ARQUITECTURA Y SISTEMAS OPERATIVOS

CLASE 9

MÁQUINAS VIRTUALES

BIBLIOGRAFIA

- Operating System Concepts. By Abraham, Silberschatz.
  - Capítulo XVIII

TEMAS DE LA CLASE

- Descripción General
- Historia
- Beneficios y Características
- Bloques de Construcción
  - *Trap-and-Emulate*
  - Traducción Binaria
  - Asistencia del Hardware
- Tipos de Máquinas Virtuales y sus implementaciones
  - Ciclo de Vida de una Máquina Virtual
  - Type 0 Hypervisor
  - Type 1 Hypervisor
  - Type 2 Hypervisor
  - Paravirtualización
  - Virtualización del entorno de programación
  - Emulación
  - Contención de aplicaciones (Contenedores)

TEMAS DE LA CLASE

- Virtualización y componentes del sistema operativo
  - Planificación de la CPU
  - Administración de Memoria
  - E/S
  - Administración del almacenamiento
  - Migración en vivo

DESCRIPCIÓN GENERAL

DESCRIPCIÓN GENERAL

- **¿Qué es la Virtualización?**

La virtualización es una técnica que permite ejecutar múltiples sistemas operativos como si cada uno tuviera acceso exclusivo a los recursos del hardware físico. Esto se logra mediante una capa de software intermedio, conocida como VMM (*Virtual Machine Manager*) o **hypervisor**, que proporciona entornos aislados llamados máquinas virtuales (VMs).

- **Conceptos Básicos**

- Sistemas Operativos Huéspedes

Son los sistemas operativos que se ejecutan dentro de una máquina virtual. A diferencia de un sistema operativo nativo, que corre directamente sobre el hardware, un sistema operativo huésped corre sobre un entorno virtual proporcionado por el VMM.

