Virtualización, contenedores y cloud

Rodolfo Baader[1]

[1]Departamento de Computación, FCEyN,
Universidad de Buenos Aires, Buenos Aires, Argentina

Sistemas Operativos, primer cuatrimestre de 2023

(2) Virtualización

- Definición: es la posibilidad de que un conjunto de recursos físicos se vean como varias copias de recursos lógicos.
- La acepción más común es pensar en una computadora realizando el trabajo de varias.

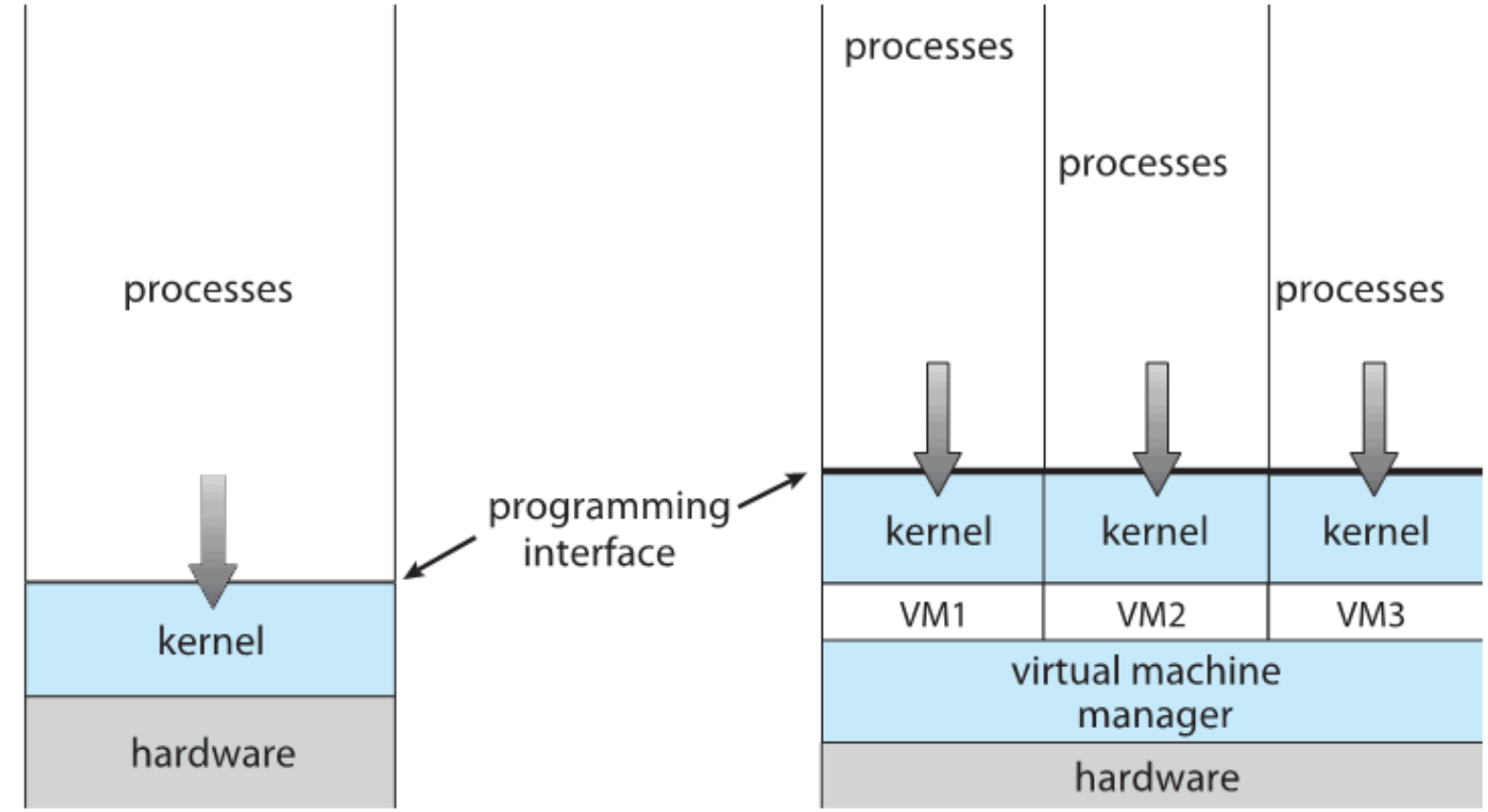
(3) Algo de historia

- Desde hace rato (por lo menos 1960) resulta tentadora la idea de tener *máquinas virtuales*.
- Ie, de mentira.
- Los objetivos son variados:
  - Portabilidad (à la Java Virtual Machine).
  - Simulación/testing.
  - Aislamiento (como chroot, jail y cía).
  - Particionamiento de HW.
  - Agrupamiento de funciones ("*consolidation*").
  - Protección ante fallas de HW (migración de HW).
  - Migración entre HW sin pérdida de servicio.

