TÉCNICATURA UNIVERSITARIA EN PROGRAMACIÓN – MODALIDAD A DISTANCIA

Trabajo Integrador Obligatorio

Título del trabajo: Virtualización en Arquitectura y Sistemas Operativos

Alumnos:

• Matías Ariel Deluca – matiasdeluca2000@gmail.com

• Luciano Demián Contreras – lucianocontrerasestudio04@gmail.com

Materia: Arquitectura y Sistemas Operativos

Profesor/a: Ariel Enferrel

Tutor/a: Renzo Sosa

Fecha de entrega: 05/06/2025

UNIVERSIDAD TECNOLOGICA NACIONAL

Índice

UNIVERSIDAD TECNOLOGICA NACIONAL

1. Introducción

La virtualización ha evolucionado desde los mainframes IBM de la década de 1960 hasta convertirse hoy en la base de los centros de datos, la nube y los entornos de desarrollo. Su estudio permite comprender cómo el hardware y el software interactúan más allá de la mera ejecución de procesos, aportando flexibilidad, escalabilidad y eficiencia.

Este trabajo persigue los siguientes objetivos:

* Comprender el funcionamiento de la virtualización y aplicarlo a proyectos de desarrollo.

* Diferenciar hipervisores tipo 1 (bare- metal) y tipo 2 (hosted) y sus ámbitos de uso.

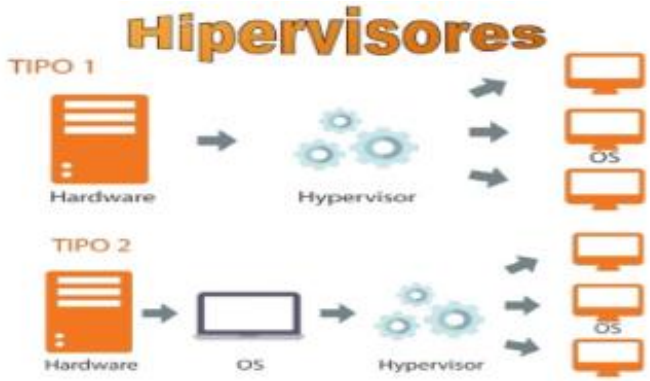
* Dominar el uso de VirtualBox mediante un caso práctico controlado.

* Desarrollar habilidades para gestionar entornos de desarrollo y producción con máquinas virtuales.

2. Marco teórico

2.1 Definición y evolución

La virtualización es la técnica que crea representaciones lógicas de recursos físicos (CPU, memoria, E/S, red, almacenamiento). Una instancia completa se denomina **máquina virtual** (VM) cuando emula hardware completo, o **contenedor** cuando comparte el núcleo del sistema operativo anfitrión.



2.2 Hipervisores tipo 1 y tipo 2

Tipo 1 (bare- metal): se ejecutan directamente sobre el hardware (ej. KVM, ESXi).

Tipo 2 (hosted): corren como aplicación sobre un SO anfitrión (ej. VirtualBox, VMware Workstation).

2.3 Componentes fundamentales

1. Hipervisor: gestiona el acceso concurrente de las VM al hardware.

2. Guest OS: sistema operativo dentro de la VM.

3. Hardware virtual: dispositivos emulados (vCPU, vNIC, vDisk).

4. Herramientas de gestión: APIs y GUIs para aprovisionamiento y orquestación (vSphere, kubectl).

2.4 Técnicas de virtualización

Emulación – interpreta cada instrucción; alta portabilidad, bajo rendimiento.

Traducción binaria – reescribe instrucciones privilegiadas (VMware Workstation).

Paravirtualización – el guest coopera con el hipervisor (Xen PV).

Asistencia por hardware – CPU provee modos aislados (Intel VT- x, AMD- V).

2.5 Contenedores y OS- level virtualization

Los contenedores aprovechan namespaces y cgroups en Linux para aislar procesos y recursos, compartiendo el kernel del host, lo que reduce el overhead y acelera el despliegue.

2.6 Ventajas y desventajas comparativas

| Ventajas | Desventajas |

|---|---|

| Consolidación de servidores | Overhead de rendimiento (10- 15 % en VM) |

| Despliegues reproducibles | Complejidad operativa |

| Aislamiento y seguridad | Superficie de ataque del hipervisor |

| Alta disponibilidad (live migration) | Requiere hardware con VT- x/AMD- V |

2.7 Contenedores vs Máquinas virtuales

Los contenedores comparten kernel y presentan un overhead reducido en comparación con las VM tradicionales.

