Contenedores

UN POCO DE HISTORIA

—

Por Gunnar Wolf

Gunnar Wolf es administrador de sistemas para el Instituto de Investigaciones Económicas de la UNAM y desarrollador del proyecto Debian GNU/Linux.
http://gwolf.org

Recuerdo la primera vez que tuve contacto con la virtualización. En 2005, en un congreso, participé en un breve taller en el que nos presentaron esta (entonces) característica tan única de las arquitecturas IBM de gama alta: los servidores s390.

¡Parecía magia! Varios sistemas operativos corriendo a velocidad nativa, compartiendo armónicamente una misma computadora. Un hipervisor gestionando y mostrando estadísticas de uso en tiempo real. Claro, a esas alturas yo ya conocía el mecanismo que empleaba VMWare para virtualizar sobre la arquitectura PC desde fines de los noventa, pero la diferencia en el rendimiento y la confiabilidad era muy notoria. Por esos años, relativamente muy poca gente consideraba la oferta de VMWare para entornos de producción.

Un año más tarde, el panorama cambió radicalmente. Tanto Intel como AMD comenzaron a incorporar extensiones de virtualización en sus procesadores de arquitectura x86 —las instrucciones y niveles de protección necesarios para permitir este mágico aislamiento que hasta entonces estaba reservado a los sistemas de muy alto costo. Si bien la virtualización apareció primero en la gama alta de estas compañías, hoy en día (diez años más tarde) prácticamente todos sus procesadores las incluyen. Y sí, cambiaron fuertemente la forma de trabajo en los centros de datos, las ofertas de servicio de los proveedores de infraestructura— y permitieron el nacimiento del, permítanme esta redundante cacofonía, nebuloso término de "la nube".

Por un par de años, hubo un claro surgimiento de la "virtualización basada en hardware". Proveedores de servicio de todo tamaño comenzaron a ofrecer "servidores privados virtuales" a precios que antes nunca hubiéramos esperado. Los administradores de sistemas de todo el mundo comenzamos a cambiar nuestra mecánica de trabajo, "particionando" el trabajo que realizaban nuestros grandes servidores monolíticos en cargas especializadas, a ser repartidas en cuantos servidores virtuales fuera necesario. No tardaron en aparecer numerosas tecnologías base, libres y propietarias.

DE VIRTUALIZACIÓN A PARAVIRTUALIZACIÓN

Quedaba aún mucho por mejorar, sin embargo, si bien la ejecución de código dentro de las máquinas virtuales ocurría ya a velocidad nativa, había inocultables capas de emulación que era necesario cruzar. ¿A qué me refiero? Una máquina virtual aparenta ser una computadora completa, al lanzar una, vemos cómo una interfaz comparable con el BIOS lanza al gestor de arranque, el sistema operativo se inicializa... la máquina virtual cree tener las tarjetas de red, unidades de disco y demás periféricos que le indiquemos (que normalmente corresponden con modelos simples y antiguos). El manejo del video es particularmente curioso; lo más común es que se presente como una tarjeta de video tonta, sin aceleración 3D, que interfacea mediante un framebuffer. Todos estos dispositivos tienen que ser emulados, y toda emulación tiene su costo.

Hoy en día, prácticamente la totalidad de herramientas de virtualización ofrece más bien un híbrido, entrando al terreno de la paravirtualización. ¿Qué es? Es cuando el engaño no es completo, cuando el hardware virtual ofrecido al sistema operativo ya no pretende ser una computadora completa, sino que el sistema operativo huésped "sabe" que está siendo virtualizado y coopera en la tarea. Esta cooperación estiba en que, en vez de buscar controlar a su hardware como si fuera real, lo toma como un buffer glorificado y envía, más bien, solicitudes al anfitrión (host) para cumplir lo que los programas requieren. Una diferencia aparentemente sutil, pero que —sobre todo en los casos de cargas de trabajo orientadas a la entrada y salida, como las bases de datos y el manejo de red— típicamente llevan a una diferencia de rendimiento de diez veces tanto.

Virtualización y paravirtualización brindan grandísimas ventajas a los administradores, y una en la que insistiré en este punto es que el "engaño" es a nivel sistema entero, cada computadora virtual puede ejecutar un sistema operativo distinto completo. Esto es particularmente útil al hablar de proveedores de servicio en la nube, pues cada cliente tendrá la flexibilidad absoluta de correr en su computadora el software que desee. Sin embargo, hacia adentro de las organizaciones, sigue siendo subóptimo.

