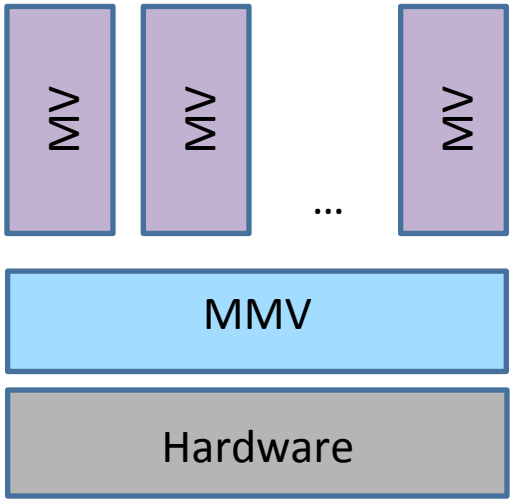
Infraestructura Computacional

Virtualización

Máquinas Virtuales

- Mecanismo para compartir recursos ofreciendo ambientes de ejecución (MV) aislados

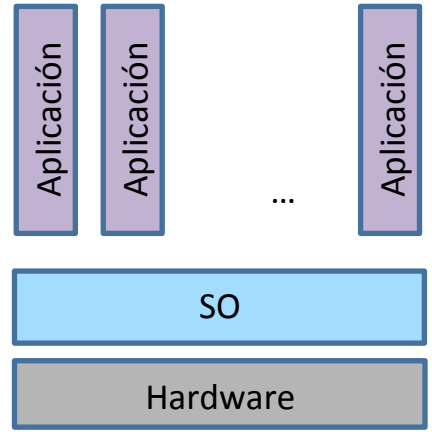


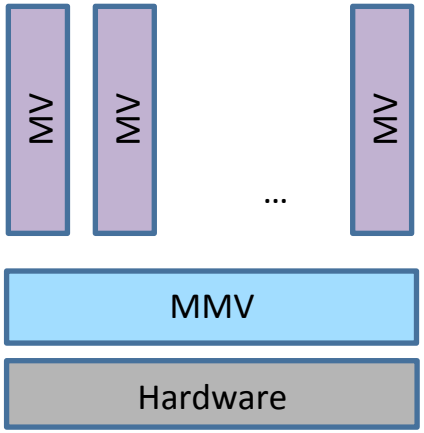
Ventajas de las Máquinas Virtuales

- Aislamiento
  - Si un servicio falla, las otras máquinas continúan funcionando
- Mayor flexibilidad para administradores y usuarios
  - Soporte a diversos requerimientos de los usuarios con un costo menor

Procesos vs. Máquinas

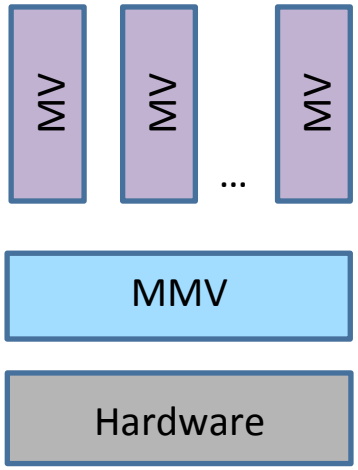
- Un solo sistema corriendo múltiples aplicaciones y servidores sobre una máquina compartida
  - SO:
    - Asignación, demanda de memoria, tráfico de red y acceso a disco de un proceso afectan el desempeño de otros
    - La contabilidad de todos los recursos usados por un servicio no es una tarea sencilla
  - MMV:
    - Correr sistemas operativos completos es más costoso en términos de inicialización y consumo de recursos





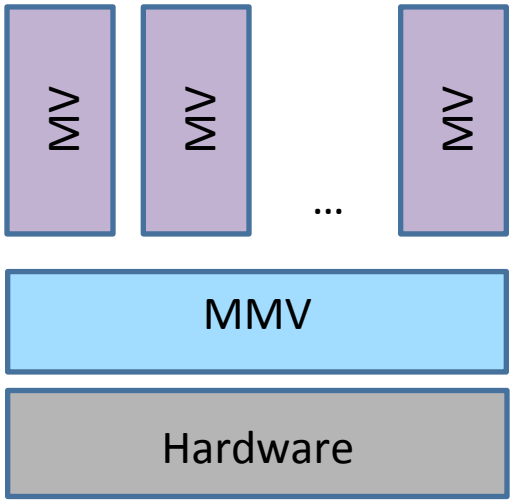
Máquinas Virtuales

- Cada ambiente de ejecución (MV) puede correr su propio sistema operacional
  - Cada ambiente de ejecución se conoce como huésped (guest)
  - El servidor sobre el que corren los huéspedes se conoce como anfitrión (host)
  - El programa de software que corre en el host y controla la ejecución de los guests se conoce como monitor de máquina virtual (MMV) o hipervisor



Requerimientos

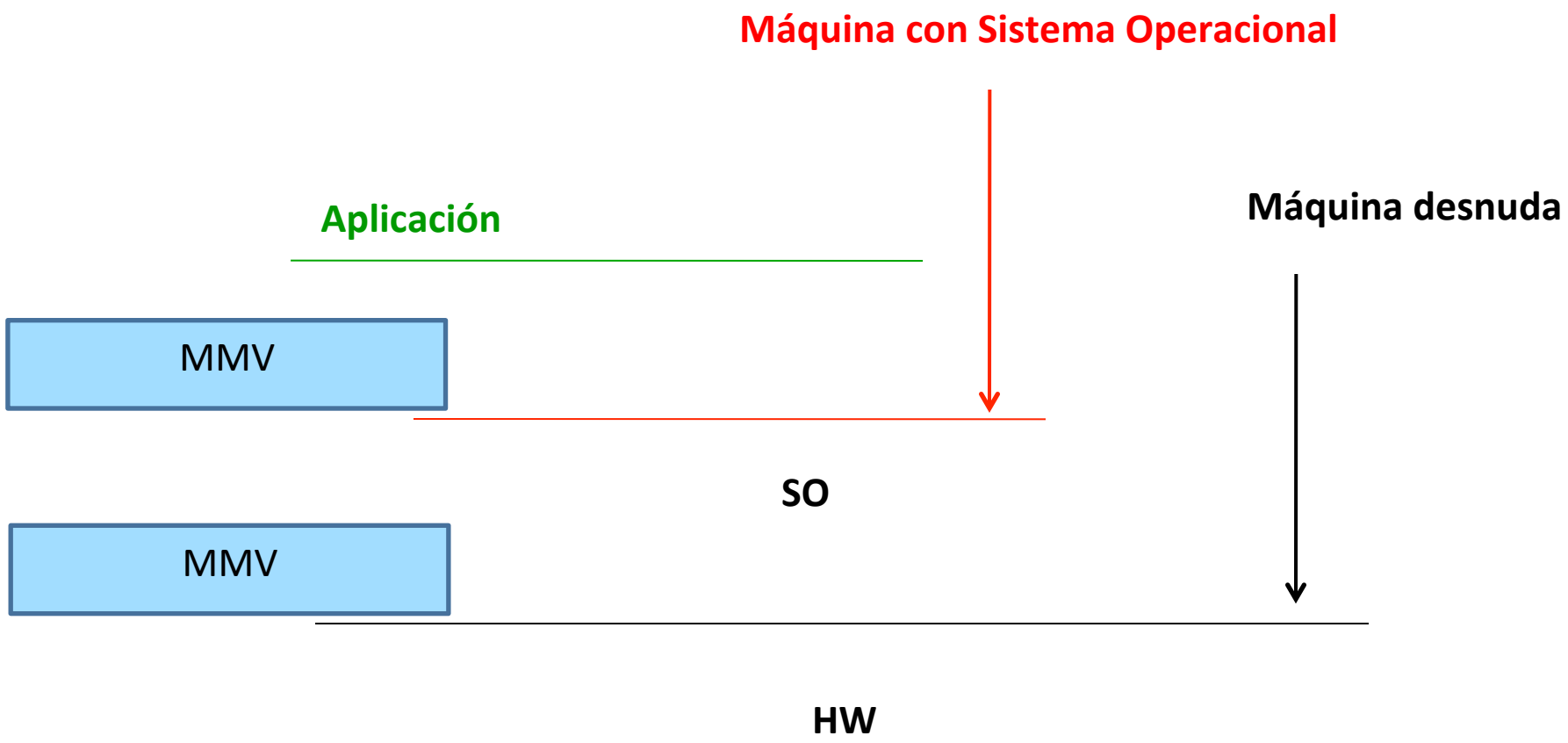
- Máquina virtual:
  - Duplicado, eficiente y asilado, de una máquina real.
- Monitor de máquina virtual - MMV (Virtual Machine Monitor - VMM):
  1. Pieza de software que provee un ambiente esencialmente idéntico a la máquina original
  2. Mínimo incremento en el desempeño
  3. El MMV tiene control total de los recursos