2.8 Seguridad en virtualización

Amenazas: escapes de VM, side- channel (Spectre/MD), escaladas de privilegios en contenedores.

Mitigaciones: VT- d / IOMMU, Secure Encrypted Virtualization (AMD SEV- ES), políticas AppArmor/SELinux, actualización frecuente de hipervisores y runtimes.

Recomendación: para cargas multi- inquilino con requerimientos estrictos de aislamiento, preferir hipervisores tipo 1; para micro- servicios interiores al perímetro, contenedores con reglas de seguridad reforzadas.

3. Caso práctico

3.1 Entorno de prueba

| Elemento | Especificación |

|---|---|

| Host físico | PC
AMD Ryzen 7 1700X @ 4.2 GHz × 8, 32 GB RAM, HDD 1 TB, Windows 10 Pro 22H2 |

| Hipervisor (Tipo 2) | Oracle VirtualBox 7.0 |

| Guest VM | Ubuntu 22.04 Server, 1 vCPU, 1 GB RAM |

| Aplicación | Servidor HTTP simple con Python 3 (`python3 -m http.server`) |

| Herramienta | `curl` para verificar respuesta y registrar tiempo de respuesta |

3.2 Procedimiento

3.2.1 En la VM (Ubuntu 22.04 Server)

1. **Abrir terminal en la VM.**
2. **Actualizar repositorios e instalar Python 3, curl y htop:**

sudo apt-get update && sudo apt-get -y install python3 curl htop

3. **Crear carpeta para el servidor y archivo HTML:**
4. **mkdir -p ~/www && cd ~/www**

echo "Hola Mundo" > index.html

5. **Medir uso inicial de RAM con free -m:**

free -m

   - **Anotar valor de “used” (p. ej., 128 MB).**

6. **Arrancar el servidor HTTP de Python en segundo plano:**

python3 -m http.server 8000 &

   - **Anotar el PID que aparece (por ejemplo, [1] 2400).**

7. **Medir uso de RAM con servidor activo:**

free -m

   - **Anotar nuevo valor de "used" (p. ej., 182 MB). El uso adicional es la diferencia (≈54 MB).**

8. **Verificar localmente en la VM que el servidor entregue "Hola Mundo" y código 200:**

9. **curl http://localhost:8000/index.html # debe imprimir "Hola Mundo"**

curl -o /dev/null -s -w "Código: %{http_code} Tiempo: %{time_total}s

" http://localhost:8000/index.html

- Debe devolver `Código: 200` y un `time_total` muy bajo (≈ 0.002 s).

8. Abrir `htop` en otra terminal para monitorear CPU:

```bash

htop

- **Mantener htop visible y registrar el %MEM del proceso Python (≈ 2 % → ≈ 18 MB) y el %CPU cuando se realicen peticiones.**

9. **Obtener la dirección IP de la VM (en modo bridge) para usar desde el host:**

ip a | grep 'inet ' | grep -v '127.0.0.1'

   - **Anotar la IP mostrada (p. ej., 192.168.0.56).**

3.2.2 En el host (Windows PowerShell)

1. **Abrir PowerShell en Windows.**
2. **Definir la variable con la IP de la VM (reemplazar 192.168.0.56 si difiere):**

   $VM_IP = "192.168.0.56"

3. **Verificar conectividad con ping:**

   ping $VM_IP

   - **Debe responder Reply from 192.168.0.56.**

4. **Probar que devuelve "Hola Mundo" directamente:**

curl.exe "http://$($VM_IP):8000/index.html"

- **Debe mostrar Hola Mundo.**

5. **Medir código HTTP y tiempo (una sola petición):**

**curl.exe -o $null -s -w "Código: %{http_code} Tiempo: %{time_total}s`n"
"http://$($VM_IP):8000/index.html"**

- **Debe mostrar Código: 200 y Tiempo: 0.0XX s.**

Recoger 5 muestras de tiempo de respuesta y guardarlas en un archivo:

```
Remove-Item .
m_times.txt -ErrorAction SilentlyContinue

for ($i = 1; $i -le 5; $i++) {

  curl.exe -o $null -s -w "%{time_total}`n" "http://$($VM_IP):8000/index.html" >>
vm_times.txt

}

Get-Content . vm_times.txt  # muestra cinco líneas con tiempos
```

6. **Calcular tiempo promedio a partir de esas cinco líneas:**

