

# Trabajo Práctico - Virtualización con VirtualBox

## Alumnos:

Kenyi Meza - [mezakenyi@gmail.com](mailto:mezakenyi@gmail.com)

Roberto Méndez - [robermen02@gmail.com](mailto:robermen02@gmail.com)

**Materia:** Arquitectura y Sistemas Operativos

**Profesor:** Ariel enferrel

**Fecha de entrega:** 05/06/2025

---

## Índice

1. Introducción
  2. Marco Teórico
  3. Caso Práctico
  4. Metodología Utilizada
  5. Resultados Obtenidos
  6. Conclusiones
  7. Bibliografía
  8. Anexos
- 

## 1. Introducción

En este trabajo práctico se explora el concepto de virtualización usando Oracle VirtualBox sobre un sistema operativo anfitrión (host) Debian 12. El objetivo principal fue crear un entorno de servidor Ubuntu Server como sistema invitado (guest), instalar y configurar Docker, y luego desplegar un contenedor que aloje un servidor web Apache.

---

## 2. Marco Teórico

- **Virtualización:** Permite ejecutar múltiples sistemas operativos en un mismo hardware físico mediante entornos virtuales.
- **Hypervisores:**
  - *Tipo 1:* Corren directamente sobre el hardware (bare-metal), como VMware ESXi.
  - *Tipo 2:* Corren sobre un sistema operativo, como Oracle VirtualBox.
- **VirtualBox:** Es un hipervisor de tipo 2 gratuito y de código abierto, ideal para pruebas y entornos educativos.

- **Docker:** Plataforma de contenedores que permite empaquetar aplicaciones y sus dependencias para ejecutarlas de forma aislada.
  - **Contenedor vs VM:** Un contenedor es más liviano y rápido que una VM completa; comparte el kernel del host, pero está aislado a nivel de sistema de archivos, red y procesos.
- 

### 3. Caso Práctico

Se utilizó Debian 12 como sistema operativo host, sobre el cual se instaló Oracle VirtualBox. Se creó una máquina virtual con Ubuntu Server 22.04 y se configuró Docker para ejecutar un servidor web Apache dentro de un contenedor.

#### Pasos realizados:

##### Instalación de VirtualBox en Debian 12

```
wget -O- https://www.virtualbox.org/download/oracle_vbox_2016.asc | \ sudo gpg  
--yes --output /usr/share/keyrings/oracle-virtualbox-2016.gpg --dearmor
```

```
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/oracle-virtualbox-  
2016.gpg] \  
https://download.virtualbox.org/virtualbox/debian bookworm contrib" | \  
sudo tee /etc/apt/sources.list.d/virtualbox.list
```

```
sudo apt update sudo apt  
install virtualbox-7.1
```

##### Creación de la Máquina Virtual:

- **SO invitado:** Ubuntu Server 22.04
- **RAM:** 4 GB
- **CPU:** 2 núcleos
- **Disco:** 25 GB
- **Red:** Modo Bridge

##### Instalación de Docker en Ubuntu Server

```
sudo apt update sudo apt  
install docker.io sudo  
systemctl enable docker sudo  
systemctl start docker sudo  
systemctl status docker
```

##### Uso de Docker para levantar Apache en un contenedor

```
docker search httpd docker pull httpd  
docker run -dit --name apache-docker -p 8080:80 httpd
```

## Prueba del servidor

Desde el navegador del host:

`http://192.168.10.236:8080/`

*La IP se obtiene con:*

`ip a`

---

## 4. Metodología Utilizada

- Investigación de documentación oficial.
  - Instalación de VirtualBox en Debian 12.
  - Creación de una VM con Ubuntu Server y red en modo puente.
  - Instalación de Docker como herramienta moderna de despliegue.
  - Ejecución de un servidor web dentro de un contenedor.
  - Verificación del correcto funcionamiento accediendo desde el host.
- 

## 5. Resultados Obtenidos

- VirtualBox quedó correctamente instalado y funcional.
  - La VM de Ubuntu Server se configuró adecuadamente.
  - Docker se ejecutó sin inconvenientes y el contenedor con Apache respondió correctamente.
  - Desde el navegador del host, se accedió exitosamente al servidor web ejecutado en la VM a través de Docker.
  - Se observaron ventajas en el uso de contenedores frente a instalaciones directas.
- 

## 6. Conclusiones

Este trabajo me permitió comprender y aplicar conceptos clave sobre virtualización y contenedores. Utilizar Docker dentro de una VM es una excelente forma de simular entornos productivos modernos. Además, aprendí a integrar tecnologías (VirtualBox + Docker) y configurar servicios de red que permitan la interacción entre host e invitado, lo cual es útil tanto en desarrollo como en administración de sistemas.