DESCRIPCIÓN GENERAL

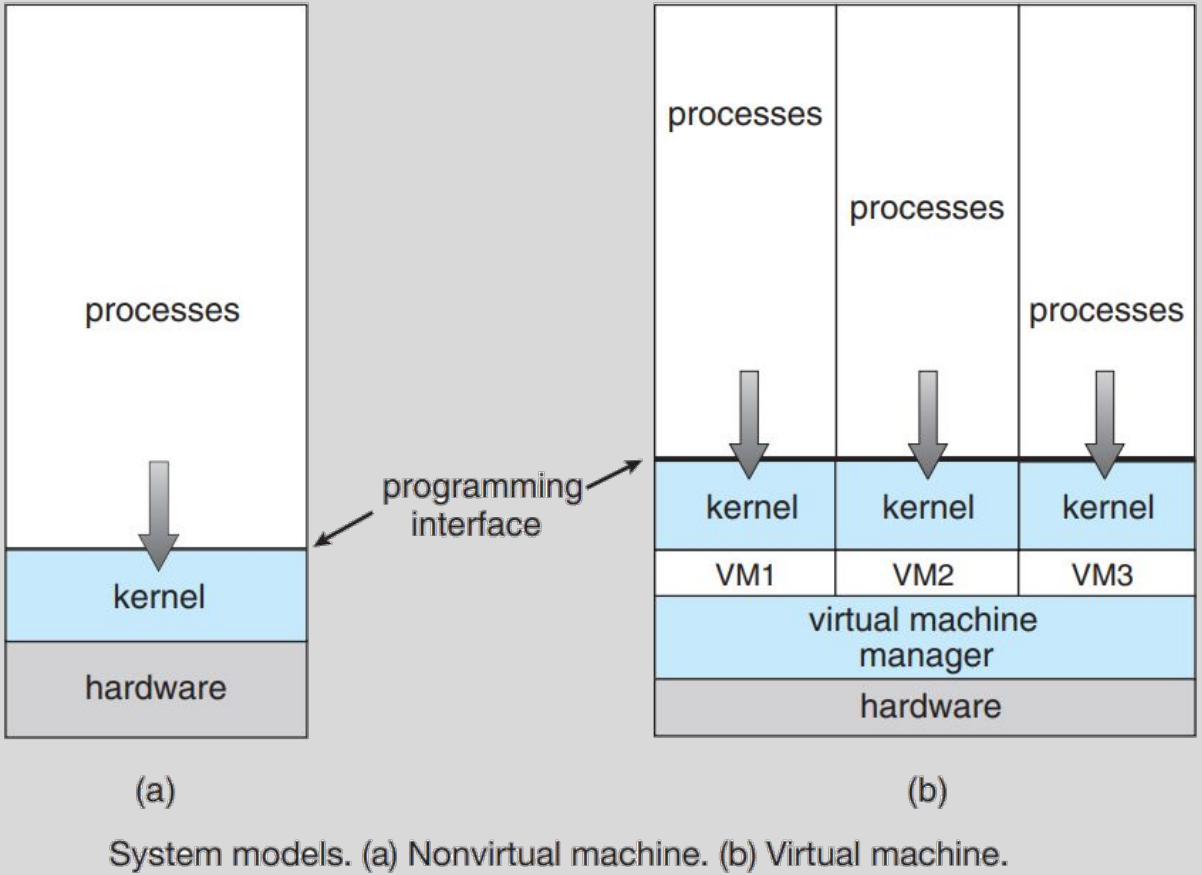
- Máquina Virtual

Una máquina virtual es una emulación de un sistema informático. Proporciona la funcionalidad de una computadora física pero implementada a través de software, utilizando los recursos físicos del host de manera compartida y administrada.

- Virtual Machine Manager (VMM) o Hypervisor

El VMM es responsable de crear y gestionar las máquinas virtuales. Controla los accesos de los sistemas operativos huéspedes al hardware del sistema, ya sea de forma directa o indirecta.

Por lo tanto, una sola máquina física puede ejecutar varios sistemas operativos simultáneamente, cada uno en su propia máquina virtual.



System models. (a) Nonvirtual machine. (b) Virtual machine.

HISTORIA

HISTORIA

Las VM surgieron en los mainframes de IBM en 1972. Con el tiempo, se formalizaron los criterios clave para definir virtualización:

- Fidelidad: La VM debe simular fielmente la máquina real.
- Rendimiento: La ejecución debe ser eficiente.
- Seguridad: El VMM controla completamente los recursos.

A fines de los 90, con el auge de las CPU Intel 80x86, tecnologías como VMware y Xen impulsaron la virtualización moderna.

BENEFICIOS Y CARACTERÍSTICAS

BENEFICIOS Y CARACTERÍSTICAS

- **Ventajas de la Virtualización**

- Aislamiento: Las VMs están completamente aisladas entre sí, evitando que fallos en una afecten a las demás.
- Consolidación: Permite ejecutar múltiples VMs en un solo hardware físico, aumentando la eficiencia.
- Facilidad de respaldo y recuperación: Las VMs pueden copiarse, clonarse o migrarse con facilidad.
- Flexibilidad: Es posible probar diferentes sistemas operativos y configuraciones sin afectar el sistema físico.

- **Virtualización y Seguridad**

El aislamiento que proporciona la virtualización también implica una barrera de seguridad. Una máquina virtual comprometida no debería afectar al host ni a otras VMs. No obstante, hay vulnerabilidades en ciertos entornos virtualizados que pueden romper ese aislamiento, por lo que, de todas formar, se deben seguir buenas prácticas de seguridad.

BLOQUES DE CONSTRUCCIÓN

BLOQUES DE CONSTRUCCIÓN

Implementar una VM completa es complejo. Algunos conceptos clave que se fueron dando a lo largo de los años:

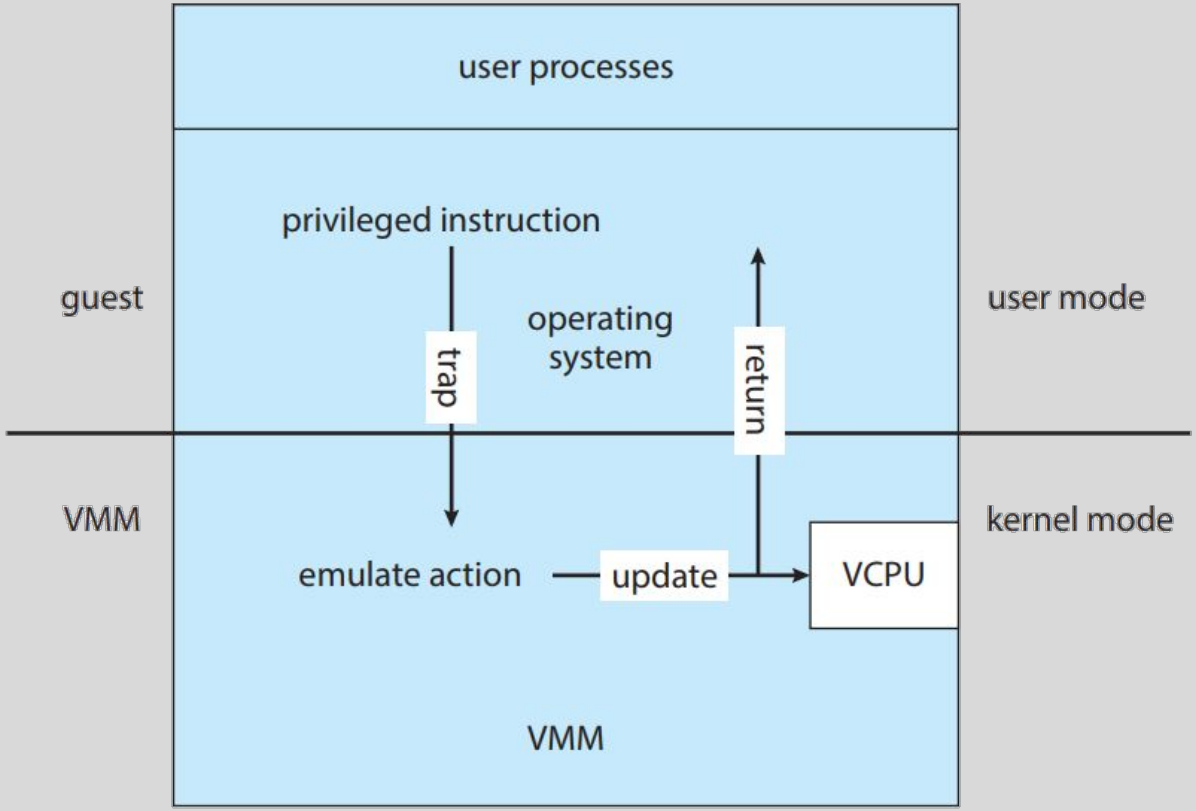
- vCPU (CPU Virtual): No ejecuta instrucciones, sino que representa el estado de la CPU real para el huésped. El VMM usa la vCPU para cambiar de contexto, similar al PCB en un SO tradicional.
- **Trap-and-Emulate**: En una arquitectura dual (usuario/kernel), el huésped se ejecuta solo en modo usuario. Si intenta ejecutar una instrucción privilegiada, genera una *trap*, que el VMM captura y **emula**. Finalmente, devuelve el control al huésped.
- **Traducción Binaria**: Para instrucciones que no generan una *trap*, la técnica anterior no funciona, en su lugar se utiliza traducción binaria. El VMM:
  - Reemplaza instrucciones especiales por código equivalente.
  - Usa caché para mejorar el rendimiento.
  - Este método es más complejo pero soluciona limitaciones del trap-and-emulate.
- Tablas de Páginas Anidadas (NPT): Permiten que el VMM controle las tablas de páginas del huésped. Sin hardware especializado, se implementan por software, pero con alto impacto en el rendimiento.

BLOQUES DE CONSTRUCCIÓN

- **Asistencia del hardware:** La virtualización moderna depende fuertemente del soporte de hardware.
  - CPU: Intel VT-x (desde 2005) y AMD-v (desde 2006) facilitan la virtualización sin necesidad de traducción binaria.
  - Permiten que el VMM cree máquinas virtuales estables y eficientes.
  - Gestión de Memoria: EPT (Intel) y RVI (AMD): Son técnicas que implementan tablas de páginas anidadas por hardware, reduciendo la sobrecarga del VMM.
  - E/S con DMA: Con DMA asistido por hardware, el VMM puede aislar la memoria de cada huésped dado que el hardware traduce direcciones de E/S para evitar interferencias entre huéspedes.
  - Interrupciones: La reasignación de interrupciones asegura que solo el huésped correspondiente reciba sus interrupciones, sin intervención del VMM.