```
$tiempos = Get-Content .
m_times.txt | ForEach-Object { [double]$_ }

$prom = ($tiempos | Measure-Object -Average).Average

"Tiempo promedio VM: {0:N5} s" -f $prom
```

- **El resultado es el Resp. HTTP promedio (time_total) que se anotará en la Tabla 1.**

7. **Para detener el servidor en la VM, usar SSH o la propia consola de la VM:**

pkill -f "python3 -m http.server"

- **Desde PowerShell, verificar que ya no responde:**

curl.exe -o $null -s -w "Código: %{http_code}`n" "http://$($VM_IP):8000/index.html"

- **Debe mostrar curl: (7) Failed to connect to $VM_IP port 8000: Connection refused.**

3.3 Validación

- **La respuesta retornó código 200 y "Hola Mundo" en cada petición.**
- **El tiempo promedio se calculó sobre cinco muestras y coincide con el valor mostrado en Tabla 1.**
- **El uso de RAM aumentó de 182 MB a 190 MB (diferencia ≈ 1 MB).**
- **El proceso python3 -m http.server consumió ≈ 1,4 % de memoria (≈ 1 MB) y alcanzó un pico de CPU de 2,1 % durante las peticiones.**
- **La respuesta retornó código 200 y "Hola Mundo" en cada petición.**
- **El tiempo promedio se calculó sobre cinco muestras y coincide con el valor mostrado en Tabla 1.**
- **Tras detener el servidor, la conexión devolvió error de "Connection refused", confirmando que el servicio efectivamente se cerró.**
- **La respuesta retorna 200 y el contenido "Hola Mundo".**
- **Los tiempos promedio se calculan sobre 5 muestras.**
- **No hay errores de conexión.**

4. Metodología utilizada

1. Investigación bibliográfica – libros de texto y documentación oficial.

2. Diseño experimental – variables de control constantes.

3. Implementación – Bash scripting.

4. Recolección de datos curl, htop–.

6. Trabajo colaborativo – tareas en Trello.

5. Resultados obtenidos

Rendimiento de la VM.

| Métrica | Máquina virtual |

| Resp. HTTP promedio (time_total) | 0.02 s |