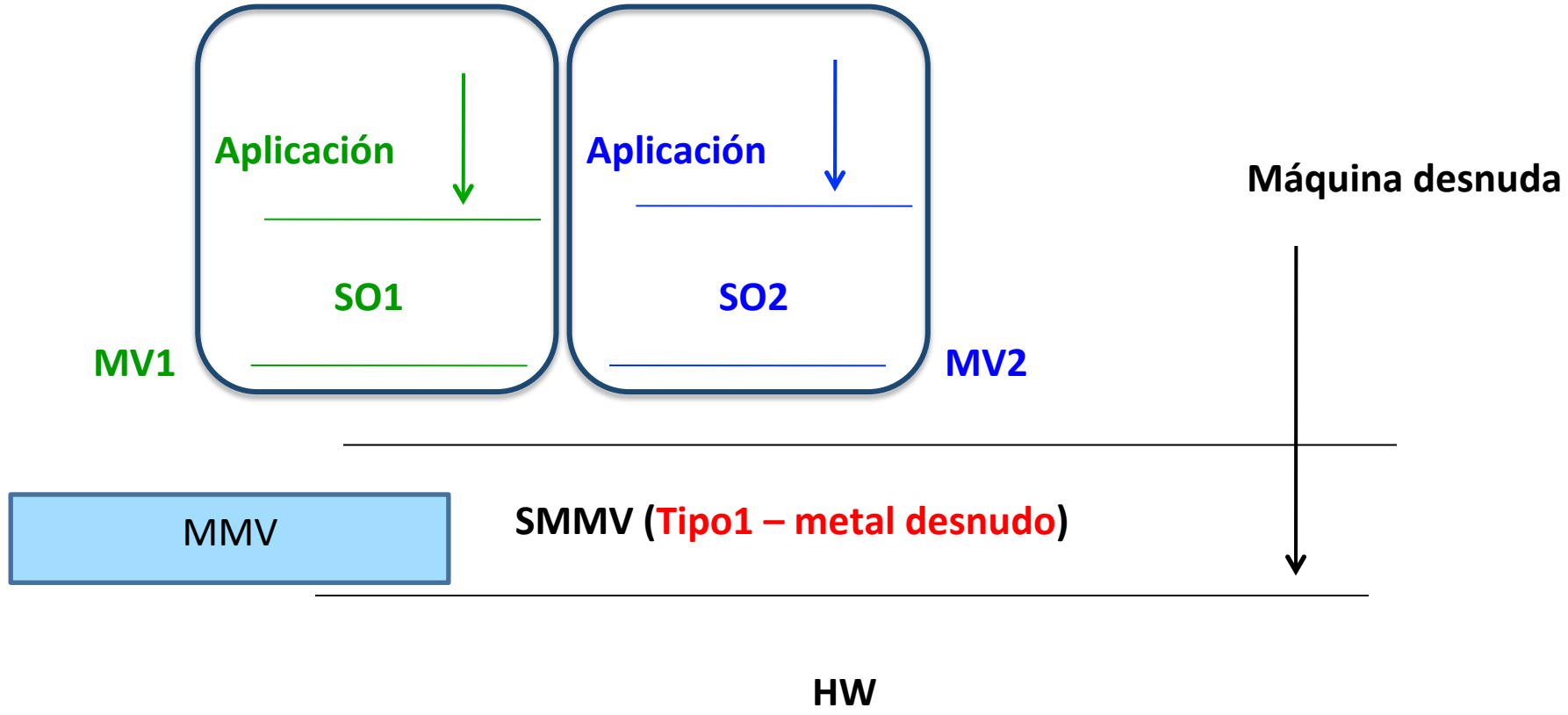
Requerimientos

- Ambiente idéntico
  - Los programas deben presentar un comportamiento idéntico, excepto por el incremento en tiempo de respuesta debido a la concurrencia
- Eficiencia
  - Un subconjunto dominante de instrucciones debe ser ejecutado directamente en el procesador para minimizar el detrimento en el desempeño
- Control de recursos
  - El MMV tiene control de todos los recursos. Los asigna y los recupera en cualquier momento.

