

Sistemas Operativos en Red

UD 02. Máquinas virtuales y contenedores



Autor: Sergi García

Actualizado Septiembre 2023



Licencia



Reconocimiento - No comercial - CompartirIgual (BY-NC-SA): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se ha de hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán diferentes símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

Importante

Atención

Interesante

ÍNDICE

1. Virtualización	3
2. Definición de máquina virtual	3
3. Tipos de máquinas virtuales	3
3.1 Máquinas virtuales de sistema (System Virtual Machines)	3
3.2 Máquinas virtuales de proceso (Process Virtual Machines)	4
4. Ventajas/Desventajas de las máquinas virtuales	4
5. Cómo configurar una máquina virtual: VirtualBox	5
6. Máquinas virtuales VS contenedores	5
6.1 ¿Contenedores o máquinas virtuales?	6
7. Contenedores: Docker	6
7.1 Ventajas de Docker	7
8. Instalando Docker	7
8.1 Comandos útiles de Docker	7
8.2 Docker Compose	8
9. Bibliografía	8

UNIDAD 02. MÁQUINAS VIRTUALES Y CONTENEDORES

1. VIRTUALIZACIÓN

La virtualización es un conjunto de técnicas hardware/software que permiten abstraer el hardware y software real y simular recursos físicos, sistemas operativos, etc.

La virtualización se suele implementar en instituciones, empresas, escuelas, etc. porque es fácil de implementar y tiene muchas ventajas.

Un ejemplo de virtualización son las máquinas virtuales. Además, existen otros ejemplos de virtualización, como la emulación de videojuegos o los contenedores. Recuerda, la virtualización es un conjunto de técnicas. Esas técnicas se utilizan para crear máquinas virtuales y contenedores.

2. DEFINICIÓN DE MÁQUINA VIRTUAL

En ocasiones, necesitamos probar un nuevo sistema operativo, una configuración o un programa en un entorno limpio y aislado. ¿Cómo podemos hacerlo fácilmente? Usando máquinas virtuales.

Una máquina virtual permite simular una computadora (con su propio sistema operativo) y ejecutar programas, configuraciones, etc. Las máquinas virtuales se crean utilizando un programa de virtualización que se ejecuta sobre el sistema operativo de una máquina real.

Hay muchos ejemplos de máquinas virtuales, cada una con sus propias características: VirtualBox, VMware, VirtualPC, Parallels, JavaVM, .NET, etc.

3. TIPOS DE MÁQUINAS VIRTUALES

Según su funcionalidad, podemos clasificar las máquinas virtuales en dos tipos:

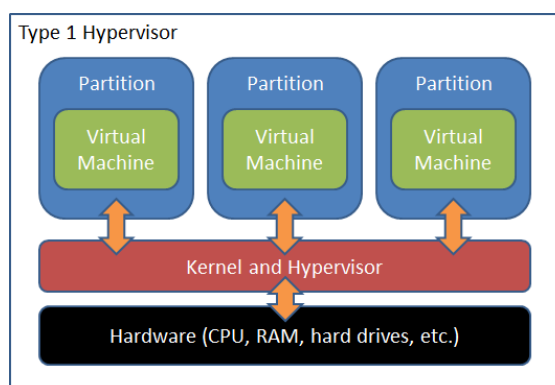
- Máquinas virtuales de sistema (System Virtual Machines).
- Máquinas virtuales de proceso (Process Virtual Machines).

3.1 Máquinas virtuales de sistema (System Virtual Machines)

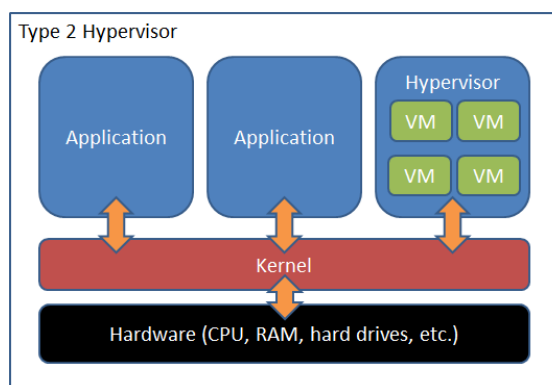
Permiten replicar la máquina real en varias máquinas virtuales, cada una con su propio SO. El software que realiza la virtualización se llama "hypervisor".

Hay dos tipos de hypervisor:

- **Hypervisor tipo 1:** el hypervisor se ejecuta directamente en el hardware.
 - Ejemplos de hypervisor tipo son: VMware ESXi (free), VMware ESX, Xen(libre), Citrix XenServer (free), Microsoft Hyper-V Server (free).



- **Hypervisor tipo 2:** el hypervisor se ejecuta como una aplicación en un sistema operativo host.
 - El hypervisor se ejecuta en un sistema operativo "host" y crea máquinas virtuales con un sistema operativo "invitado".
 - Normalmente, el sistema operativo de la máquina real se denomina "host" y el sistema operativo de la máquina virtual se denomina "invitado".
 - **Desventaja:** el sistema operativo anfitrión comparte recursos con el sistema operativo invitado.
 - Ejemplos de hypervisor de tipo 2: VirtualBox (gratis), VMware Workstation, VMware Player (gratis), QEMU (gratis).



! **Atención:** en este módulo, usaremos un hypervisor tipo 2. En nuestros apuntes de clase, el hypervisor Tipo 2 se denominará simplemente “Máquina virtual”.

3.2 Máquinas virtuales de proceso (Process Virtual Machines)

Este tipo de máquinas virtuales se ejecuta como una aplicación normal dentro de un sistema operativo host y admite un único proceso. Se crea cuando se inicia ese proceso y se destruye cuando sale. Su propósito es proporcionar una programación independiente de la plataforma.

Algunos ejemplos:

- **Máquina virtual de Java:** el “compilador de Java” genera “códigos de bytes de Java” y una “máquina virtual de Java” ejecuta esos “códigos de bytes de Java” en cada sistema operativo donde existe una “máquina virtual de Java”.
- **.NET:** ejecuta aplicaciones .NET donde esté implementado .NET (Mono para Linux, diferentes versiones de Windows, etc.).

4. VENTAJAS/DESVENTAJAS DE LAS MÁQUINAS VIRTUALES

Ventajas de las máquinas virtuales:

- Puedes utilizar varios sistemas operativos al mismo tiempo.
- Puedes probar un sistema operativo antes de instalarlo en una máquina real.
- Puedes utilizar aplicaciones que no estén disponibles en su sistema operativo host.
- Puedes emular un tipo diferente de computadora (con otro conjunto de instrucciones).
- Puedes crear entornos de prueba.
- Puedes guardar el estado actual y restaurarlo más tarde (consume espacio en disco).
- Es fácil de clonar o realizar copias de seguridad.
- Puedes ahorrar energía, recursos, espacio, etc. emulando ordenadores antiguos, lo cual es Respetuoso con el medio ambiente (evita construir ordenadores, componentes, etc.)

Desventajas de las máquinas virtuales:

- Comparten recursos con otras máquinas virtuales y con el sistema operativo host.
- El rendimiento es inferior al de las máquinas reales.

5. CÓMO CONFIGURAR UNA MÁQUINA VIRTUAL: VIRTUALBOX

VirtualBox es un "hipervisor tipo 2" gratuito disponible para los sistemas operativos más populares. Está disponible en <https://www.virtualbox.org/>

Hay varios tutoriales disponibles en Internet que explican cómo hacerlo:

- Cómo instalar VirtualBox en Ubuntu: <https://www.youtube.com/watch?v=QkJmahizwO4>
- Cómo instalar VirtualBox en Windows 10: <https://youtu.be/8mns5yqMfZk>
- Como configurar una máquina en VirtualBox: https://youtu.be/H_ustCy4Ks8

Para mejorar su rendimiento, VirtualBox le permite instalar en su sistema operativo invitado una aplicación llamada "Guest additions". Es muy útil y recomiendo instalarlo.

- Instalar "Guest additions" en Ubuntu: <https://youtu.be/Q84boOmiPW8>
- Instalar "Guest additions" en Windows 10: https://youtu.be/Bb_kJd3ISxQ

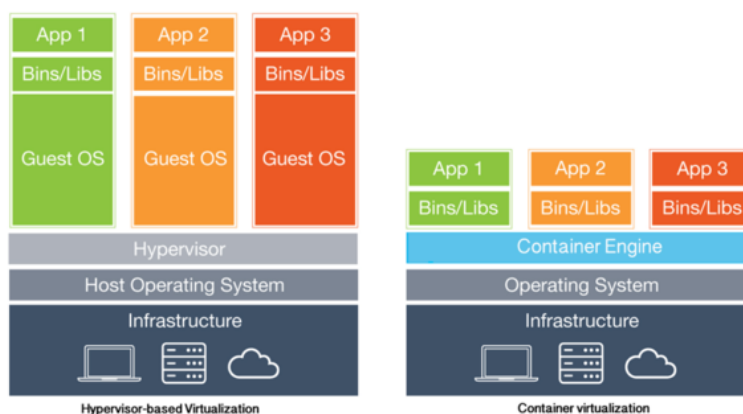
6. MÁQUINAS VIRTUALES VS CONTENEDORES

Las máquinas virtuales consisten en un hypervisor que funciona sobre un hardware físico, simulando uno o varios "hardware falsos". Cada uno de esos "hardware falsos" permite emular máquinas virtuales (cada una con su propio sistema operativo).

