con el sistema operativo. Este aislamiento garantiza que si una aplicación falla, no afecte a otras ni al sistema en general, y limita el acceso de las aplicaciones a direcciones de memoria reservadas para el sistema operativo, protegiendo los datos críticos.

MODO KERNEL:

En el modo kernel, el núcleo del sistema operativo y los controladores comparten un único espacio de direcciones virtuales, lo que significa que un controlador en este modo no está aislado de otros controladores ni del sistema operativo. Esto implica que un error en un controlador en modo kernel puede comprometer los datos del sistema o provocar un bloqueo total del sistema, ya que todos los componentes operan en un entorno menos seguro comparado con el modo de usuario.

INTERRUPTIONS VS TRAPS

INTERRUPTIONS: Una interrupción es una señal, ya sea de hardware o software, que indica que un evento requiere atención inmediata del procesador. Las interrupciones de hardware son generadas por dispositivos físicos, como teclados o tarjetas de red, y pueden ocurrir en cualquier momento de manera asíncrona. Las interrupciones de software, por otro lado, pueden ser generadas por el propio sistema operativo o aplicaciones para solicitar servicios o indicar eventos especiales. Cuando se produce una interrupción, el procesador interrumpe el proceso en curso, guarda su estado, y ejecuta una rutina manejadora de interrupciones para atender el evento. Después de manejar la interrupción, el procesador regresa al proceso original.

TRAPS: Una trampa es una interrupción generada por software que ocurre cuando un programa de usuario necesita invocar servicios del sistema operativo o enfrenta un error de ejecución. Ejemplos comunes incluyen una división por cero o un acceso ilegal a la memoria. Las trampas también se utilizan para solicitar servicios específicos al sistema operativo a través de llamadas al sistema. A diferencia de las interrupciones, las trampas son eventos síncronos que se producen durante la ejecución de instrucciones, lo que significa que están directamente asociadas con el flujo de ejecución del programa. Cuando se produce una trampa, el sistema guarda el estado del programa de usuario, cambia al modo supervisor, y ejecuta el código del núcleo del sistema operativo para manejar la solicitud.

BIBLIOGRAFÍA

(S/f). Thepower.education. Recuperado el 29 de agosto de 2024, de <https://thepower.education/blog/que-es-el-kernel-cual-es-su-trabajo-y-como-funciona>

*Modo de usuario y modo kernel*. (s/f). Microsoft.com. Recuperado el 29 de agosto de 2024, de <https://learn.microsoft.com/es-es/windows-hardware/drivers/gettingstarted/user-mode-and-kernel-mode>

*Trap vs Interrupt in Operating System - javatpoint*. (s/f). Www.javatpoint.com. Recuperado el 29 de agosto de 2024, de <https://www.javatpoint.com/trap-vs-interrupt-in-operating-system>