Clase 5-virtualización

miércoles, 16 de abril de 2025 18:08

QUé es? Realziar abstracción de recursos de la computadora. Una capa abstracta que desacopla el hardware físico del sistema.
Esconde detalles técnicos mediante la encapsulación
Permite que una compu haga el trabajo de varias mediante compartir recursos de un único dispositivo de hardware

Antes de la virtualización desaprovechabas el hardware mucho más

Hoy tenés un servidor grande con varias máquinas virtuales. Lo que se busca es que sean idependientes. Si falla una no falla el resto

Por qué se usa?

- Si tengo que correr algo no seguro no cago el resto
- Tiene un entorno limitado
- Permite simular redes de computadoras independientes
- Para correr varios so en simultáneo
- Etc etc etc

Cada vm tiene su conjunto de hardware virtua sobre el que se ejecuta el so guest. El guest ve solo lo que se le da, no el hw real.

Las vms están representadas en archivos dentro de un file system.:

- Es fácil de almacenar, copiar, hacer backup, restaurar
- Fácil agregar recursos

Componentes:
Software host: quien simula
Guest: lo que se quiere simular

VMM: monitor de máquinas virtuaees
Hypervisor: programa que se corre sobre el hardware para implementar las vm
Se ejeuta en modo kernel
El so guest se ejecuta en modo usuario. Incluso el modo kernel dentro del so se ejecuta en modo usuario
Si en el guest hay algo que se quiere ejecutar en modo kernel, se hace una trap y lo atiede el hipervisor

Instrucciones privilegiadas:
Cuando se virtualiza, el guest emite una instrucción privilegiada en modo usaurio que no puede ser ignorada, se genera ua trap al so
Instrucciones sensibles: modifican registros del procesador pero algunas se ejecutan en modo usuario. EN la diapo dice full modo kernel
Instrucciones privilegiadas: interrupción al ejecutarse en modo usuario
No privilegiadas: modo normal xd

Anillos de privilegios en x86:
Cuatro anillos, el 0 es el del control total. Kernel corre ahí
El nivel de privilegios dice si una instrucción privilegiada que controla funciones básicas de la cpu se pueden ejecutar sin generar ua excepción

- En Ring 0 se ejecutan directamente
- En otros Rings se invocan a través de una excepción

Se busca que los Hypervisores se ejecuten en el Ring 0 (o lo mas cercano) y los SO guest se ejecuten en el 1 o 3 (x86)

Trap and emulate:

- Funciona similar a la emulación pero realiza una interpretación selectiva
- Aplicaciones y sistema operativo ejecutan en modo usuario
- Aplicaciones ejecutan nativamente en el hardware
- VMM ejecuta en modo privilegiado
- Cuando se ejecuta una instrucción privilegiada en el guest SO (en modo usuario) se produce un "trap" al VMM
- VMM ejecuta las instrucciones necesarias y retorna el control el guest SO
- No puede ser utilizado en todas las ISAs. Debe cumplir con el teorema de Popek and Goldberg
- x86 no cumple con el teorema: no todas las instrucciones sensitivas son privilegiadas (por ej. popf)

Trap an emulate no servía para x86, y como era el más usado fue como upa capaz la virtualización se nos caga

Hypervisores:

- Gen 1: basada en software
- Gen 2: paravirtualización. SO sabe que está siendo virtualizado

Hypervisor tipo 1:
Se instala sobre el hardware directamente, tiene el control total del hardware.
No tiene un so host.
Se VMW ejecuta en modo kernel, ring 0
Cada vm se ejecuta como un prceso de usuario en ring 3
SO guest no necesita ser modificado, no se entenra que está siendo virtualizado.
CUando se hace una operación sensible se hace un trap y lo procesa el hypervisor.
Hypervisor tipo 2:
Sobre el hardware tengo el so original y arriba le meto el guest.
Se ejecuta como un programa de usuario sobre el so host.
Interpreta un sbconjunto de las instrucciones de hardware de la máquina sobre la que corre
Hay que emular el hardware qe se mapea a los so guest

No entendí la pag 32 :c

- **Tipo 1**
- Se ejecuta sobre el HW
- Hypervisor se ejecuta en modo kernel real
- SO guest en modo kernel "virtual" (pero es modo usuario)

- **Tipo 2**
- Se ejecuta como un programa de usuario sobre un SO host.
- Arriba de él, están los SO guests.
- Interpreta un conjunto de instrucciones de máquina.
- El SO host es quién se ejecuta sobre el HW

Emulación:
Softare que permite ejecutar software pensado para un procesador sobre uno de otro tipo

Toda instrucción que emite la aplicación se captura por el emulador captura en las que necesite para ejecutar en el hw físico real
TODO pasa por el emulador

FUll vitualización:
Es particionar un proce físico en varios contextos, cada uno corre sobre el mismo proce.
Es la que más se usa
No se da cuenta que está siendo virtualizado
Los bloques con interrupciones sensibles se modifican x el VMM
Los bloques chill se ejecutan directamente en el hardware
Solución a lo del x86: cuando encontraban una instrucción sensible se modificaba para qe spi. Combinaba tradución binaria y ejecución directa deja cacheada la traducción así se ahorra volver a buscarla.

Intel: tiene dos modos: modo root y no root xd
Crea a la par otros anillos aparte en los que corre el so guest. Estos rings están en el modo 0 del so original. Igual no tiene acceso al hardware, tiene que pedir acceso al hypervisor para cositas sensibles (modo no root rootn't)

Paravirtualización:
Se modifica so guest para uqe trabaje directamente con VMM para mejorar el rendimiento
Se agrega software al guest. No hay traducción binaria
Hypercall: so guest en vez de invocar instrucciones sensibles llama a una api del hypervisor que so llamadas a procedimientos.
Paravirtualización es cuandos e eliminan instrucciones sensibles. Si no se eliminan todas se lama paravirtualización y vmm debe hacer traducción de las que no estén.
Tecnicas:

- Recompilando el kernel del sistema guest:
  - Driver y forma de llamar a la api están en el kernel
  - Hay que meter un so modificado/especifico
- Instalando drivers paravirtuaizados:
  - Paravirtualización parical.
  - Es la que se usa hoy

Virtualiación completa puede generar problemas de performance al tener que emular todo el hardware.
Paravirtualización completa tiene mejor performance pero soporta pocos so porque lo tiene que modificar.
Paravirtualización en drivers es un punto medio