La virtualización ha tenido mucho éxito en los últimos años en el despliegue de aplicaciones y aprovisionamiento de infraestructuras, pero ahora el paradigma que triunfa son los "Contenedores ligeros" o simplemente "Contenedores".

Los "contenedores" son una tecnología similar a las máquinas virtuales, pero con la gran diferencia de que en lugar de utilizar un hypervisor simulando un sistema operativo completo, los contenedores usan el propio sistema operativo del anfitrión, sin simular hardware ni otro sistema operativo. Dicho de otra forma, realmente cada contenedor es en realidad un "entorno privado" del sistema operativo anfitrión, con sus propios procesos, su propia memoria y su propio disco.

Gracias a que los contenedores no simulan un sistema operativo, el sistema es más rápido (no pasa por la capa de virtualización) y el tamaño de los contenedores es mucho menor (ya que no necesitan guardar el sistema operativo completo), haciéndolos eficientes, más fáciles de migrar, iniciar, recuperar, pasar a la nube, etc.



6.1 ¿Contenedores o máquinas virtuales?

En resumen, es útil ejecutar un contenedor si queremos:

- Obtener mejor rendimiento que con una máquina virtual clásica.
- Desarrollar una aplicación que pueda distribuirse sin problemas de configuración ni dependencia en el mismo sistema operativo que nuestro host.
- Desarrollar una aplicación fácil de migrar a la nube con casi cualquier cambio.
- Ejecutar múltiples copias de la misma aplicación (conjunto de aplicaciones que funcionan juntas).


Si queremos flexibilidad (por ejemplo, usar un sistema operativo diferente al sistema operativo host) o ejecutar múltiples aplicaciones diferentes en diferentes sistemas operativos, debemos usar máquinas virtuales.

7. CONTENEDORES: DOCKER

Existen tecnologías que nos permiten utilizar contenedores Linux, como LXC o LXD. Son populares, pero el sistema de contenedores más popular es Docker.

Puede encontrar más información sobre las diferencias entre las tecnologías de contenedores en: <https://unix.stackexchange.com/questions/254956/what-is-the-difference-between-docker-lxd-and-lxc>

Docker es un proyecto de código abierto que automatiza la implementación de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de la virtualización a nivel del sistema operativo en Linux.

 **Interesante:** tienes mucha información sobre cómo usar Docker con ejemplos prácticos en mi curso de Docker, disponible en <https://sergarb1.github.io/CursoIntroduccionADocker/>.

Docker utiliza funciones de aislamiento de recursos del kernel de Linux:

- **cgroups:** grupos de control, es una característica que limita las cuentas y aísla el uso de recursos como CPU, memoria, E/S de disco, red, etc.) de una colección de procesos.
- **namespaces:** limita qué recursos pueden ser vistos por un conjunto de procesos.

Esto permite que se ejecuten "contenedores" independientes dentro de una única instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales.

Podemos crear contenedores (con comandos como "docker run"), podemos comunicarnos con contenedores Docker vinculando entrada/salida desde una consola al contenedor (comandos "docker attach") y con comandos como "docker cp" para copiar archivos.

Por lo tanto, Docker no admite interfaz gráfica. Sin embargo, es posible manejarlos con interfaz gráfica con algunas de estas soluciones:

- Instalar un servidor X y conectarse con un cliente XWindows al contenedor Docker.
- Usando software de administrador remoto como VNC (Recomendamos instalar NoVNC en nuestro contenedor Docker). Con esta opción no necesitamos un cliente especial y podemos operar directamente desde el navegador.

Más información: [https://es.wikipedia.org/wiki/Docker_\(software\)](https://es.wikipedia.org/wiki/Docker_(software))

7.1 Ventajas de Docker

Algunas de las ventajas de Docker son:

- Independencia de la plataforma: permite el uso de contenedores en cualquier sistema compatible: puede ser Windows, Mac, Linux, etc.
- En los sistemas Windows, si tienen capacidades de Linux, funciona en modo nativo. En los que no los tienen, instala una máquina virtual (en Virtual Box) y dentro de esa máquina virtual lanza Docker.
- Es muy fácil crear e iniciar un contenedor.
- Cada contenedor tiene su propio entorno de red, configurable y compartido con otros contenedores si es necesario.
- Hay imágenes de Docker. Son como "plantillas". Podemos crear tantos contenedores como queramos bajo una misma imagen.
 - Además, podemos descargar imágenes de terceros y utilizarlas para crear contenedores.
- Controlamos las versiones de todo el software dentro del contenedor: sistema operativo, versión para nuestra base de datos, servidor de aplicaciones, etc. Esto elimina problemas de configuración al portar el sistema de una máquina a otra.
- Existen soluciones de clustering y alta disponibilidad como Kubernetes o Docker swarm que se utilizan en entornos de producción.
- Dispone de un potente buscador de imágenes ya pregeneradas en "Docker Hub" donde podremos encontrar tanto imágenes oficiales como imágenes personales compartidas por la comunidad.
- Tiene soporte en los principales sistemas de la nube: Azure, AWS, Google Cloud, OVH. De hecho, cuando contratas un VPS (Servidor Privado Virtual), suele ser un contenedor.

8. Instalando Docker

! Atención: aunque es posible, NO recomendamos instalar Docker en un sistema diferente a Linux. Para fines educativos, es mejor utilizar una máquina virtual Linux e instalar Docker que instalar Docker en Windows o macOS. **Hazlo bajo tu riesgo.**

Instalación de Docker en Ubuntu Linux:

- <https://docs.docker.com/engine/install/ubuntu/>
- <https://www.youtube.com/watch?v=wCSMDtHPBso>

Instalación de Docker en Windows:

- <https://docs.docker.com/desktop/install/windows-install/>

Instalación de Docker en macOS

- <https://docs.docker.com/desktop/install/mac-install/>

8.1 Comandos útiles de Docker

La documentación oficial de Docker es <https://docs.docker.com/reference/>

Algunos comandos útiles de Docker son:

- `"docker run -it image"`: crea un contenedor con la imagen dada. Si esa imagen no está en la máquina, la descarga automáticamente desde Docker Hub y crea el contenedor. El parámetro "-it" vincula la entrada y salida del contenedor a la consola actual.
 - CUIDADO: si ejecutas este comando dos o tres veces... ¡¡crearás dos o tres contenedores!! Para arrancar de nuevo un contenedor parado y no duplicarlo, utiliza el comando `"docker start container"`.
- `"docker start -i container"`: inicia un contenedor creado previamente. Si ha creado un contenedor y desea ejecutarlo nuevamente, debe usar este comando. El parámetro -i

vinculará la entrada del contenedor a la consola actual.

- `"docker ps"`: permite ver las máquinas Docker actualmente en ejecución.
- `"docker ps -a"`: permite ver todas las máquinas Docker actualmente en ejecución o no.
- `"docker image ls"`: permite ver las imágenes de Docker (no confundir con contenedores) que ha descargado en su máquina.
- `"docker cp origen destino"`: permite copiar archivos entre una máquina real y un contenedor Docker.
- `"docker login"`: comando para iniciar sesión en "Docker hub" usando la consola.
- `"docker pull urlmaquina:etiqueta"`: comando para descargar una imagen de Docker. Si no se configura otro sitio, las intenta descargar de "Docker hub".
- `"docker commit etiqueta"`: comando para confirmar una etiqueta con los cambios realizados en un contenedor Docker.
- `"docker push urlmaquina:etiqueta"`: comando para cargar cambios en una imagen de Docker en "Docker hub".

En nuestro Aules (Moodle) hay un Docker "Cheatsheet" en español con varios comandos y sus ejemplos asociados.

<https://raw.githubusercontent.com/sergarb1/CursoIntroduccionADocker/main/FuentesCurso/Docker%20CheatSheet%20COMPLETA.pdf>

8.2 Docker Compose

"Docker Compose" es una herramienta que ayuda a configurar uno o varios "contenedores Docker" mediante un archivo en formato YAML. Es muy fácil de usar y muy práctico para crear/eliminar rápidamente "contenedores Docker".

Más información en: <https://docs.docker.com/compose/>

Para encontrar ejemplos prácticos útiles de cómo "Docker Compose", puedes leer la sección "Docker Compose" del curso <https://sergarb1.github.io/CursoIntroduccionADocker/>

9. BIBLIOGRAFÍA

[1] Virtualización

<https://en.wikipedia.org/wiki/Virtualization>

[2] Hypervisores

<https://en.wikipedia.org/wiki/Hypervisor>

[3] Máquinas virtuales

https://en.wikipedia.org/wiki/Virtual_machine

[4] Docker

[https://en.wikipedia.org/wiki/Docker_\(software\)](https://en.wikipedia.org/wiki/Docker_(software))