RAÍCES, JAULAS, CONTENEDORES

Como siempre, la respuesta a una nueva necesidad viene del pasado, con nuevas maneras de emplear lo ya conocido, de refinamientos técnicos que mejoran

la implementación de la idea —y en este caso, de la mano de bastante mercadotecnia que llevó a mucha gente a ver hacia el lugar obvio.

Los contenedores pueden definirse como virtualización a nivel sistema operativo. Los usuarios de máquinas virtuales ya no serán engañados con lo que parece ser una computadora completa, sino que únicamente con algo que dice ser un sistema operativo dedicado a ellos. Y, siendo el sistema operativo el principal encargado de agrupar al hardware en clases, gestionar su manipulación de bajo nivel, y proveer abstracciones uniformes para los programas que corren en espacio de usuario…la diferencia es enorme.

Pero vamos por partes, ¿cómo se originaron los contenedores?, ¿cuáles son las ideas primigenias?

En 1982, Bill Joy, uno de los arquitectos del Unix de BSD y fundador de Sun Microsystems, creó la llamada al sistema chroot() para auxiliarse en el desarrollo de Unix. Dado que desarrollaba en la misma computadora donde hacía sus pruebas, quería poder mantener aislado su entorno de desarrollo y construcción del entorno de pruebas. La manera más sencilla (y, además, elegante) de hacerlo es por medio de esta llamada, se invoca con un argumento, y a partir del momento en que un proceso la llama con el nombre de un directorio y hasta el final de su ejecución, el sistema operativo le dará a dicho proceso una vista parcial del sistema, limitada al directorio que le es indicado. Esto es, tras un chroot('/home/gwolf/test'), si el proceso en cuestión da un cd /, el sistema operativo lo llevará a /home/gwolf/test.

En los manuales de sistema es común encontrar advertencias de seguridad — chroot() no está pensado para proveer aislamiento, únicamente conveniencia — Hay varias maneras por medio de las cuales un proceso puede escapar de esta jaula, además de que su funcionamiento se limita a restringir la vista del sistema de archivos, sin proteger otros aspectos del sistema como las interfaces de red, la lista de procesos, o el acceso a dispositivos.

El primer sistema operativo en ofrecer lo que hoy conocemos como contenedores (bajo el nombre de jails) fue la versión 4 de FreeBSD, liberada en el año 2000. Poco después, esta funcionalidad apareció en Solaris, Linux e incluso para Windows; en los dos últimos casos, esto fue inicialmente mediante productos de terceros (Parallels Virtuozzo Containers para Windows, Vserver y OpenVZ para Linux, OpenVZ siendo una rama libre de Virtuozzo). Varios años después de la introducción de esta funcionalidad por medio de terceros, tanto en el caso de Linux (con lxc desde 2009) como recientemente en el de Windows (desde 2015) han sido incorporados a la plataforma base del sistema operativo.

La virtualización a nivel sistema operativo implica que los contenedores engañarán a los procesos huéspedes presentándoles una vista limitada en los siguientes aspectos:

- Tablas de procesos
- Señales y comunicación entre procesos (IPC)
- Interfaces de red
- Dispositivos de hardware
- Límites en el consumo de recursos (espacio en disco, memoria, o ciclos de CPU)
- Información del sistema, como el nombre del equipo o la hora actual

Esto permite a los árboles de procesos que habitan dentro de un contenedor tener una vista de sistema completo, viéndose obligados a respetar las asignaciones de recursos de los demás contenedores. Uno de los principales atractivos para los administradores de sistemas es que un contenedor cuyos servicios no están siendo activamente utilizados permanece efectivamente dormido, con un nivel de consumo de recursos verdaderamente mínimo — A diferencia de la virtualización de sistema entero, en este caso el sistema operativo conoce directamente todos los eventos a que están esperando cada uno de los procesos, y mientras dicho evento no ocurra, los contenedores no pasarán de ser una estructura en memoria.

A últimos años, los contenedores se han ido popularizando ya no únicamente como una herramienta para que el administrador de sistemas particione lógicamente la carga de trabajo de servidores o provea infraestructura de cómputo con acceso privilegiado a distintos usuarios manteniendo garantías de seguridad, sino también como mecanismo de distribución de software, instalado y preconfigurado —la implementación más conocida es el proyecto docker. En lo particular, como administrador de sistemas de vieja guardia y formado en la escuela tradicionalista, le veo importantes carencias a su planteamiento…pero sin duda ha logrado una importante penetración de mercado.

Como sea, la tecnología de contenedores es una genial herramienta con la que contamos hoy, que ha cambiado de raíz la manera en que se despliegan los servicios. Invito a todos los lectores que no se hayan aún adentrado a comprenderla y utilizarla, a no dudar en hacerlo.