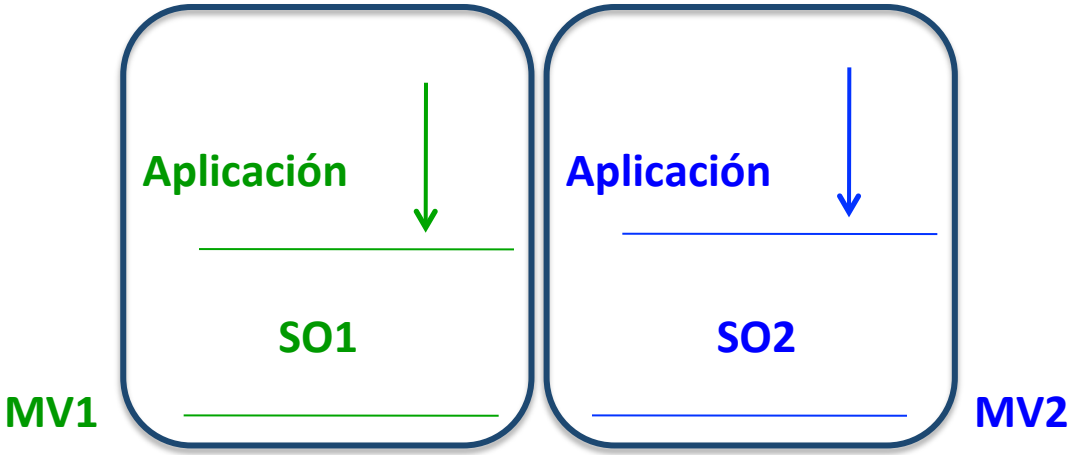
Implementación



Hipervisor Tipo 1



Hipervisor Tipo 2



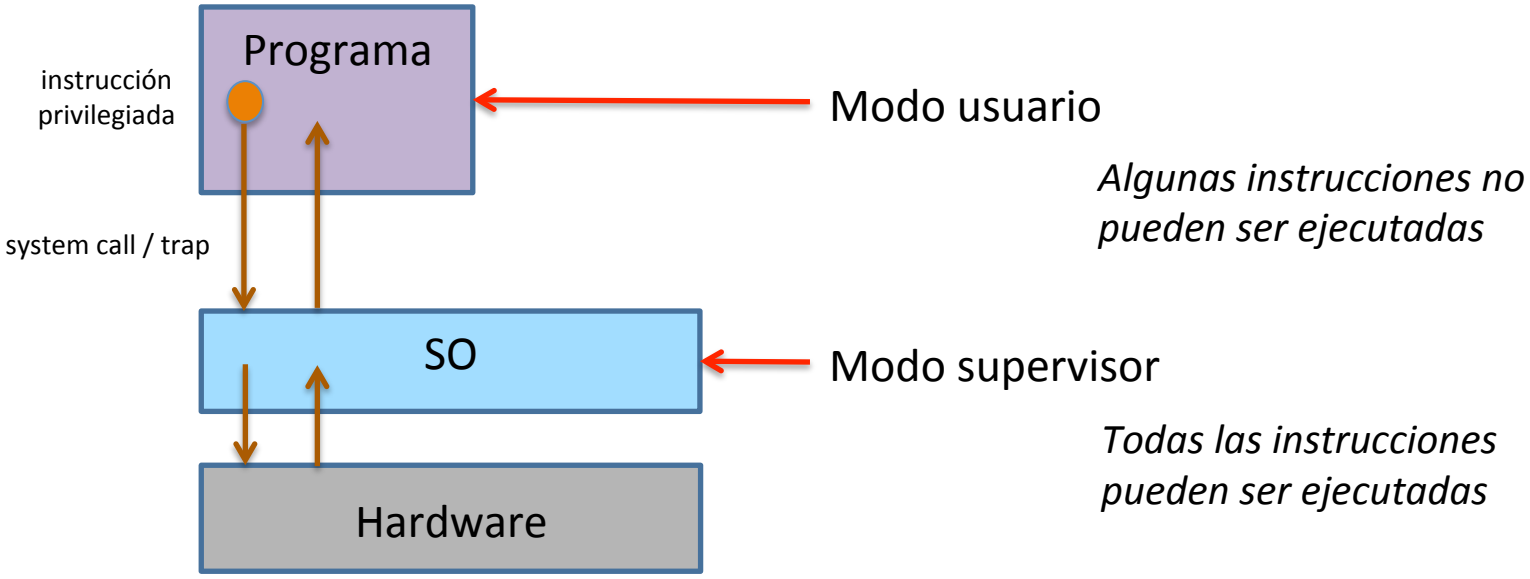
MMV

SMMV (Tipo2 – hosted)

SO Host

HW

Implementación de MV



SO Tradicional

Instrucciones Sensitivas

- Conjunto determinado de instrucciones que controlan el estado de la máquina
  - Modo del procesador
  - Registros de relocalización
  - Instrucciones de E/S

Instrucciones Privilegiadas

- Conjunto determinado de instrucciones que solo se ejecutan si el procesador corre en modo supervisor
  - Si son ejecutadas en modo usuario se produce una trampa y el sistema operacional toma el control

Usuario

Supervisor

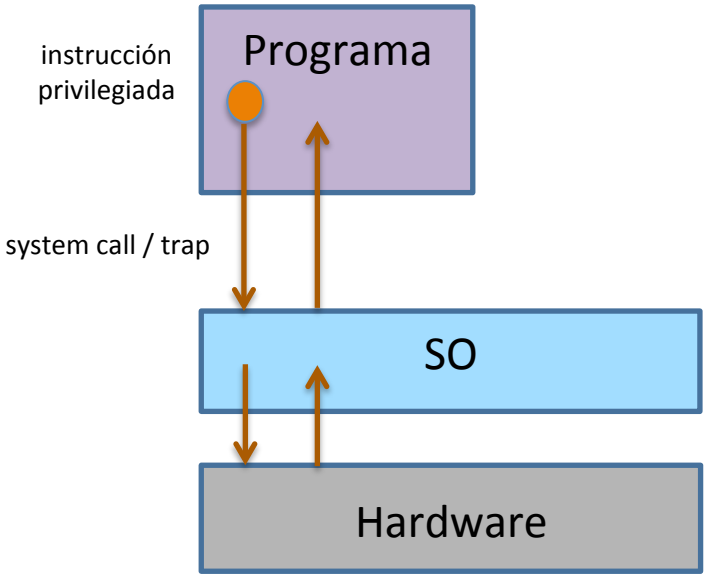
Trampas (trap)

- Una instrucción va a una trampa:
  - El control pasa automáticamente a la rutina de manejo
    - Cambio en el modo del procesador
      - Cambio de modo usuario a modo supervisor
    - Almacena el contador de programa y otros valores actuales
    - Salto a la rutina de manejo
    - Retorno al contexto inicial
    - Cambio en el modo del procesador
      - Cambio de modo de supervisor a usuario

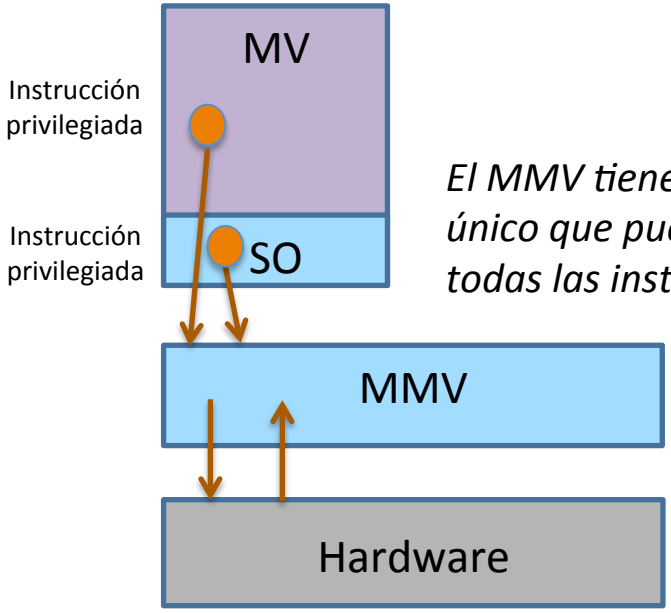
Instrucciones Sensitivas y Privilegiadas

- Todas las instrucciones sensitivas deberían ser privilegiadas
  - Controlan el estado de la máquina
  - Deben ser controladas en un ambiente multiprocesador o multiusuario para garantizar "consistencia"

Implementación de MV



SO Tradicional



El MMV tiene el control. Es el único que puede ejecutar todas las instrucciones.

Ambiente con Máquinas Virtuales

Hipervisores Tipo1

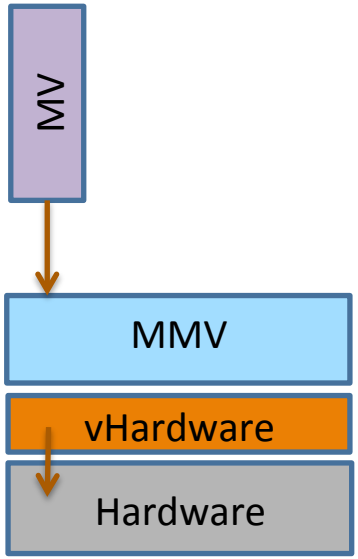
- El MMV tiene control total de los recursos
  - Cuando se produce una trampa, el MMV toma el control,
  - revisa si una aplicación o un sistema operativo huésped generaron la trampa
  - y ejecuta el procedimiento correspondiente

Hipervisores Tipo2

- Traducción Binaria
  - El hipervisor revisa el conjunto de instrucciones de un programa
    - Las instrucciones sensibles se sustituyen por llamadas a procedimientos del hipervisor que emulan estas instrucciones
    - Se ejecuta el bloque de instrucciones modificado

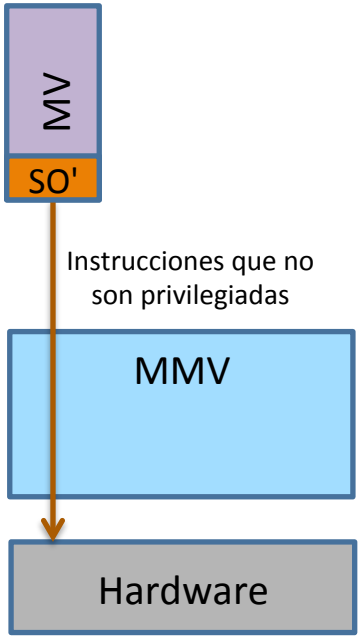
Arquitecturas

- Virtualización completa
  - El hardware virtual es funcionalmente idéntico al hardware real
    - Se crea un nivel de virtualización del hardware que se encarga de las operaciones sobre el hardware real
    - El sistema operacional huésped se desacopla del hardware y por tanto no requiere modificaciones
    - El nivel de virtualización introduce carga adicional (time execution overhead) por la traducción de los pedidos entre el hardware y el software
  - Desventaja
    - En algunos casos es deseable que el SO vea recursos reales tanto como virtuales (soportar tareas sensitivas al tiempo)