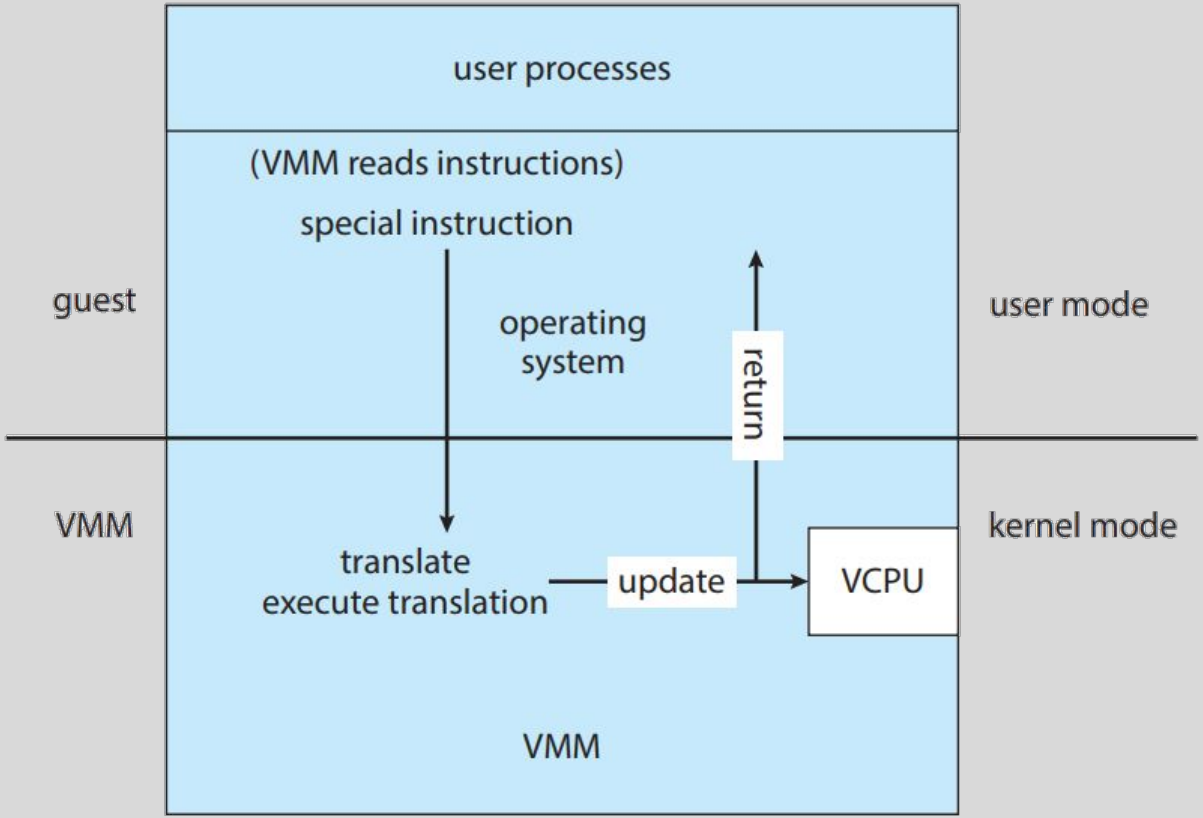
Gracias a la asistencia de hardware, hoy es posible crear hypervisors ligeros, eficientes y seguros. La virtualización ha cambiado profundamente la manera de administrar sistemas, permitiendo la escalabilidad y la agilidad requeridas por la nube y los centros de datos modernos.

- **Trap-and-Emulate**



Trap-and-emulate virtualization implementation.

- **Traducción binaria**



Binary translation virtualization implementation.

BLOQUES DE CONSTRUCCIÓN

- **Virtualización Total vs Paravirtualización**

- Virtualización Total

El sistema operativo huésped no necesita ser modificado. El VMM intercepta todas las instrucciones privilegiadas y las maneja de manera que parezca que el huésped corre sobre hardware real.

- Paravirtualización

El sistema operativo huésped se modifica para que sepa que está en un entorno virtualizado y utilice llamadas específicas al VMM. Esto permite un mejor rendimiento, aunque requiere acceso al código fuente del huésped.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

Analizaremos las implementaciones en general, con el entendimiento de que los VMM aprovechan la asistencia de hardware cuando está disponible.

- **El ciclo de vida de una máquina virtual**

Independientemente del tipo de *hypervisor*, el ciclo de vida de una máquina virtual se resume:

1. Creación: Se definen parámetros como CPU, memoria, red y almacenamiento.
2. Ejecución: El sistema operativo huésped corre dentro del entorno virtual.
3. Finalización: Se elimina la VM y se liberan sus recursos.

Estos pasos son bastante sencillos en comparación con crear, configurar, ejecutar y eliminar máquinas físicas. Incluso crear una máquina virtual a partir de una existente puede ser tan sencillo como hacer clic en el botón "Clonar" y proporcionar un nuevo nombre y dirección IP.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

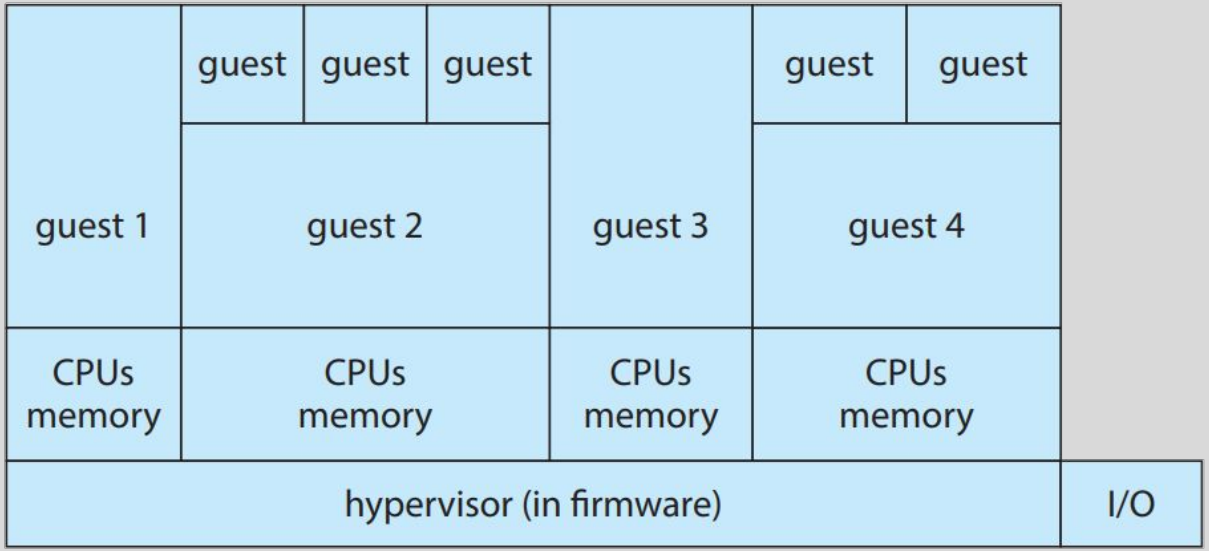
- **Type 0 Hypervisor**

Los sistemas operativos no necesitan hacer nada especial para aprovechar sus funciones. El VMM se implementa en firmware y se carga al iniciar. Esto genera que el conjunto de características de un hypervisor de tipo 0 sea menor que el de los otros tipos. Características:

- Implementado directamente en hardware (firmware).
- Se ejecuta antes que cualquier sistema operativo.
- Su funcionalidad es limitada en comparación con otros tipos.
- Todos los huéspedes deben tener sus propios dispositivos de E/S, o el sistema implementa una partición de control que comparte los dispositivos mediante servicios específicos. En esta partición, un sistema operativo huésped brinda servicios (como redes) a través de *daemons*, y el *hypervisor* enruta las solicitudes de E/S adecuadamente.

Así cada huésped actúa como si ejecutara sobre hardware real.

Dado que la virtualización de tipo 0 es muy similar a la ejecución de hardware puro, cada sistema operativo huésped, es un sistema operativo nativo con un subconjunto de hardware disponible. Por esto, cada uno puede tener sus propios sistemas operativos huésped.



Type 0 hypervisor.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Type 1 Hypervisor**

También conocido como bare-metal. Los *hypervisors* de tipo 1, en cierto sentido, se están convirtiendo en el "sistema operativo del centro de datos"

- Es un sistema operativo especializado que corre directamente sobre el hardware.
- Su objetivo no es ejecutar aplicaciones, sino máquinas virtuales.
- Proporciona planificación de CPU, administración de memoria, E/S, seguridad y una API de administración.

Ventajas:

- Alta consolidación de sistemas. Por ejemplo, en lugar de tener diez sistemas funcionando al 10 % de utilización cada uno, un centro de datos podría tener un solo servidor que gestione toda la carga.
- Migración de cargas de trabajo sin interrupciones.
- Balanceo automático de recursos.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Type 1 Hypervisor**

Este tipo 1 de *hypervisor* también incluye a varios sistemas operativos de propósito general con funcionalidad VMM. Son sistemas operativos con sus funciones habituales y, a la vez, proporciona una VMM que permite que otros sistemas operativos se ejecuten como huéspedes.

Estos *hypervisors* suelen ofrecer menos funciones de virtualización que otros de tipo 1. En muchos sentidos, tratan a un sistema operativo huésped como un proceso más, pero proporcionan un manejo especial cuando este intenta ejecutar instrucciones especiales.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Type 2 Hypervisor**

Son administradores de máquinas virtuales a nivel de aplicación, donde la participación del sistema operativo es mínima. Se ejecuta como una aplicación sobre un sistema operativo existente, e incluso el host desconoce que la virtualización se está produciendo dentro del VMM.

- Ejemplo típico: **VMware Workstation**.
- Tiene menor rendimiento comparado con los otros tipos debido a la sobrecarga del sistema operativo host.
- Ventaja: Portabilidad y facilidad de uso para usuarios sin privilegios administrativos.
- Útil para pruebas, entornos de desarrollo o educación.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Paravirtualización**

La paravirtualización funciona de forma diferente a otros tipos de virtualización. Aquí el huésped debe modificarse para ejecutarse en el hardware virtual paravirtualizado. La ventaja de este trabajo adicional, al eliminar la necesidad de emular completamente el hardware, es un uso más eficiente de los recursos y una capa de virtualización más pequeña.

Con la evolución del hardware, hoy ya no se requieren huéspedes modificados y, en esencia, no necesita el método de paravirtualización. Sin embargo, todavía se utiliza en otras soluciones, como los *hypervisors* de tipo 0.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Virtualización del entorno de programación**

Si definimos la virtualización como la duplicación de hardware, esto no se trata realmente de virtualización, pero la idea es no limitarnos a esa definición. Es posible definir un entorno virtual, basado en API, que proporciona las características que deseamos tener disponibles para un lenguaje específico y los programas escritos en ese lenguaje.

Aquí no se virtualiza hardware, sino un entorno de ejecución. Un ejemplo claro es la Java Virtual Machine (JVM), que permite ejecutar aplicaciones Java en múltiples plataformas.

- Proporciona gestión de memoria, seguridad y portabilidad.
- Lenguajes interpretados como Python también siguen esta lógica, donde el programa corre dentro de un intérprete.
- Aunque no es virtualización en el sentido tradicional, sigue el principio de abstracción de entorno de ejecución.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

- **Emulación**

La virtualización es probablemente el método más común para ejecutar aplicaciones diseñadas para un sistema operativo en otro, pero en la misma arquitectura de CPU. A diferencia de la virtualización, donde huésped y host comparten arquitectura, en la emulación, el sistema huésped puede tener una arquitectura completamente diferente.

Usos típicos:

- Ejecutar software antiguo.
- Probar sistemas diseñados para otras arquitecturas.

Desventaja principal: el rendimiento, ya que simular otra arquitectura requiere traducir cada instrucción, lo cuál suele implicar la ejecución de muchas más instrucciones.

Ejemplo: Emuladores de consolas de videojuegos.

TIPOS DE MÁQUINAS VIRTUALES Y SUS IMPLEMENTACIONES

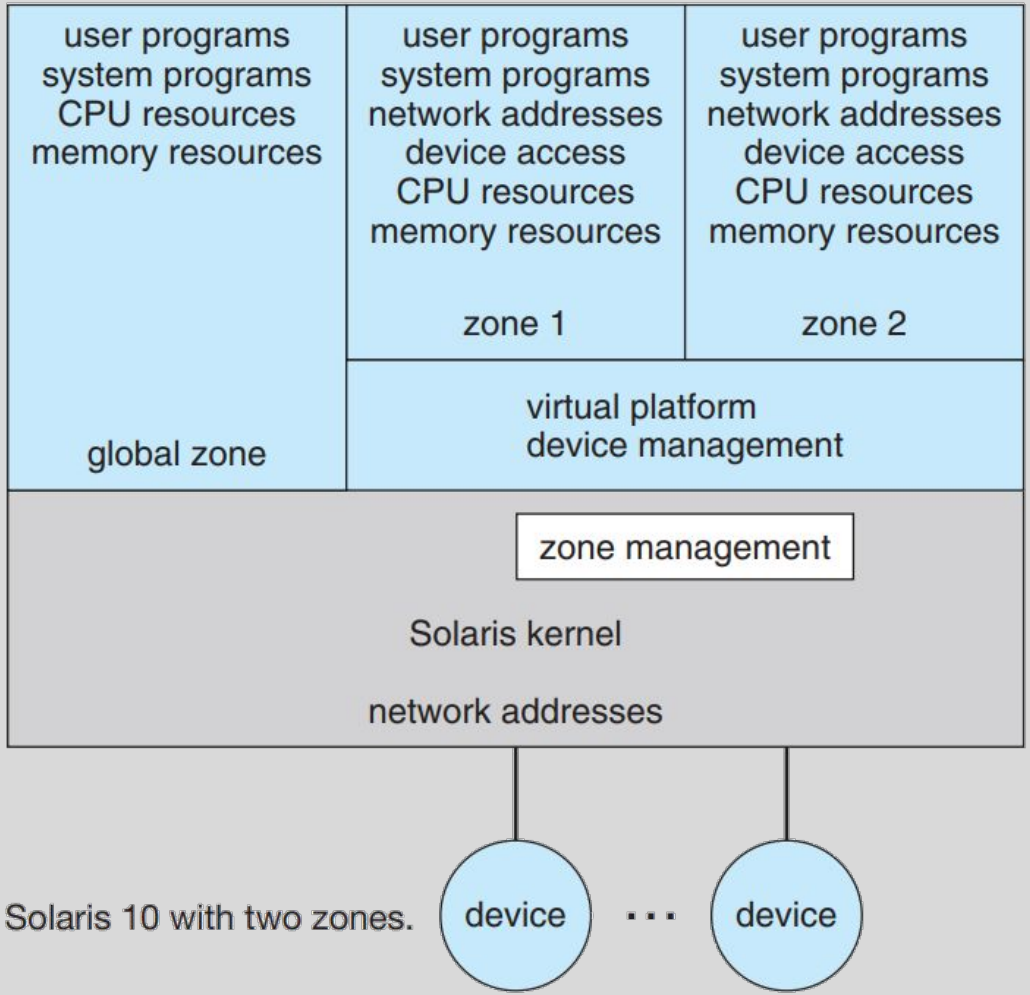
- **Contención de Aplicaciones (Contenedores)**