---

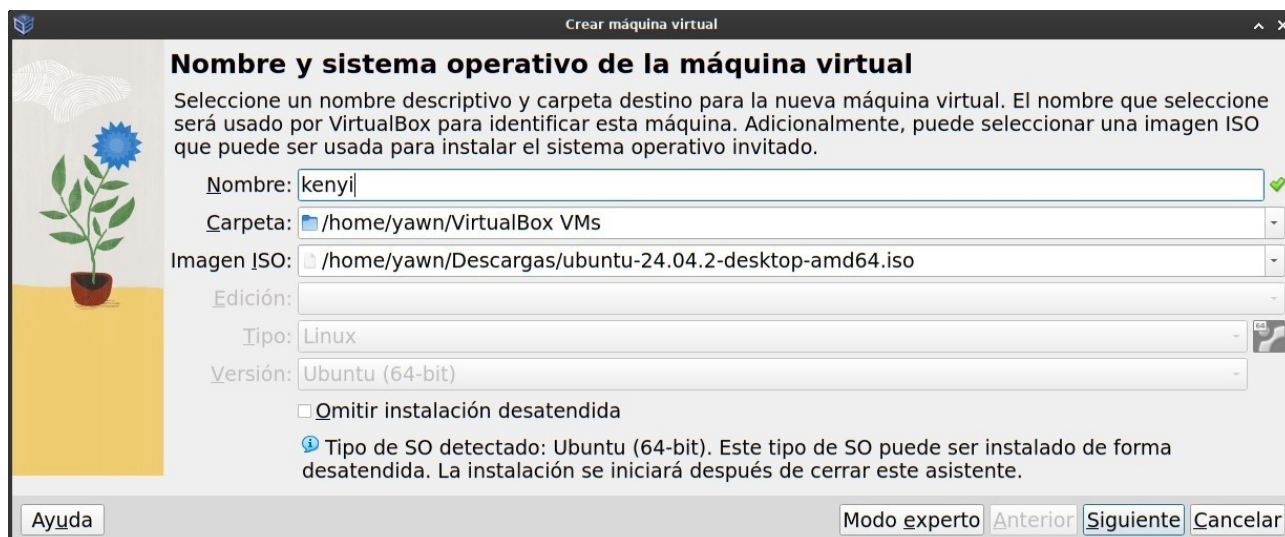
## 7. Bibliografía

- Docker Docs (instalación en Ubuntu):

<https://docs.docker.com/engine/install/ubuntu/>

- VirtualBox:  
<https://www.virtualbox.org>
- Ubuntu Server:  
<https://ubuntu.com>

## 8. Anexos



**Crear máquina virtual**

**Nombre y sistema operativo de la máquina virtual**

Seleccione un nombre descriptivo y carpeta destino para la nueva máquina virtual. El nombre que seleccione será usado por VirtualBox para identificar esta máquina. Adicionalmente, puede seleccionar una imagen ISO que puede ser usada para instalar el sistema operativo invitado.

Nombre:

Carpeta:

Imagen ISO:

Edición:

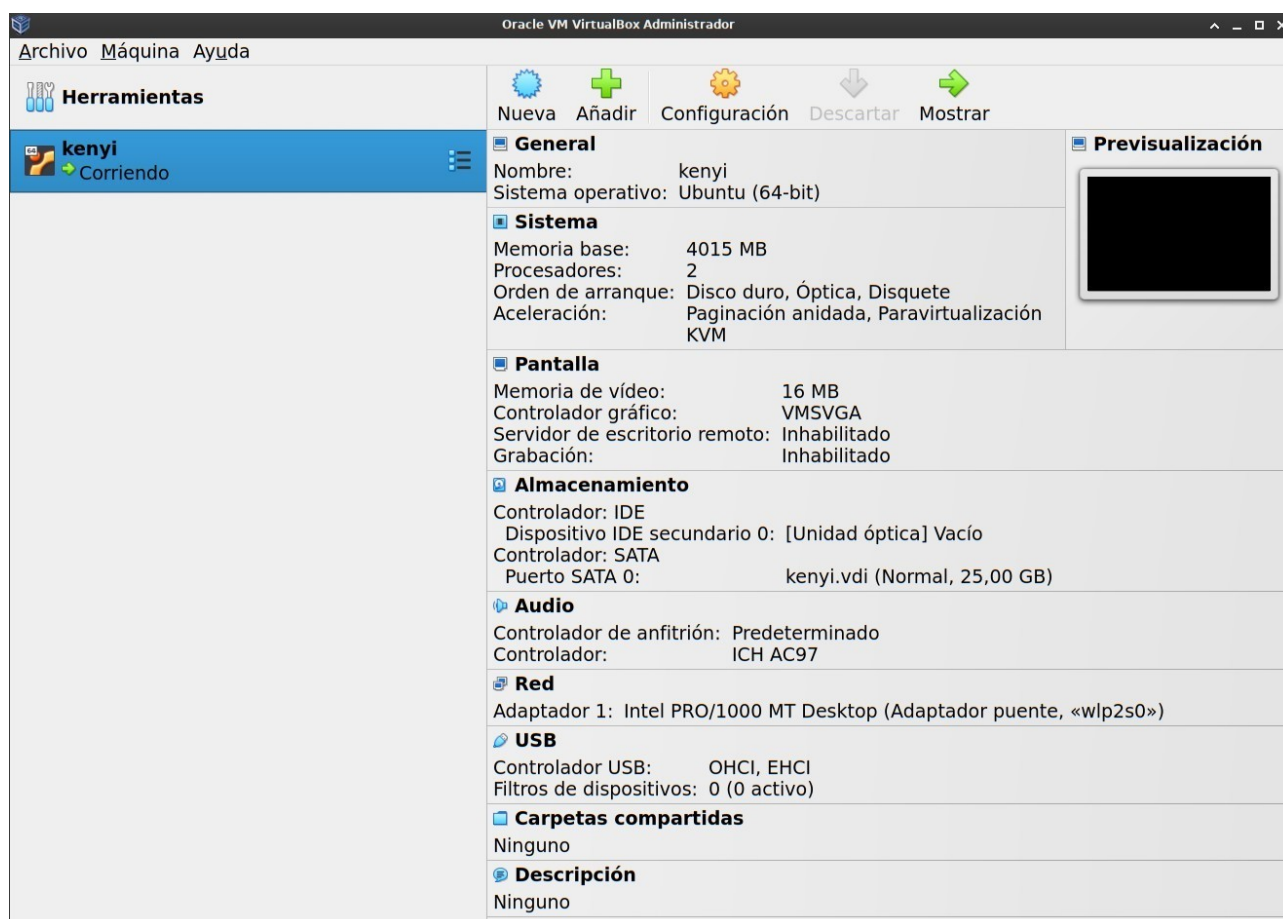
Tipo:

Versión:

☐ Omitir instalación desatendida

Tipo de SO detectado: Ubuntu (64-bit). Este tipo de SO puede ser instalado de forma desatendida. La instalación se iniciará después de cerrar este asistente.

Ayuda Modo experto Anterior Siguiente Cancelar



**Oracle VM VirtualBox Administrador**

Archivo Máquina Ayuda

Herramientas

kenyi Corriendo

Nueva Añadir Configuración Descartar Mostrar

**General**

Nombre: kenyi

Sistema operativo: Ubuntu (64-bit)

**Sistema**

Memoria base: 4015 MB

Procesadores: 2

Orden de arranque: Disco duro, Óptica, Disquete

Aceleración: Paginación anidada, Paravirtualización KVM

**Pantalla**

Memoria de vídeo: 16 MB

Controlador gráfico: VMSVGA

Servidor de escritorio remoto: Inhabilitado

Grabación: Inhabilitado

**Almacenamiento**

Controlador: IDE

Dispositivo IDE secundario 0: [Unidad óptica] Vacío

Controlador: SATA

Puerto SATA 0: kenyi.vdi (Normal, 25,00 GB)

**Audio**

Controlador de anfitrión: Predeterminado

Controlador: ICH AC97

**Red**

Adaptador 1: Intel PRO/1000 MT Desktop (Adaptador puente, «wlp2s0»)

**USB**

Controlador USB: OHCI, EHCI

Filtros de dispositivos: 0 (0 activo)

