TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

Trabajo Práctico – Virtualización

Presentado por:

Israel Garcia Moscoso – isragadiel@gmail.com
Juan Esteban Gelos – juan_gelos@yahool.com

Materia: Arquitectura y Sistemas Operativos
Profesor: Diego Lobos
Tutor: Nicolas Carcaño
Fecha de Entrega: 5 de junio de 2025

Índice

1. Introducción

La virtualización es una tecnología que permite crear versiones virtuales de recursos informáticos como sistemas operativos, servidores, almacenamiento o redes.

Con la virtualización se puede tener varios sistemas funcionando dentro de una misma máquina física, de forma aislada e independiente.

A grandes rasgos podemos decir que la virtualización se divide en dos ejes principales: 1) Virtualización tradicional a través de la configuración de máquinas virtuales (virtualización del hardware); y 2) Virtualizaciones modernas a través de contenedores, que virtualizan aplicaciones dentro del mismo sistema operativo del equipo que lo contiene.

Claro está que existen otros tipos de virtualizaciones como las de almacenamiento, las de red, las de escritorio y las de aplicaciones.

En este trabajo pretendemos abordar la virtualización de hardware, explorando conceptos esenciales, realizando la instalación de Oracle VirtualBox, creando y configurando una máquina virtual a partir de una imagen de Ubuntu. Posteriormente cargaremos Python, desarrollaremos una aplicación simple y la ejecutaremos.

2. Marco Teórico

La virtualización de hardware es el proceso mediante el cual se emula un entorno de hardware utilizando software especializado denominado hipervisor. Este software permite ejecutar múltiples máquinas virtuales (VMs) sobre un solo equipo físico, en las cuales cada VM opera como si tuviera su propio conjunto independiente de recursos como procesador, memoria, almacenamiento, red, etc.

Una pequeña reseña histórica. ¿Cómo surge la virtualización?

En las décadas de 1960/70, la empresa IBM desarrolló IBM System/370 que incorporó por primera vez una virtualización avanzada del hardware mediante hipervisores. Este sistema permitía dividir el sistema físico en múltiples entornos virtuales completamente independientes, donde cada entorno virtual podía ejecutar su propio sistema operativo. Esto les permitía mejorar el uso de costosos mainframes, aislando procesos de distintos usuarios, lo que también les permitía probar diferentes sistemas operativos sin necesidad de varias máquinas físicas.

Esto pasó a un segundo plano en la década de los 80 y 90 debido al abaratamiento del hardware, pero a partir de los años 2000, VMware y otras empresas la revivieron para servidores y centros de datos, siendo hoy la base fundamental de la computación en la nube, contenedores y entornos de desarrollo.

La virtualización tiene los siguientes elementos clave:

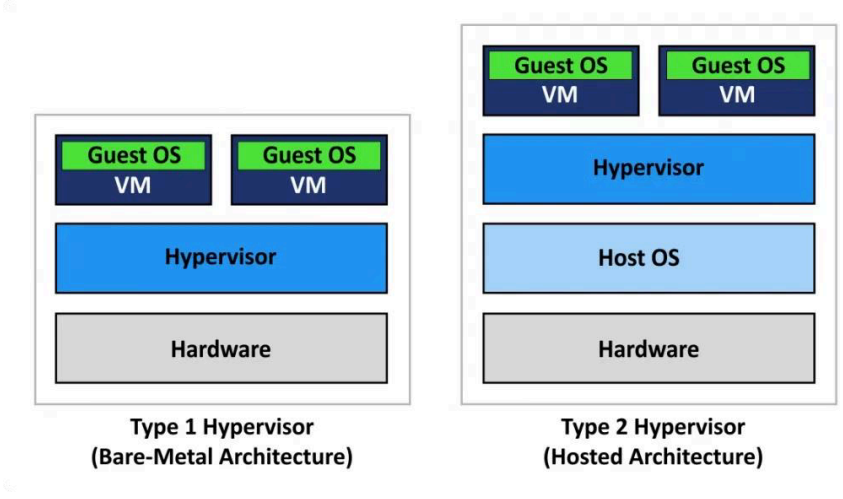
1. Hipervisor: Es el software que se encarga de crear y gestionar las máquinas virtuales. Puede instalarse directamente sobre el hardware (hipervisor de tipo 1) o sobre un sistema operativo anfitrión (hipervisor de tipo 2).
2. Máquina virtual (VM): Es una instancia de sistema operativo y aplicaciones que corre de manera aislada sobre un entorno virtualizado.
3. Host: Es la máquina física sobre la cual se ejecutan las máquinas virtuales.

4. Guest: Es el sistema operativo que se ejecuta dentro de una máquina virtual.

Desglosamos los elementos de la virtualización de hardware:

1. Hypervisor: como mencionamos anteriormente, también conocido como monitor de máquina virtual o VMM, es un software que crea y ejecuta máquinas virtuales (VM). Permite que un equipo host admita varias VM invitadas compartiendo virtualmente sus recursos, como memoria y procesamiento. Hacen posible la virtualización al traducir las solicitudes entre los recursos físicos y virtuales.