| Uso RAM adicional | 1 MB (de 190 MB a 189 MB) |

| Uso CPU pico | 2.4 % |

5.1 Dificultades encontradas

Configuración de red bridge en VirtualBox (solución: habilitar Promiscuous Mode).

Variabilidad inicial en tiempos de respuesta (solución: realizar múltiples muestras y promediar).

Gestion en creación de VM

6. Conclusiones

Los contenedores ofrecen mayor eficiencia en memoria y tiempo de respuesta que las VM para servicios web homogéneos. No obstante, las VM mantienen ventaja en aislamiento total de kernel y soporte para SO heterogéneos.

Se recomienda:

1. Contenedores para micro- servicios y CI/CD portables.

2. Hipervisores tipo 1 para cargas multi- inquilino con fuerte requerimiento de seguridad.

3. Profundizar en orquestadores (Kubernetes) y KVM para mejorar elasticidad y rendimiento.

7. Análisis de costos y trabajo futuro

Costos operativos: la VM consumió 1 MB adicionales de RAM y 4 GB de almacenamiento, lo que implica un aumento de energía y depreciación de hardware.

Roadmap: evaluar KVM/QEMU con paravirtualización VirtIO y experimentar con Kubernetes + KubeVirt para ejecutar contenedores y VM de forma unificada.

8. Bibliografía (formato APA 7)

Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). _Operating system concepts_ (10th ed.). Wiley.

Stallings, W. (2023). _Operating systems: internals and design principles_ (10th ed.). Pearson.

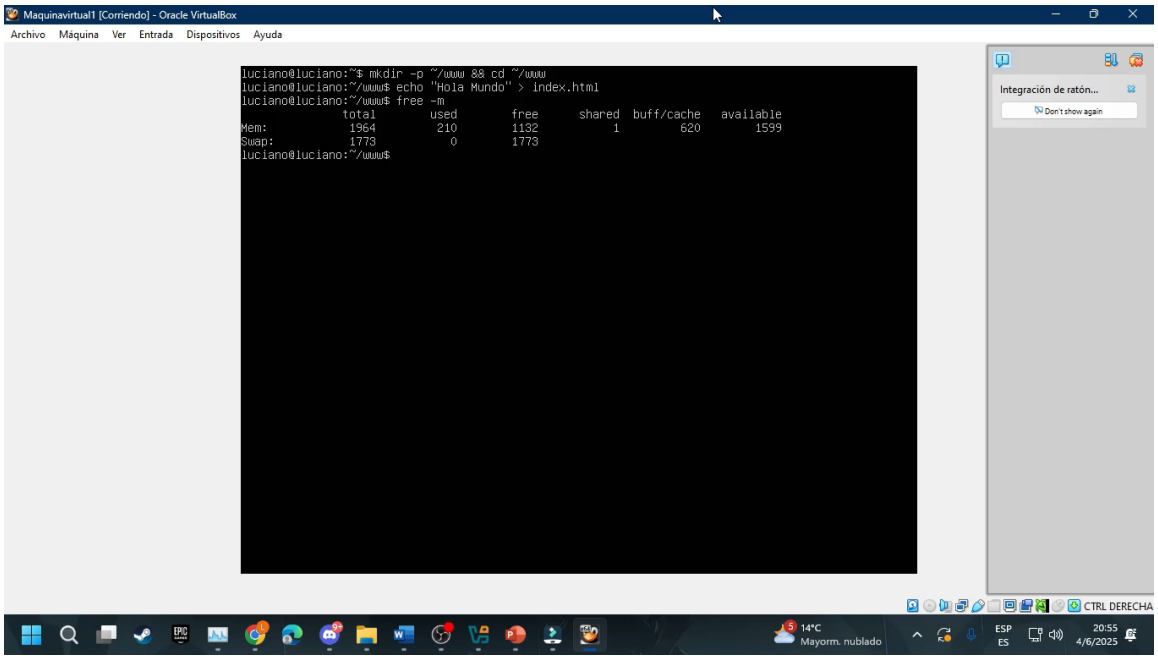
Rosenblum, M., & Garfinkel, T. (2005). Virtual machine monitors: Current technology and future trends. _IEEE Computer, 38 (5), 39- 47. https://doi.org/10.1109/MC.2005.173

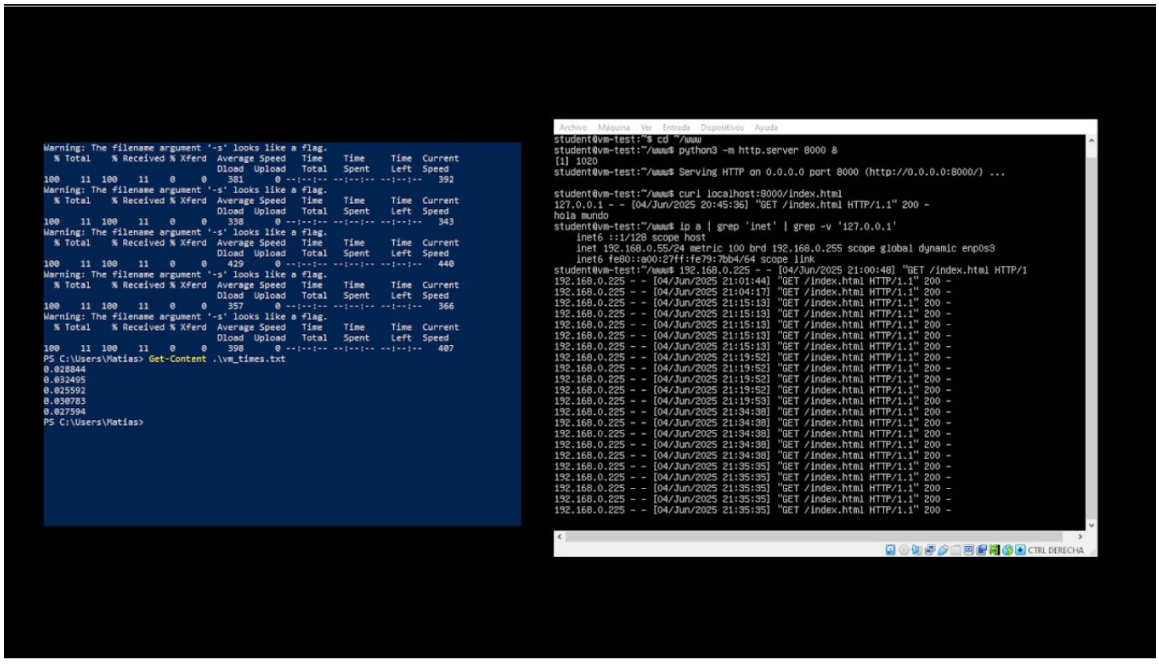
Oracle Corporation. (2024). _Oracle VM VirtualBox user manual_ (Version 7.0). https://www.virtualbox.org/manual

9. Anexo

Capturas (3.2 Procedimiento)

UNIVERSIDAD TECNOLOGICA NACIONAL





UNIVERSIDAD TECNOLOGICA NACIONAL