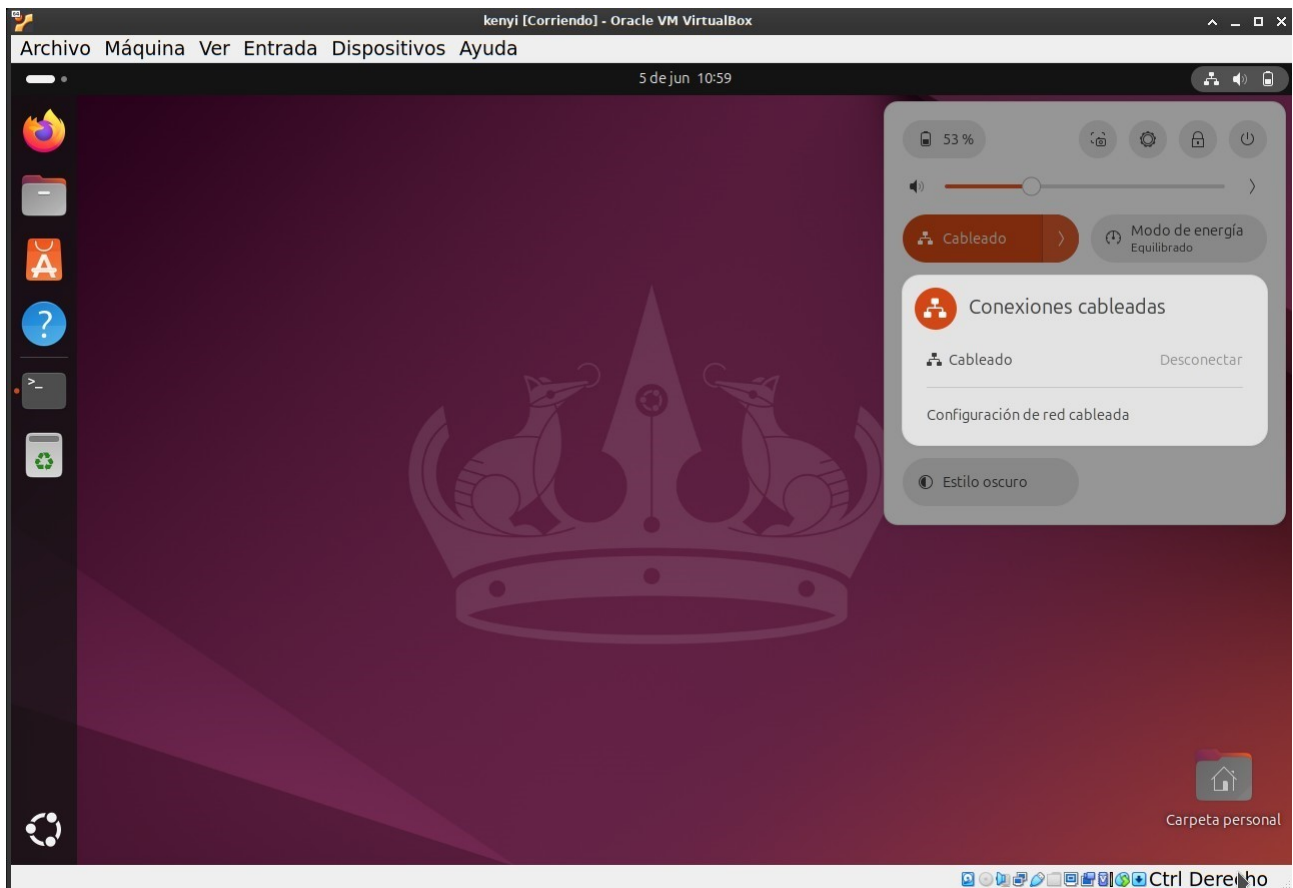
**Carpetas compartidas**

Ninguno

**Descripción**

Ninguno

**Previsualización**



No seguro192.168.10.253



# Apache2 Default Page

## Ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

### Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in** `/usr/share/doc/apache2/README.Debian.gz`. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2` and is managed using `systemd`, so to `start/stop` the service use `systemctl start apache2` and `systemctl stop apache2`, and use `systemctl status apache2` and `journalctl -u apache2` to check status. `system` and `apache2ctl` can also be used for service management if desired. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

### Document Roots

By default, Ubuntu does not allow access through the web browser to any file outside of those located in `/usr/share`.