Arquitecturas

- Paravirtualización
  - Presenta una abstracción de la máquina virtual que es similar, pero no idéntica, al hardware real
    - No existe el nivel de virtualización del hardware
    - Requiere cambios en los sistemas operacionales huésped para responder a la ejecución sobre un hipervisor
    - Ofrece mejor desempeño



Paravirtualización en Xen

- Memoria virtual
  - Los SO huéspedes son responsables por la creación y administración de las tablas de páginas con intervención mínima del hipervisor
    - Cuando un OS huésped necesita una nueva tabla, este separa espacio en su propio espacio de memoria y cede los privilegios de escritura directa al hipervisor
    - Las actualizaciones siguientes son verificadas por el hipervisor

Comparación

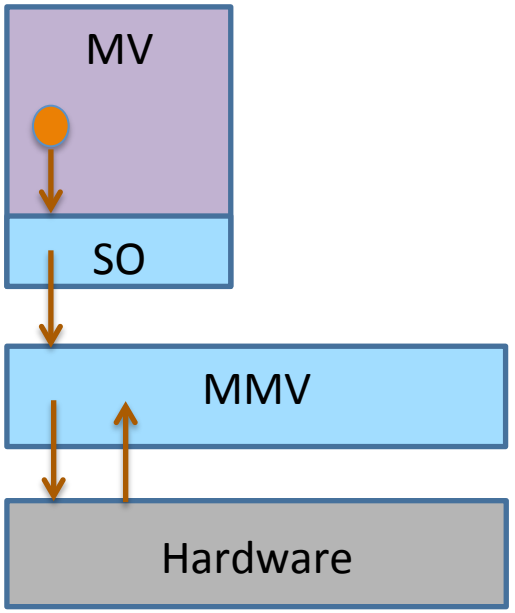
Arquitecturas	Ventajas	Desventajas
Virtualización completa	Los SO guests no necesitan modificaciones	Costo en tiempo de desempeño por el manejo de las instrucciones
Paravirtualización	Mejor desempeño	Los SO guests requieren pequeñas modificaciones

Aspectos a Tener en Cuenta

- Memoria virtual
- Modo de ejecución
- Hipervisor tipo1 vs hipervisor tipo2
- Arquitectura
- Eficiencia

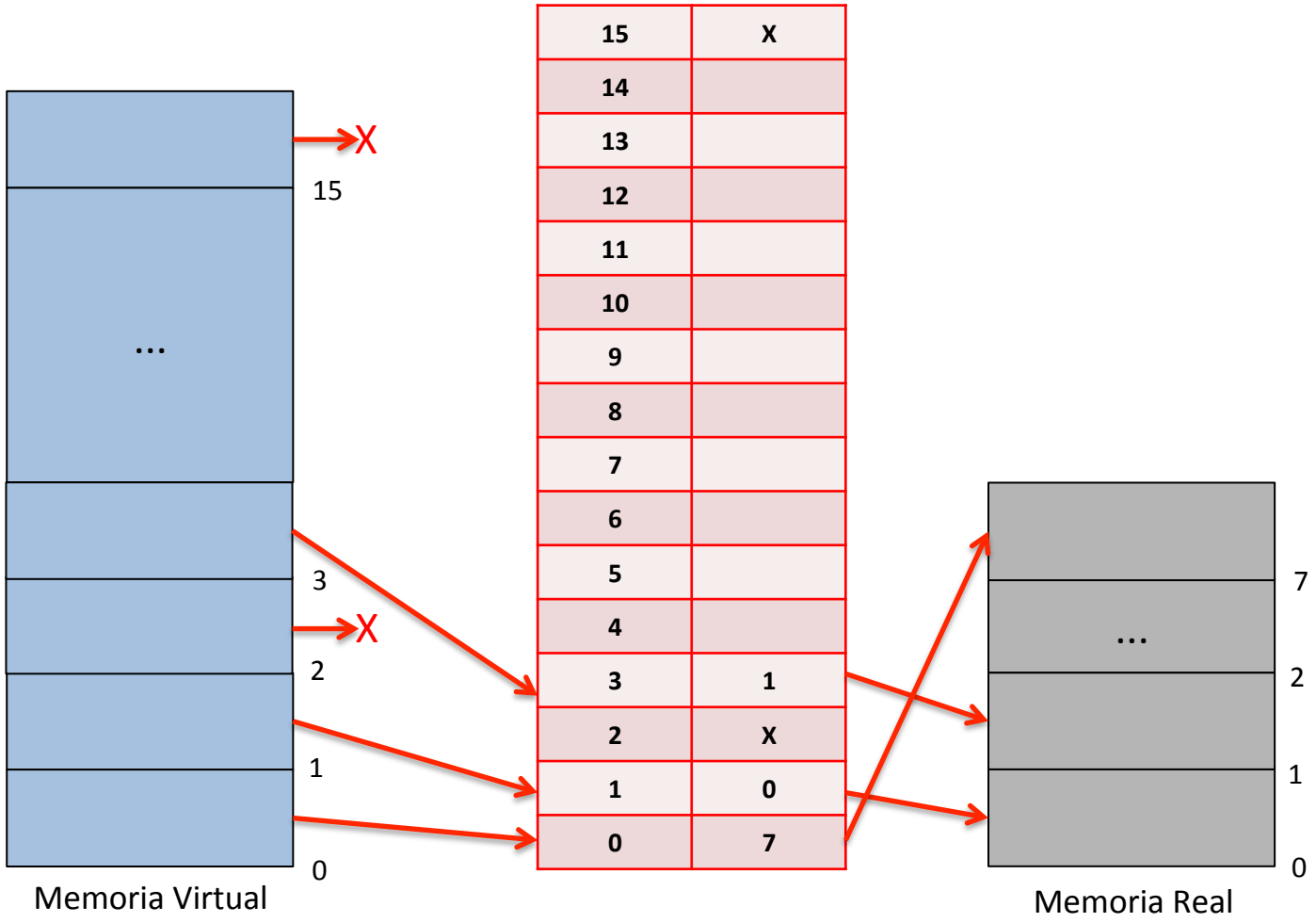
Memoria Virtual

- Las máquinas virtuales no usan directamente la memoria de la máquina
  - El MMV tiene el control
  - El sistema operacional de la MV (SO tradicional) cree que está administrando los recursos de la máquina