(4) Virtualización

- Concepto de VMM/Hypervisor
- Características esenciales de los VMM (Popek y Goldberg, 1974)
- Fidelidad
- Performance
- Safety

(5) Virtualización



Non-virtual machine

Virtual machine

(6) Simulación y emulación

- Una forma posible de lograr esto es mediante la *simulación*:
  - En el sistema *anfitrión* se construye una variable de estado artificial que representa al sistema *huésped*.
  - Se lee cada instrucción y se modifica el estado como si ésta se ejecutase realmente.
- Sin embargo:
  - El mecanismo puede ser muy lento.
  - ¿Cómo se simulan las interrupciones, DMA, concurrencia, etc.?

(7) Simulación y emulación (cont.)

- Otra forma es mediante la *emulación de HW*:
  - Acá el sistema emulado se ejecuta realmente en la CPU del anfitrión.
  - Se emulan componentes de HW.
  - Ie, cuando la máquina virtual cree que está haciendo E/S de un dispositivo, en realidad lo está haciendo contra el controlador de máquina virtuales.
  - Éste, a su vez, hace de proxy contra el dispositivo real o la emulación que se esté usando.
  - El grueso del código se corre mediante *traducción binaria*.
- Problemas:
  - ¿Cómo logro separación de privilegios? Toda la máquina virtual corre en modo usuario.
  - ¿Qué pasa con la velocidad de acceso a los dispositivos?

(8) Virtualización asistida por HW

- En este contexto nace la intención de lograr virtualización asistida por HW, especialmente para lograr evitar los siguientes problemas (retengo los nombres de Intel):
  - Ring aliasing: tengo programas escritos para modo kernel, pero en realidad se están ejecutando en modo usuario. Puedo tener problemas de permisos para ejecutar ciertas instrucciones.
  - Address-space compression: ¿cómo hago para que la máquina virtual no pueda pisar memoria del propio emulador? Recordemos que desde el punto de vista del anfitrión son un único proceso.
  - Non-faulting access to privileged state: algunas instrucciones privilegiadas generan un trap cuando se ejecutan sin permiso. Eso es bueno porque puedo atrapar el trap y simularlas. Pero otras no. ¿Cómo hago?
  - Interrupt virtualization: hay que simularle las interrupciones al SO huésped.

(9) Virtualización asistida por HW (cont.)

- seguimos con los problemas...
  - Access to hidden state: hay parte del estado del procesador que no es consultable por software.
  - Ring compression: como tanto el kernel huésped como sus programas corren en realidad en el mismo nivel de privilegio, no hay protección entre kernel y programas de usuario.
  - Frequent access to privileged resources: si bien el controlador de máquinas virtuales puede bloquear el acceso a ciertos recursos, haciendo que se genere un trap, esto puede ser un cuello de botella para recursos accedidos frecuentemente.

(10) Virtualización asistida por HW (cont.)

- Para solucionar estos problemas los fabricantes agregaron soporte para la virtualización en el HW.
- En el caso de Intel, agregaron al procesador las extensiones *VT-x*, que proveen dos modos:
  - VMX root: Las instrucciones se comportan de manera similar, pero hay algunas extensiones (anfitrión).
  - VMX non-root: El mismo set de instrucciones pero con comportamiento restringido (huésped).
- Se proveen (10) instrucciones para alternar fácilmente entre ambos modos.

(11) Virtualización asistida por HW (cont.)

- Se agrega la *Virtual Machine Control Structure* (en memoria).
  - Campos de control: indican qué interrupciones recibe el huésped, qué puertos de E/S, etc.
  - Estado completo del huésped.
  - Estado completo del anfitrión.
- La idea es que el HW sale automáticamente de modo VMX non-root cuando el huésped realiza alguna acción que está "prohibida" de acuerdo a la VMCS.
- En ese momento, el controlador de la máquina virtual recibe el control y emula, ignora o termina la acción "prohibida".