Existen dos tipos principales de hipervisores: el de "Tipo 1" (o "bare metal") y el de "Tipo 2" (o "alojado"). El de tipo 1 actúa como un sistema operativo ligero y se ejecuta directamente en el hardware del host, mientras que el de tipo 2 se ejecuta como una capa de software en un sistema operativo, como otros programas informáticos.



Cómo funciona el Hypervisor Tipo 1 (Bare-metal)

Se ejecuta directamente sobre el hardware físico, eliminando la necesidad de un sistema operativo anfitrión. Su función es asumir el rol que normalmente tendría un sistema operativo tradicional, gestionando y compartiendo los recursos del hardware entre múltiples máquinas virtuales (VMs).

Cuando el hardware se enciende, el hypervisor de tipo 1 se carga directamente desde el firmware (como BIOS o UEFI) y toma control total del hardware.

El hypervisor administra directamente todos los recursos físicos del sistema:

• CPU: Divide los ciclos de CPU entre las VMs según sea necesario, asignando núcleos virtuales (vCPU).

• Memoria RAM: Asigna bloques de memoria física a cada VM, asegurando aislamiento para evitar que una VM acceda a la memoria de otra.

• Almacenamiento: Utiliza controladores para interactuar directamente con los discos físicos y proporciona discos virtuales a las VMs.

• Dispositivos de Red: Configura adaptadores virtuales que simulan interfaces de red para cada VM.

El hypervisor crea un entorno virtual para cada VM, en el que cada una cree que tiene acceso exclusivo al hardware, proporcionando dispositivos virtuales (como discos, tarjetas de red y adaptadores gráficos) que simulan el hardware físico.

Los controladores de hardware del hypervisor se comunican directamente con dispositivos como la CPU, GPU, y almacenamiento.

Esto elimina intermediarios, reduciendo la latencia y mejorando el rendimiento.

Cómo funciona el Hypervisor Tipo 2 (Alojado)

Se ejecuta como una aplicación dentro de un sistema operativo anfitrión.

Cuando el hardware se enciende, el sistema operativo anfitrión (Windows, Linux, macOS) se carga primero, luego, se ejecuta el hypervisor como una aplicación dentro de ese sistema operativo.

El hypervisor no interactúa directamente con el hardware, sino que solicita acceso al hardware a través del sistema operativo anfitrión. Este proceso introduce una capa adicional de latencia.

El hypervisor tipo 2, depende del sistema operativo anfitrión para gestionar los recursos del hardware, lo que puede limitar su capacidad de optimización.

Además, el sistema anfitrión consume recursos (CPU, memoria, etc.), reduciendo los disponibles para las VMs.

Existen varias ventajas al utilizar un hypervisor que aloja varias máquinas virtuales:

Velocidad: permiten la creación instantánea de máquinas virtuales, a diferencia de los servidores físicos.

Eficiencia: ejecutan varias máquinas virtuales en los recursos de una máquina física también permiten un uso más eficiente de un servidor físico. Es más rentable y eficiente

energéticamente ejecutar varias máquinas virtuales en una máquina física que ejecutar varias máquinas físicas infrautilizadas para la misma tarea.

Flexibilidad : permiten que los sistemas operativos y sus aplicaciones asociadas se ejecuten en una variedad de tipos de hardware porque el hipervisor separa el sistema operativo del hardware subyacente, por lo que el software ya no depende de dispositivos o controladores de hardware específicos.

Portabilidad : permiten que varios sistemas operativos residan en el mismo servidor físico (máquina host). Dado que las máquinas virtuales que ejecuta el hipervisor son independientes de la máquina física, son portátiles. Los equipos de TI pueden trasladar cargas de trabajo y asignar recursos de red, memoria, almacenamiento y procesamiento entre varios servidores según sea necesario, moviéndose de una máquina a otra o de una plataforma a otra. Cuando una aplicación necesita mayor capacidad de procesamiento, el software de virtualización le permite acceder sin problemas a otras máquinas.

La desventaja de los hipervisores alojados es que la latencia es mayor que la de los hipervisores físicos. Esto se debe a que la comunicación entre el hardware y el hipervisor debe pasar por la capa adicional del SO.

2. Máquina virtual (VM): puntualizando, una máquina virtual es un entorno de software que emula una computadora física. Se ejecuta dentro de una computadora real (host) y se comporta como si fuera una computadora independiente, con su propio sistema operativo, procesador, memoria, almacenamiento y red.

Una máquina virtual se crea y administra a través de un hypervisor, a través del cual se asigna parte de los recursos físicos del host como CPU, RAM, disco, etc.. Una vez creada, se le puede instalar un sistema operativo completo como Windows, Linux o incluso otro macOS. La utilidad de una MV radica en la posibilidad de hacer pruebas de software o sistemas operativos; simular entornos de red; aislar entornos (por ejemplo, para pruebas de seguridad); recuperación ante desastres ya que permite la realización de snapshot; reducción de costos en centros de datos; desarrollo multiplataforma; y educación.

Los componentes principales de una VM son:

1. Sistema operativo invitado (guest OS): el SO que corre dentro de la VM.
2. Disco duro virtual (VHD, VDI, etc.): archivo que simula un disco físico.
3. Memoria asignada: la cantidad de RAM tomada del host.