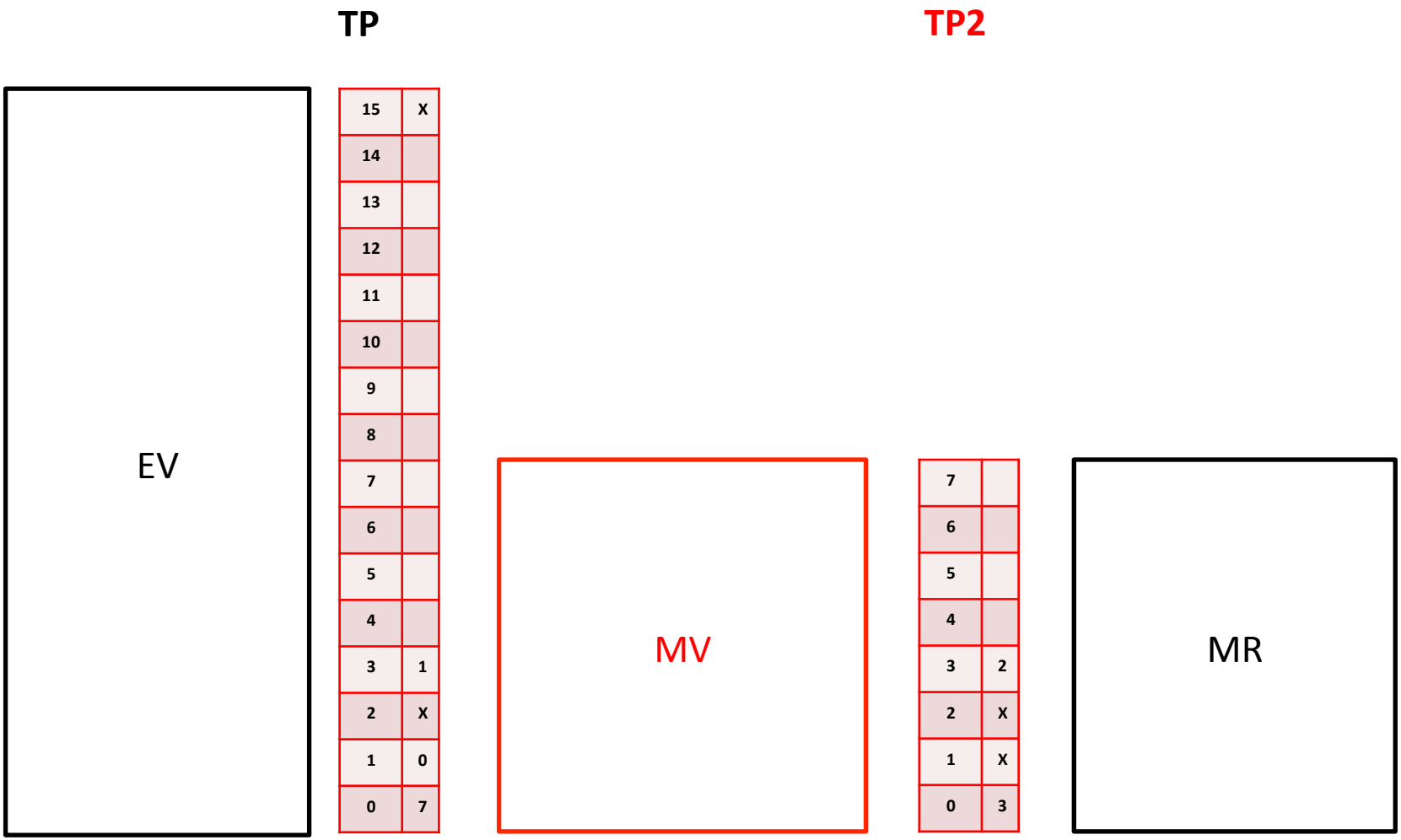
El MMV debe resolver el problema

Implementación

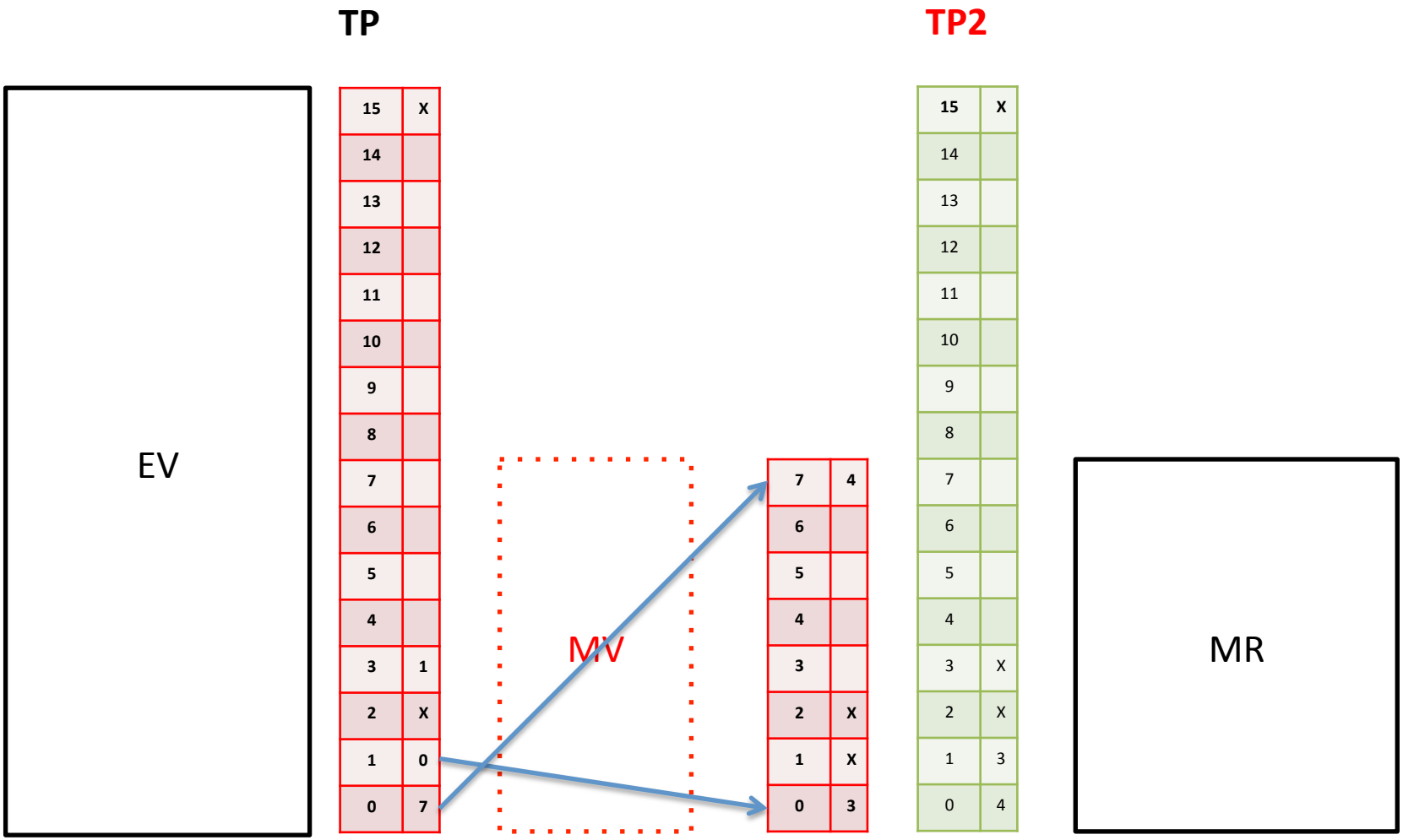


Implementación

MMV

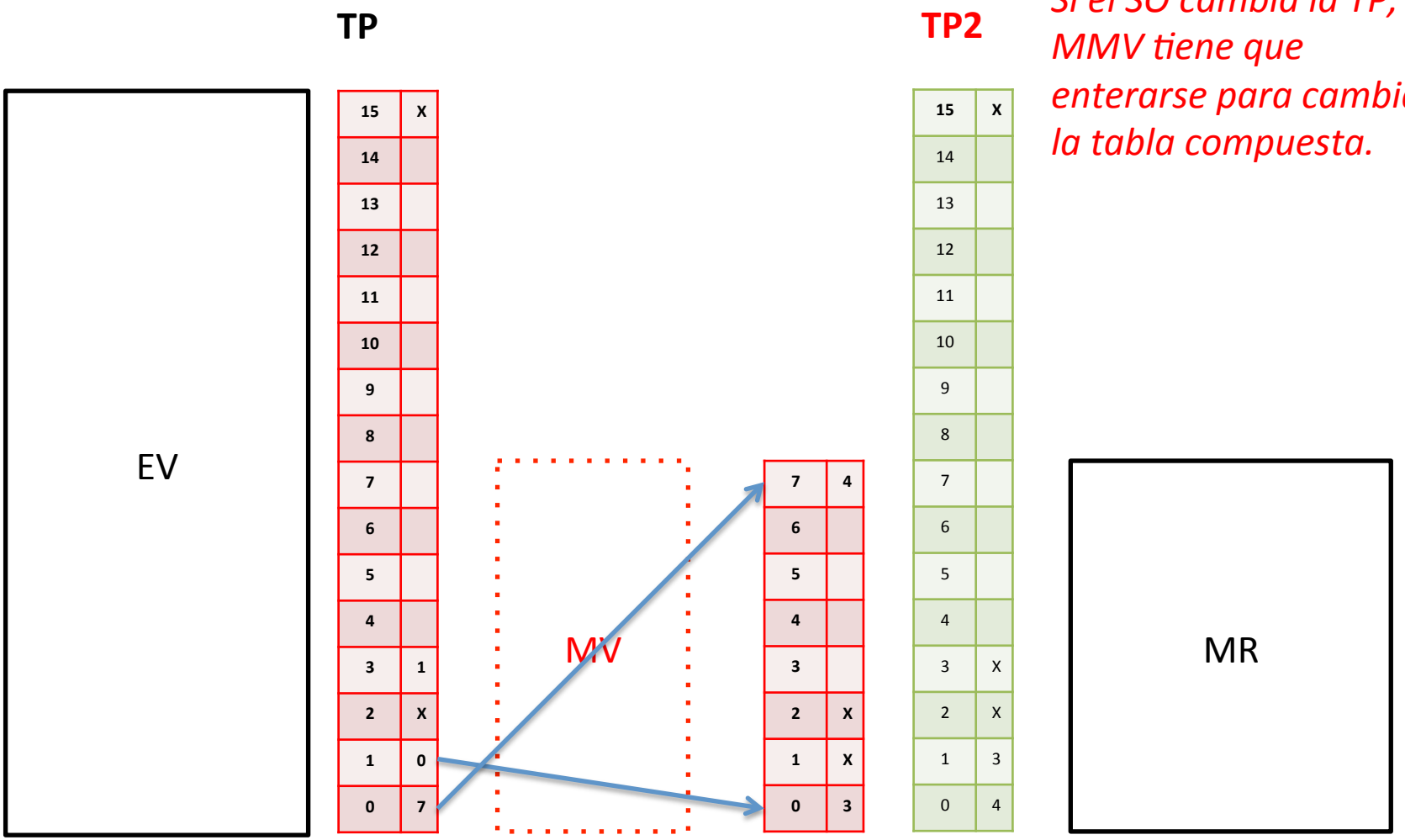


Implementación



Implementación

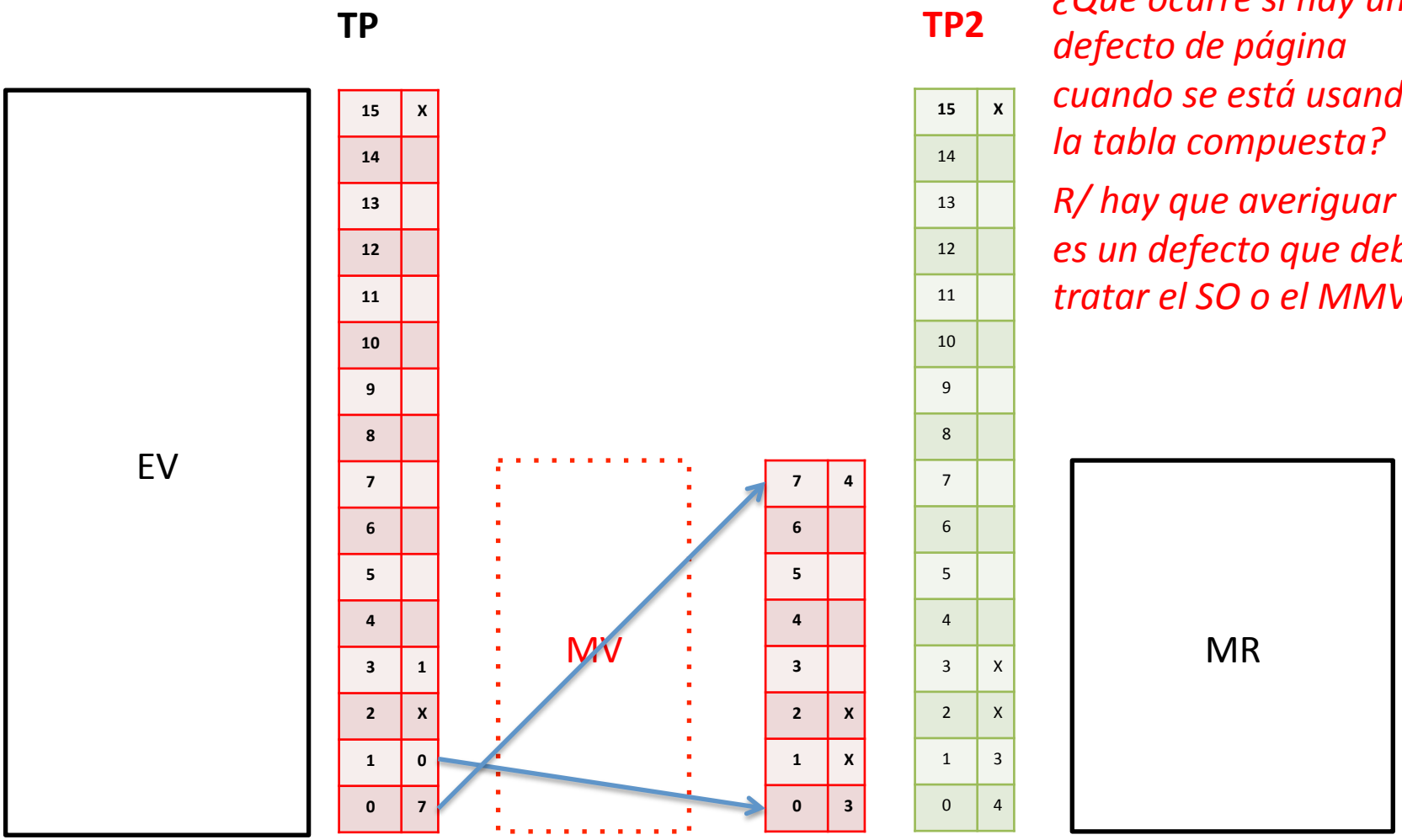
MMV



Si el SO cambia la TP, el MMV tiene que enterarse para cambiar la tabla compuesta.

Implementación

MMV

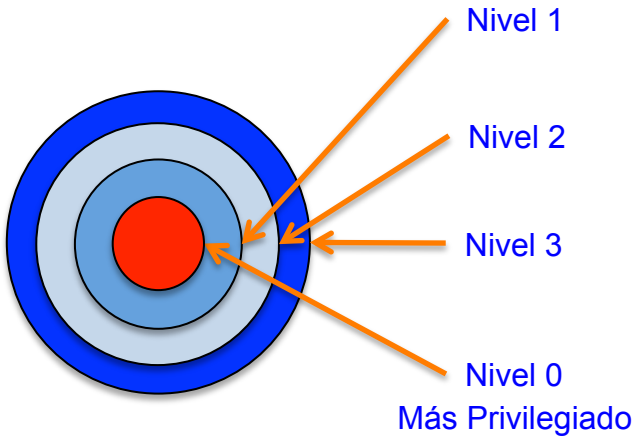


¿Qué ocurre si hay un defecto de página cuando se está usando la tabla compuesta?

R/ hay que averiguar si es un defecto que debe tratar el SO o el MMV

Modo de Ejecución

- CPU
  - El SO ya no es la entidad más privilegiada en el sistema
  - El hipervisor es la entidad más privilegiada
    - La arquitectura x86 maneja 4 anillos de privilegio
      - Los SO tradicionales pueden ser portados para correr en el anillo 1 y el MMV en el anillo 0
      - Las instrucciones privilegiadas ejecutadas por los sistemas operativos van al hipervisor y allí son manejadas



Hipervisor

- Tipo1 vs Tipo2
  - Una implementación tipo 1 (bare metal)
    - implementa sus propios mecanismos
    - tiene control total sobre el hardware de la máquina
  - Una implementación tipo 2 (hosted)
    - cuenta con los servicios del sistema operacional host
    - pero el MMV ***NO*** tiene control total sobre las decisiones del sistema operacional host

Arquitectura

- No todas las instrucciones sensitivas son provilegiadas
  - ¿Qué ocurre en un ambiente con máquinas virtuales cuando una aplicación ejecuta una instrucción privilegiada?
  - ¿Qué ocurre cuando el SO de una MV intenta ejecutar una instrucción privilegiada?