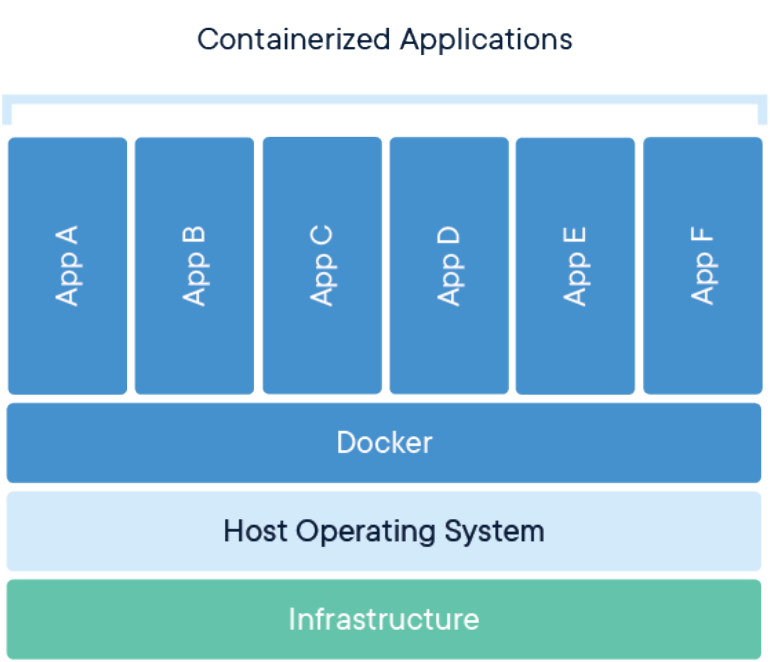
(12) Desafíos/problemas

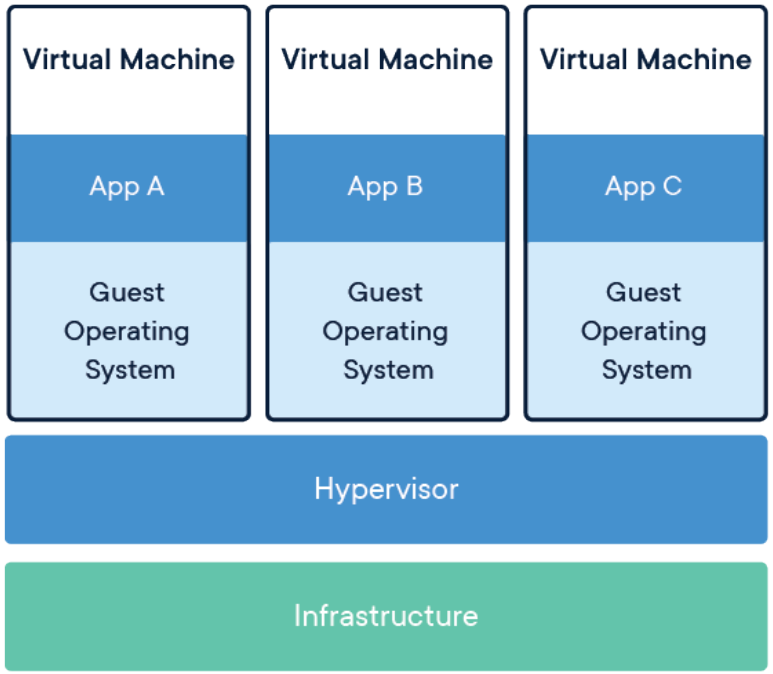
- ¿Qué pasa con las optimizaciones que tenían el kernel y el FS para acceder al disco de manera eficiente?
- ¿Y con picos de carga en más de una CPU?
- Único punto de falla: falla una pieza de HW real, caen varias máquinas virtuales.
- Pero también al revés.