A veces, el objetivo de la virtualización es proporcionar un método para separar aplicaciones, gestionar su rendimiento, el uso de recursos, y crear una forma sencilla de iniciarlas, detenerlas, moverlas y administrarlas. Para estos casos, tal vez no se necesite una virtualización completa. Cuando todas las aplicaciones están diseñadas para el mismo sistema operativo, no es necesario virtualizar el hardware. En cambio, se puede usar un sistema de contenedores.

- Ejemplo: Zonas de Oracle Solaris
  - Virtualizan el sistema operativo, no el hardware. Dando la impresión de que los procesos dentro de un contenedor son los únicos procesos del sistema.
  - Cada contenedor actúa como un sistema aislado.
  - Pueden tener su propio planificador de recursos.
  - Mucho más livianos que las máquinas virtuales completas.

Los contenedores son mucho más ligeros que otros métodos de virtualización, y son fáciles de automatizar y gestionar, lo que da lugar a herramientas de orquestación como Docker y Kubernetes.

Estas herramientas permiten automatizar y coordinar sistemas y servicios. Su objetivo es simplificar la ejecución de conjuntos completos de aplicaciones distribuidas, al igual que los sistemas operativos simplifican la ejecución de un solo programa.



Solaris 10 with two zones.

MIGRACIÓN EN VIVO

MIGRACIÓN EN VIVO

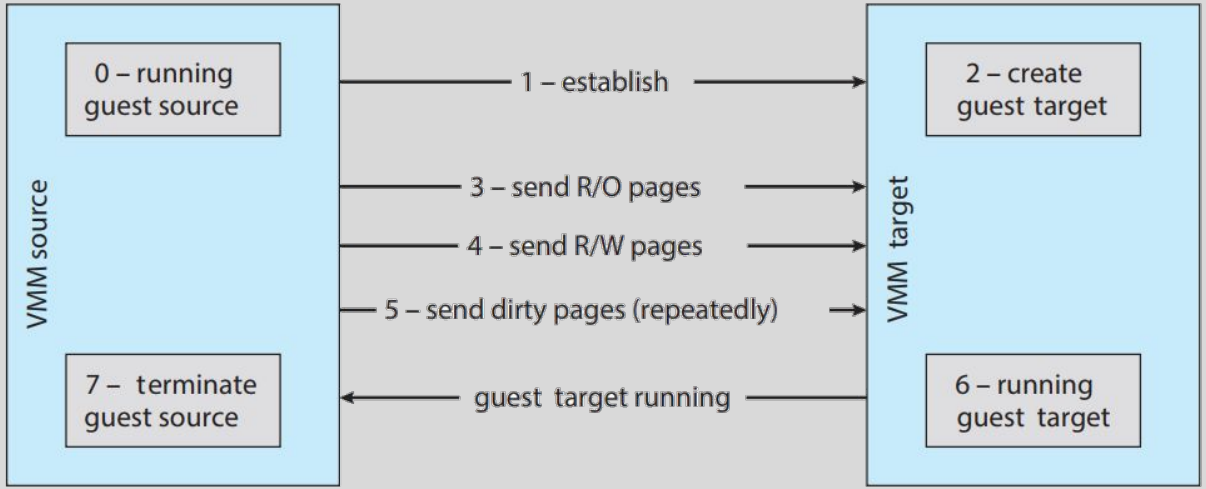
Es una funcionalidad avanzada disponible en hypervisors tipo 0 y tipo 1. Permite mover una máquina virtual en ejecución de un host a otro sin interrupción perceptible para el usuario.

El VMM migra un huésped mediante los siguientes pasos:

1. El VMM de origen contacta al destino.
2. Se crea el entorno de ejecución en el nuevo host.
3. Se copian las páginas de memoria (primero las de solo lectura).
4. Se copian las páginas de lectura/escritura (y se marcan como limpias).
5. Se repite el paso anterior para las páginas “sucias”.
6. Se congela el invitado, se transfiere el estado final, y se lo ejecuta en el nuevo host.

Posibilita optimizar el uso de recursos, hacer mantenimiento sin cortes de servicio, y balancear cargas de trabajo.

La migración en vivo permite gestionar los centros de datos de formas completamente nuevas. Por ejemplo, las herramientas de gestión de virtualización pueden supervisar todos los VMM de un entorno y equilibrar automáticamente el uso de recursos moviendo invitados entre ellos.

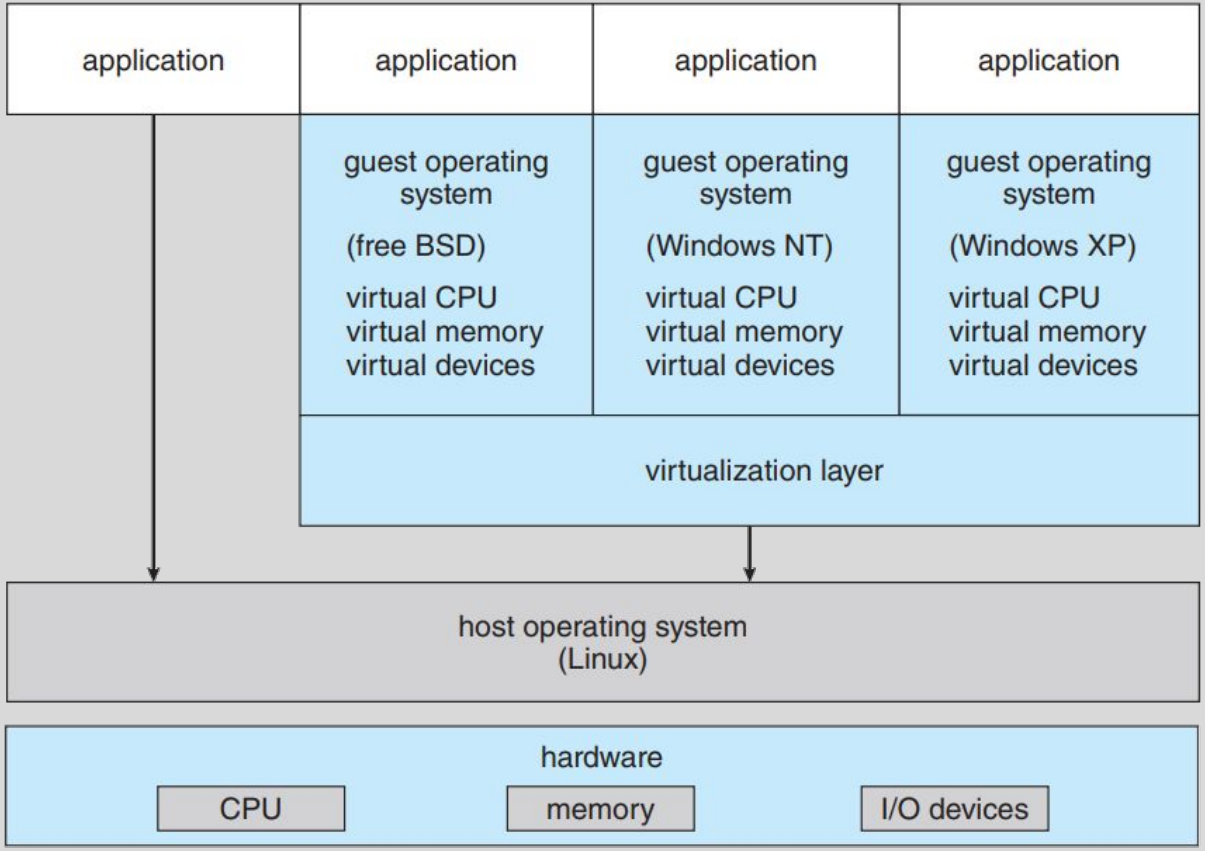


Live migration of a guest between two servers.

EJEMPLO HYPERVISOR TYPE 2

- **Ejemplo de Hypervisor Tipo 2**: VMware Workstation

- Aplicación comercial que permite ejecutar múltiples VMs sobre hardware Intel x86.
- Funciona como una aplicación más dentro del sistema operativo.
- Ideal para entornos de pruebas, desarrollo o estudio.
- Permite crear, ejecutar, pausar, detener, y clonar máquinas virtuales de forma simple.



VMware Workstation architecture.

Muchas Gracias

Jeremías Fassi

Javier E. Kinter