Caso del Pentium de Intel

- Estudio de la Escuela de Posgrados de la Escuela Naval de los Estados Unidos
  - 17 instrucciones del conjunto de instrucciones del Pentium son sensitivas, pero no son privilegiadas
    - El MMV no puede controlar el efecto de estas instrucciones
    - Esta arquitectura no es virtualizable

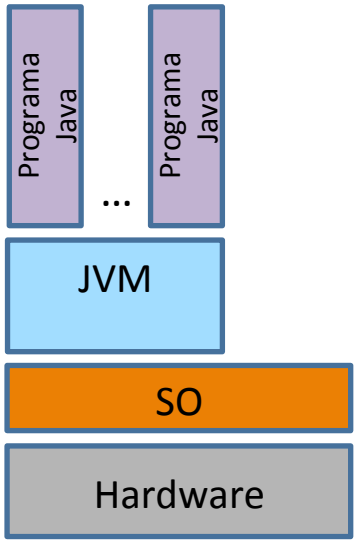
[Robin & Irvine, 2000]

Eficiencia

- Para mejorar la eficiencia la mayoría de las instrucciones se ejecutan directamente en la máquina
  - Un buen porcentaje de las instrucciones no modifican el estado de la máquina
  - Las instrucciones privilegiada generan una interrupción y el MMV toma el control
  - Paravirtualización

Virtualización en Software

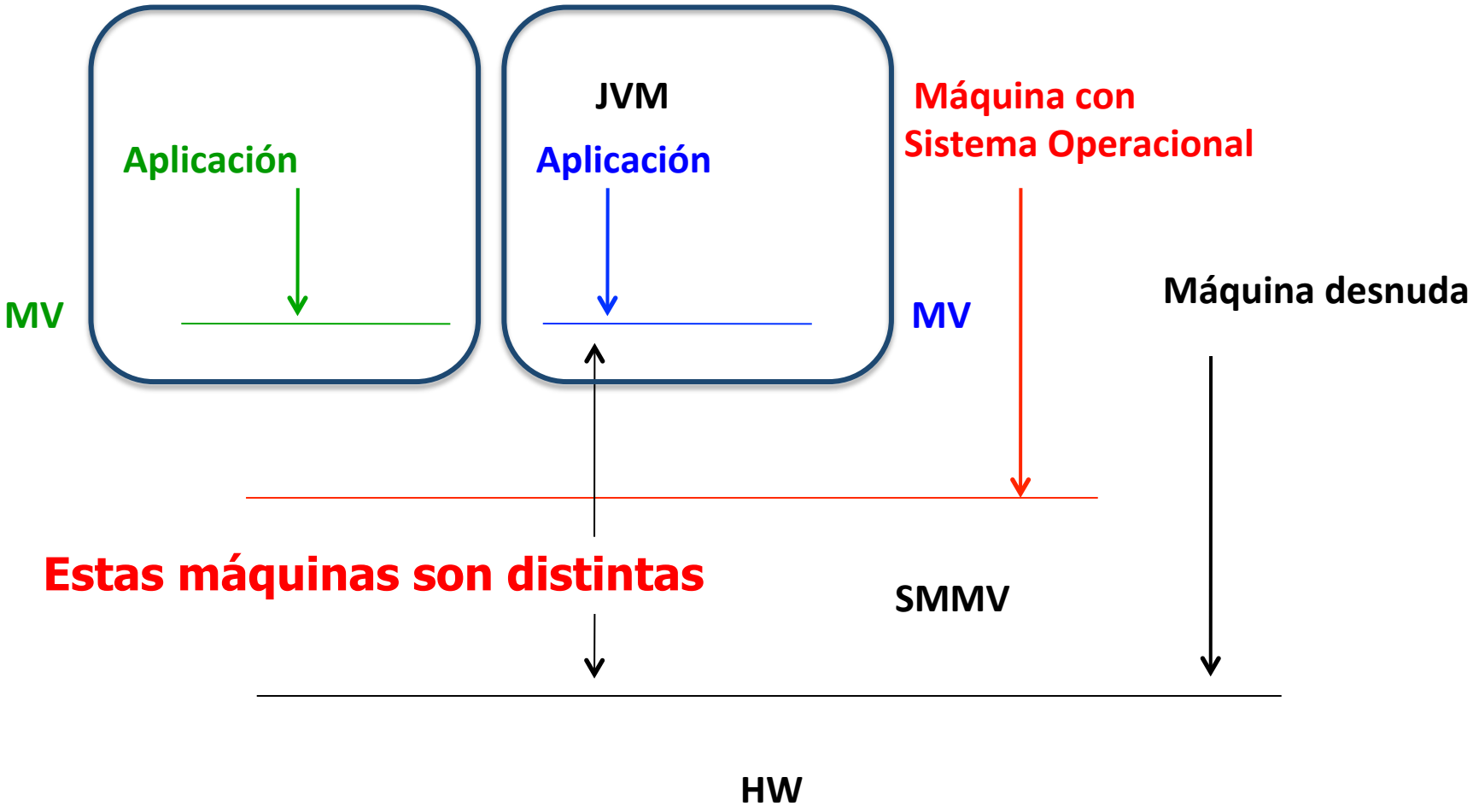
- Presenta una abstracción de la máquina que no se relaciona con el hardware real
  - Todas las instrucciones deben ser interpretadas
    - Ejemplos
      - JVM
      - Emuladores de Intel sobre Motorola



JVM

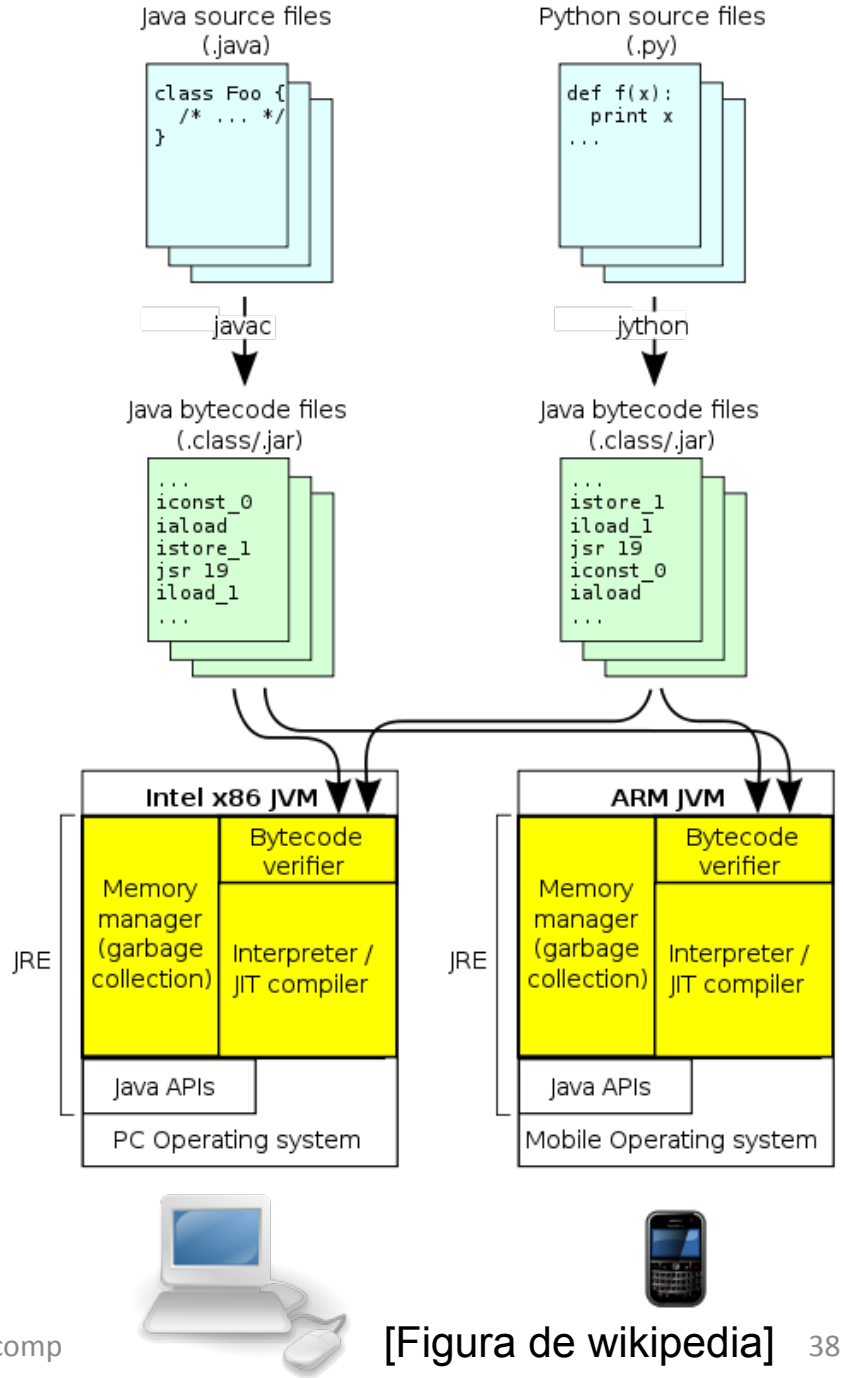
- ¿Qué diferencias hay entre
  - una máquina virtual con virtualización total o paravirtualización y
  - una máquina virtual Java?

JVM


Aplicación
MV
JVM
Aplicación
MV
Máquina con
Sistema Operacional
Máquina desnuda
Estas máquinas son distintas
SMMV
HW

JVM

- Las máquinas virtuales de Java corren sobre un SO
  - Todas las instrucciones son interpretadas
  - Un compilador genera bytecode que es ejecutado por el interpretador correspondiente



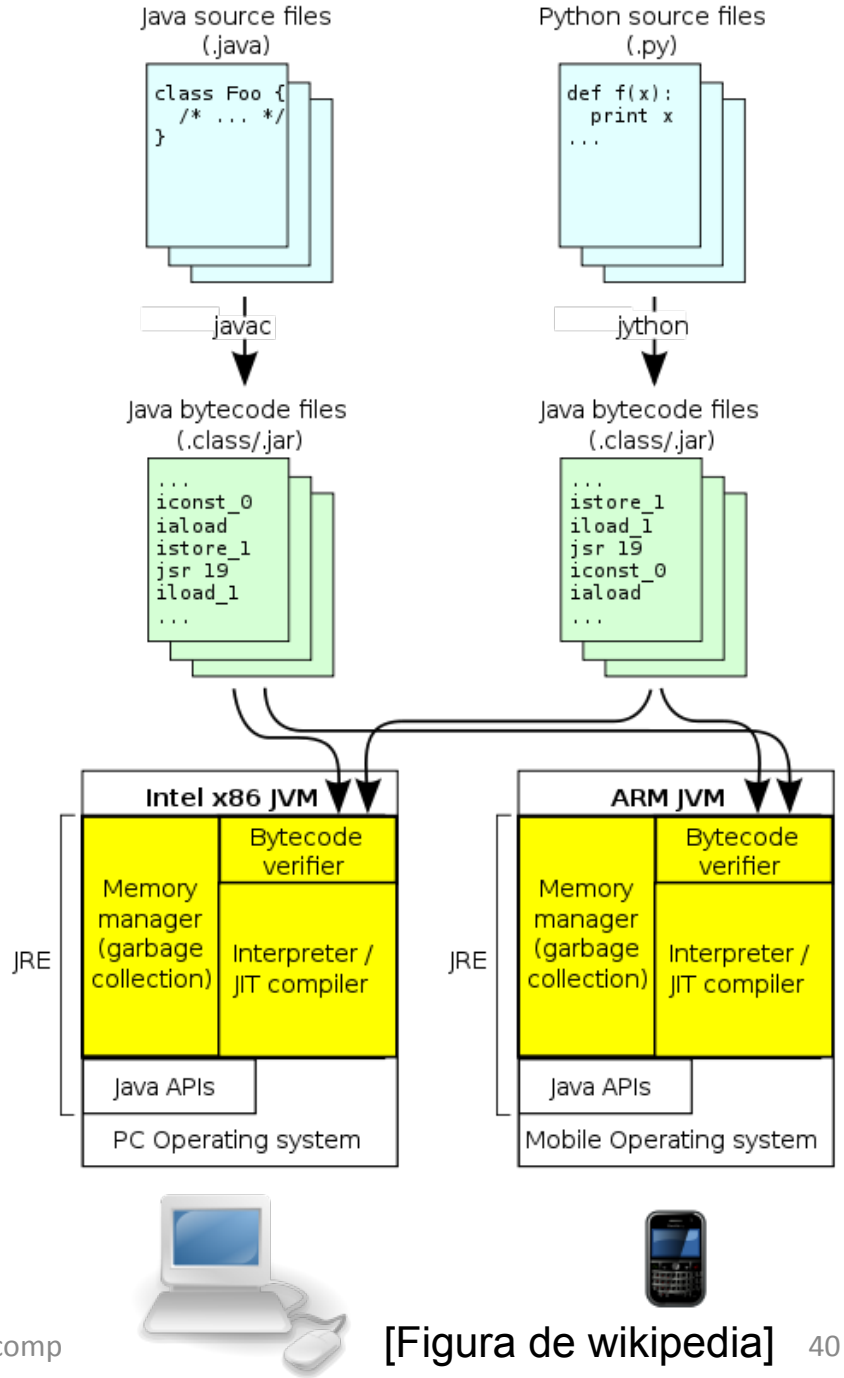
[Figura de wikipedia]

Instrucciones Interpretadas

- Lenguaje compilado vs. lenguaje interpretado
  - Ejecución
  - Desempeño
  - Dependencia de la plataforma

JVM

- Las máquinas virtuales de Java corren sobre un SO
- Pueden ser implementadas para correr directamente en hardware
  - Procesadores Java



[Figura de wikipedia]

JVM

- ¿Quién tiene el control del hardware?
- ¿Quién decide cuándo asignar el procesador?
- ¿Qué ventajas ofrece la JVM?
- ¿Qué ventajas de desempeño ofrece la JVM?

Comparación

Arquitecturas	Ventajas	Desventajas
Virtualización completa	Los SO guests no necesitan modificaciones	Costo en tiempo de desempeño por el manejo de las instrucciones
Paravirtualización	Mejor desempeño	Los SO guests requieren pequeñas modificaciones
Virtualización en software	Una aplicación corre en cualquier plataforma sin procedimientos adicionales	Costo en tiempo de desempeño por el manejo de las instrucciones

Ventajas de las Máquinas Virtuales

- Aislamiento
- Menor requerimiento de máquinas físicas
- Mayor flexibilidad para administradores y usuarios
  - Soporte a aplicaciones heredadas (legacy)
  - Ejecución de diferentes versiones de sistemas operativos
  - Migración de máquinas virtuales

Referencias

- *Fundamentos de Sistemas Operativos*. Silberschatz, Galvin y Gagne,. Ed. McGrawHill, 2006.
- *Requerimientos Formales para Arquitecturas Virtualizables*. G. Popek y R. Goldberg. Communications of the ACM, 1974.
- *Xen and the Art of Virtualization*. P. Barham, B. Dragovic, K. Fraser, S. hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield. SOSP 2003.
- *Analysis of the Intel Pentium's Ability to Support Secure Virtual Machine Monitor*. J.S. Robin, C. Irvine. USENIX 2000.