2.- Tipos de virtualización

En el campo de la computación en la nube, la virtualización desempeña un papel fundamental al permitir la creación de entornos virtuales que aprovechan eficientemente los recursos de hardware. Existen diferentes tipos de virtualización, y en esta sección vamos a explorar tres de ellos: Emulación, virtualización y paravirtualización. A continuación, exploraremos estos tipos de virtualización y proporcionaremos ejemplos de cada uno:

- **Emulación:** La emulación es un enfoque de virtualización que permite ejecutar un sistema operativo o una aplicación en un entorno que simula el hardware original. En este caso, el hardware virtualizado es diferente al hardware físico subyacente. Un ejemplo común de emulación es el software de emulación de videojuegos arcade antiguos, donde se recrean las características de una consola de juegos con hardware de los año 80.
  La emulación proporciona una gran flexibilidad, ya que permite ejecutar sistemas operativos y aplicaciones diseñadas para un hardware específico en un hardware diferente. Sin embargo, debido a la naturaleza de la emulación, es menos eficiente en términos de rendimiento que otros enfoques de virtualización, ya que hay que virtualizar por completo el hardware emulado.

- **Virtualización tradicional:** La virtualización tradicional, también conocida como virtualización completa, se basa en la creación de máquinas virtuales (VM) que ejecutan sistemas operativos y aplicaciones en un hardware virtualizado e independiente unas de otras. En este caso, cada máquina virtual tiene que tener su propio sistema operativo independiente del SO anfitrión. Ejemplos populares de software de virtualización tradicional son VMware, Microsoft Hyper-V, VirtualBox, etc. Las máquinas virtuales creadas necesitan ser ejecutadas sobre un software especial que las interprete y gestione, llamado hipervisor, también conocido como VMM (Virtual Machine Monitor), que es el que se encarga de realizar las peticiones hardware al sistema operativo anfitrión o emular el hardware que no esté disponible en la máquina real. Su función principal es dividir los recursos físicos de la máquina anfitrión en múltiples entornos virtuales.
  La virtualización tradicional permite ejecutar múltiples sistemas operativos en un único servidor físico y proporciona un alto nivel de aislamiento entre las máquinas virtuales. Cada máquina virtual se comporta como si estuviera ejecutándose en su propio hardware dedicado. Esto facilita la consolidación de servidores y la gestión de infraestructuras complejas.
  Existen dos tipos principales de hipervisores (gestores e interpretes de VM):

- **Hipervisor de tipo 1** (o "bare metal"): Este tipo de hipervisor se ejecuta directamente sobre el hardware físico del servidor. No requiere de un sistema operativo anfitrión adicional. Los hipervisores de tipo 1 ofrecen un mayor rendimiento y eficiencia, ya que pueden acceder directamente a los recursos del hardware. Algunos ejemplos populares de hipervisores de tipo 1 son VMware ESXi, Microsoft Hyper-V y Xen.
- **Hipervisor de tipo 2** (o "hosted"): Este tipo de hipervisor se ejecuta como un software dentro de un sistema operativo anfitrión. El sistema operativo anfitrión proporciona los controladores de dispositivo y los servicios básicos, mientras que el hipervisor se ejecuta como una aplicación dentro de él. Los hipervisores de tipo 2 son más fáciles de instalar y configurar, pero pueden tener un rendimiento ligeramente inferior debido a la capa adicional del sistema operativo anfitrión. Algunos ejemplos de hipervisores de tipo 2 son VMware Workstation, VirtualBox, QEMU y Parallels Desktop.
- **Hipervisor híbrido**: Además de estos dos tipos principales, también existe una variante llamada "hipervisor híbrido", que combina características de los hipervisores de tipo 1 y tipo 2. Estos hipervisores se ejecutan directamente sobre el hardware pero también requieren de un sistema operativo anfitrión mínimo para proporcionar servicios adicionales. Un ejemplo de este tipo híbrido puede ser Proxmox, que necesita de una distribución Debian con el kernel de Linux y el hipervisor de tipo 2 de código abierto QEMU, como base para la gestión de las máquinas virtuales, y KVM (Kernel-based Virtual Machine) para la gestión de contenedores LXC.

- **Paravirtualización:** La paravirtualización es un enfoque de virtualización en el que el sistema operativo invitado se modifica para ser consciente de que se está ejecutando en un entorno virtualizado. A diferencia de la virtualización tradicional, donde se emula el hardware, en la paravirtualización se utilizan interfaces especiales para acceder al hardware subyacente de manera más eficiente.
  En la paravirtualización, el sistema operativo invitado y el sistema operativo anfitrión colaboran para mejorar el rendimiento y la eficiencia general. Esto permite una comunicación más rápida y directa entre el sistema operativo invitado y el hardware real, lo que resulta en un mejor rendimiento en comparación con la virtualización tradicional.
  Las máquinas paravirtualizadas comparten el kernel del SO anfitrión y por tanto, no se puede virtualizar máquinas con sistemas operativos incompatibles.
  El hipervisor ofrece un interfaz especial para acceder a los recursos. En ocasiones, es necesario la adaptación del sistema operativo de la máquina virtual. Ofrecen el máximo rendimiento, pero no se pueden usar sistemas operativos sin modificaciones o hardware especifico.

- **Paravirtualización ligera:** También llamada virtualización a nivel de sistema operativo, o virtualización basada en contenedores. Es un método de virtualización en el que, sobre el núcleo del sistema operativo se ejecuta una capa de virtualización que permite que existan múltiples instancias aisladas de espacios de usuario. A cada espacio de usuario aislado lo llamamos contenedor. Por lo tanto, un contenedor es un conjunto de procesos aislado, que se ejecuta en un servidor, y que accede a un sistema de ficheros propio, tiene una configuración red propio y accede a los recursos del host (memoria y CPU). Podemos hacer la siguiente clasificación de contenedores:
  - **Los contenedores hardware o de sistema**.
  - **Los contenedores software o de aplicación**.

2.1.- Contenedores de sistemas (LXC)

La virtualización ligera a nivel de hardware, también conocida como contenedores hardware, es una tecnología que permite ejecutar múltiples instancias aisladas de sistemas operativos en un único servidor físico pero aprovechando el sistema operativo anfitrión para acceder al hardware. Un ejemplo común de esta tecnología es LinuX Containers (LXC) que aprovecha el núcleo de Linux del SO del servidor anfitrión (o host).

> Evidentemente, no podemos crear un contenedor hardware LXC con un núcleo del SO diferente al de Linux.

Los contenedores a nivel de hardware ofrecen una gran flexibilidad y agilidad, ya que se pueden iniciar y detener rápidamente, y ocupan menos recursos en comparación con las máquinas virtuales convencionales. Esto los hace ideales para implementar aplicaciones aisladas en entornos de desarrollo, pruebas o producción. Permite una mayor densidad de utilización de recursos, al compartir eficientemente el hardware del sistema, lo que resulta en un menor consumo energético y reserva de recursos en comparación con la virtualización completa. Además, los contenedores LXC son rápidos de crear, clonar y desplegar, lo que facilita la escalabilidad y la replicación de distintos escenarios.

Son una opción más interesante que los contenedores de aplicaciones, como por ejemplo Docker, cuando tenemos la necesidad de emular un hardware específico pero sin sobrecargar al servidor con la virtualización completa de las tradicionales máquinas virtuales (un ejemplo de máquina

virtual tradicional son las creadas con VirtualBox)

La virtualización LXC se basa en la creación de contenedores ligeros y aislados de los errores producidos en otros LXC. Estos contenedores permiten ejecutar múltiples instancias de sistemas operativos Linux de forma eficiente y compartiendo los recursos del sistema anfitrión, como CPU, memoria y almacenamiento.

Cada contenedor LXC incluye su propio entorno de usuario y su espacio de archivos aislado, lo que permite la ejecución de aplicaciones y servicios de manera independiente dentro del contenedor. Los contenedores pueden comunicarse entre sí y con el sistema operativo anfitrión mediante mecanismos de comunicación definidos, como sockets UNIX o redes virtuales utilizando la pila del protocolo TCP/IP.

El hardware de los contenedores LXC puede ser configurado, cómo ejemplo podemos reservar memoria máxima de RAM para utilizar, número de núcleos o hilos del procesador a utilizar, etc... Es decir, **podemos hacer lo mismo que podías hasta ahora hacer con las máquinas virtuales pero sin tener que emular el hardware por completo.**

> En resumen, los contenedores de sistema **LXC** es una tecnología que permite la virtualización a nivel de sistema operativo en entornos Linux, a través de paravirtualización ligera para compartir los recursos del sistema anfitrión. Esta tecnología **ofrece beneficios en términos de eficiencia de recursos** y escalabilidad en el desarrollo, pruebas y despliegue continuo de aplicaciones.

2.2.- Contenedores de aplicación (Docker)

La virtualización a nivel de aplicación se basa en el uso de contenedores para encapsular una aplicación y sus dependencias en un entorno aislado. Docker es una de las tecnologías más populares que se utiliza en este enfoque pero no la única.

Un contenedor Docker permite empaquetar una aplicación junto con todas las bibliotecas y archivos necesarios para su ejecución en un entorno aislado. Los contenedores Docker son livianos, portátiles y se pueden ejecutar en cualquier sistema operativo compatible con Docker (Linux, Windows sobre emulación del kernel de Linux, Mac, etc).

La virtualización a nivel de aplicación ofrece ventajas en términos de portabilidad, eficiencia y escalabilidad. **Permite a los desarrolladores crear, probar y desplegar aplicaciones de manera más rápida y sencilla, asegurando que la aplicación se ejecute de la misma manera en diferentes entornos sin la necesidad de preocuparse del hardware concreto subyacente.**

Tanto los contenedores sistemas como los contenedores de aplicación son tecnologías importantes en el campo de la computación en la nube. Cada una tiene sus propias características y casos de uso, y es importante comprender sus diferencias y ventajas para tomar decisiones informadas al diseñar y administrar infraestructuras en la nube.

Con Docker, puedes agrupar una aplicación en una especie de "paquete" estándar que se utiliza para el desarrollo y despliegue de software. Este paquete enjaula la aplicación y sus dependencias (librerías) necesarias para que la aplicación funcione sin problemas en cualquier entorno informático.

Antes de Docker, las empresas solían usar máquinas virtuales (VM) para ejecutar aplicaciones aisladas. Estas máquinas virtuales permiten a los desarrolladores dividir un servidor físico en varios. Sin embargo, este enfoque tiene algunas desventajas:

Cada máquina virtual contiene una copia completa del sistema operativo y de la aplicación, así como otros archivos necesarios para su funcionamiento. Estos archivos pueden ocupar mucho espacio en el disco duro, a veces decenas de gigabytes. Además, la virtualización del hardware para cada máquina virtual puede requerir muchos recursos y ralentizar el rendimiento.

> El arranque de las máquinas virtuales tradicionales suele ser incluso más lento que el arranque del SO de la máquina anfitriona.

Cuando cae un contenedor por un fallo del software de la propia aplicación o de alguna de sus dependencias, puede volverse a inicializar el contenedor completo en segundos, sin producirse apenas corte en el servicio.

Sin embargo, los contenedores de aplicaciones no nos sirven para emular escenarios concretos con hardware concretos, ya que la filosofía es la abstracción completa del hardware de la máquina anfitriona. No están diseñados, por tanto, para configurar o limitar el tipo de hardware del contenedor, al contrario de los contenedores LXC, aunque se podrían hacer utilizando comandos especiales.

Los contenedores Docker, originalmente, son un derivado de los contenedores LXC pero pensados para ser utilizados por desarrolladores software y usuarios, utilizando en principio, todos los recursos disponibles por el sistema operativo del host anfitrión.

Es muy común utilizar contenedores Docker sobre un contenedor LXC para limitar el hardware que pueda utilizar las aplicaciones ejecutadas en Docker.

> En resumen, **los contenedores de aplicaciones (Docker) encapsulan aplicaciones y sus dependencias, abstrayéndose por completo del hardware del servidor real sin perder la velocidad nativa de una instalación tradicional** y optimizando los recursos, pudiendo ser compartidos por otros contenedores o procesos. Las ventajas en la seguridad, escalabilidad, velocidad de cómputo y economía de recursos hacen que esta tecnología sea la base del éxito de la computación en la nube.

Revisión #11
Creado 30 abril 2024 14:33:19 por Daniel Cano Verdú
Actualizado 30 abril 2024 16:54:56 por Daniel Cano Verdú