(13) Escenarios de uso de la virtualización

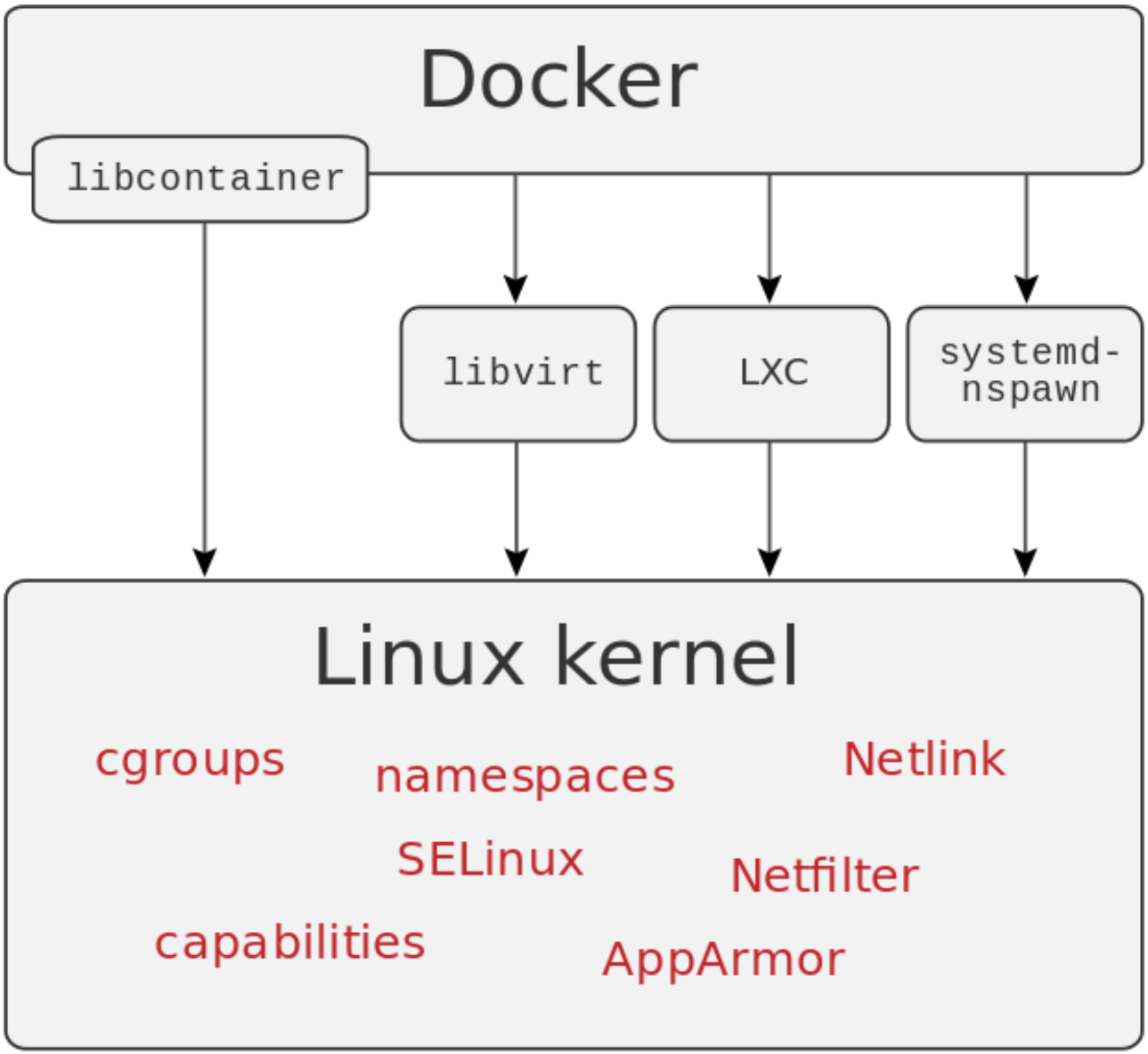
- Correr sistemas viejos.
- Aprovechamiento de equipamiento
- Desarrollo/testing/debugging.
- ¿Abaratar costos?

(14) Contenedores





(15) Contenedores



(16) Tecnologías provistas por el kernel

- Cgroups: Control Groups - limita lo que se puede usar
  - Memoria
  - CPU
  - I/O
  - Red
- Namespaces: Provee a los procesos una visión propia del sistema.

(17) Tecnologías provistas por el kernel - Redes - IPC - privilegios

- Netfilter: Filtrado de paquetes (fw), nateo
- Netlink: comunicación entre el kernel y procesos de usuario
- Capabilities: Permiten asignar a un aplicativo ciertos privilegios usualmente reservados al administrador.

(18) Tecnologías provistas por el kernel - MAC

- SELinux: Más versatil, más complejo de configurar.
- AppArmor: Más sencillo, menos funcionalidad.
- Ambos funcionan utilizando LSM (Linux Security Modules)

(19) Orquestación de aplicaciones contenerizadas

- Kubernetes: Plataforma de código abierto para automatizar la implementación, el escalado y la administración de aplicaciones en contenedores.
- Openshift (y OKD): usa Kubernetes de base, pero agrega restricciones de seguridad por defecto, interface web mas completa, manejo de roles, facilidades para el desarrollador.

(20) Cloud


PR OVIDER
SCALABILITY
UTILITY
INTERCLOUD
USER
VIRTUALIZA TION
RELIABILITY
DATA
COMPUTING
CLIENT
COMMUNITY
INFRASTRUCTURE
PHONE
SaaS
SECURITY
IaaS
APPLICA TION
CONTENT
CLOUD
MOBILE
DEVICE
TABLET
SERVICE
ONLINE
COST
PaaS
PRIV AT E
PLA TFORM
SLA
PUBLIC
SERVER
HYBRID
REMOTE
API
ARCHITECTURE
CENTER
NETWORK
ENGINEERING
BROWSER
WEB
MONIT ORING
DATABASE
ST ORAGE
IDENTITY

(21) Bibliografía

- `http://www.cs.cornell.edu/home/ulfar/ukernel/ukernel.html`
- "Formal Requirements for Virtualizable Third Generation Architectures", de Popek y Goldberg.
- "Intel® Virtualization Technology: Hardware Support for Efficient Processor Virtualization". `http://www.intel.com/technology/itj/2006/v10i3/1-hardware/1-abstract.htm`
- "A comparison of software and hardware techniques for x86 virtualization", de Adams y Agesen.