4. CPU virtual: núcleos de procesador asignados por el hypervisor.
5. Interfaces de red virtuales: para conectarse a internet o redes locales.

Las ventajas de usar máquinas virtuales son:

1. Aislamiento: Lo que pase en una VM no afecta al sistema principal
2. Flexibilidad: Podés tener varias VMs con distintos sistemas operativos
3. Portabilidad: Una VM es un archivo que podés mover o copiar Snapshots: Guardás un estado y podés volver a él más tarde
4. Aprovechamiento de recursos: Múltiples VMs en una sola máquina física

Como desventajas podemos mencionar:

1. Rendimiento inferior: Nunca es igual al de un sistema instalado físicamente
2. Requiere buena PC: Para ejecutar varias VMs, necesitás buen hardware
3. Administración compleja: En entornos grandes, requiere planificación

3. Caso Práctico

Realizaremos una virtualización de hardware a través del siguiente ejercicio:

1. Instalación de Oracle Virtual Box
2. Descarga de la versión 24.04 LTS de Ubuntu Desktop
3. Creación de una máquina virtual (MV), con la imagen de Ubuntu descargada
4. Configuración de la MV: 4 Gb de RAM, 2 núcleos de CPU y 20 Gb de almacenamiento en disco.
5. Instalación de Ubuntu como sistema operativo de esa MV.
6. Verificación de Python3 preinstalado, caso contrario lo instalaremos.
7. Crear una aplicación simple en Python
8. Correr la aplicación.

4. Metodología Utilizada

Se desarrolló bajo un enfoque práctico-exploratorio, con el objetivo de implementar y analizar un entorno de virtualización funcional. La metodología se dividió en las siguientes etapas:

- Preparación del entorno:
    - Instalación de VirtualBox en el sistema host (Windows).
    - Descarga de la imagen ISO de Ubuntu Desktop.

- Implementación de la máquina virtual:
    - Creación de una VM desde VirtualBox con 4 Gb de RAM, 2 núcleos de CPU y 20 Gb de almacenamiento de disco.
    - Instalación del sistema operativo Ubuntu dentro de la VM.

- Configuración del entorno virtualizado:
    - Instalación de Guest Additions.
    - Habilitación de carpeta compartida (Downloads) y portapapeles bidireccional.
    - Instalación de software adicional (VSC) con las extensiones necesarias.

- Registro y verificación:
    - Capturas de pantalla de cada fase (Ver anexos 1 a 6).
    - Comprobación de la comunicación entre host y VM, a través de la carpeta compartida, se copiaron los archivos que guardaban los prints de pantalla que posteriormente fueron incluidos en los anexos de este documento.
    - Registro de problemas y soluciones aplicadas.

5. Resultados Obtenidos

- El hypervisor de tipo 2 elegido (VirtualBox) se instaló correctamente.
- El sistema operativo pudo ser instalado en la máquina virtual sin inconvenientes.
- La configuración de la MV se realizó teniendo en cuenta que era simplemente para un trabajo práctico y no en función a las necesidades operativas reales para un desempeño eficiente.
- Instalamos un conjunto de herramientas que nos brindaron mejor rendimiento e integración con el sistema host (Guest Additions)
- Instalamos VSC como editor de código.
- Usamos la carpeta compartida para mover una aplicación que ya estaba desarrollada en la máquina host, para probar su ejecución en la MV.
- Ejecutamos la aplicación tanto desde el editor de código como desde la terminal, obteniendo los mismos resultados y una ejecución sin problemas, igual que si estuviéramos trabajando sobre la host.

6. Conclusiones

7. Bibliografía

Material de estudio de la Cátedra Arquitectura y Sistemas Operativos de la carrera Tecnicatura Universitaria en Programación a distancia, dictada por la Universidad Tecnológica Nacional.

Historia:

- https://walternavarrete.com/historia-sobre-virtualizacion-de-sistemas/
- https://www.techtarget.com/searchitoperations/feature/The-history-of-virtualization-and-its-mark-on-data-center-management
- https://www.studocu.com/latam/document/politecnico-instituto-tecnologico-de-las-americas/linguistica-aplicada-a-la-ensenanza-del-espanol-como-lengua-materna/historia-y-fundamentos-de-la-virtualizacion-desde-los-primer/90780423

hipervisor:

- https://www.vmware.com/topics/hypervisor

Ubuntu:

- https://ubuntu.com/desktop (descarga)
- https://help.ubuntu.com/lts/ubuntu-help/index.html
- https://docs.ubuntu.com/ (documentación)

Citrix:

- https://docs.citrix.com/es-es/citrix-workspace-app-for-windows

8. Anexos

- 1 y 2
- 3 Creación de una máquina virtual (MV), con la imagen de Ubuntu descargada
- 4 Configuración de la MV: 4 Gb de RAM, 2 núcleos de CPU y 20 Gb de almacenamiento en disco.
- 5 Instalación de Ubuntu como sistema operativo de esa MV.
- 6 Instalación de VSC y ejecución de una aplicación desarrollada en Python.


UTN
TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA