

# Virtualization

## Tópicos de Informática para Automação

Mário Antunes

October 06, 2025

## Exercises

### Practical Lab: Exploring Virtualization & Emulation

This guide will walk you through different forms of virtualization, from lightweight emulation to full-blown server management. You will use **VirtualBox** (for Windows/macOS) or **QEMU** (for Linux) as your primary tool.

---

#### Part 1: Host Setup - Your Virtualization Tool

First, install the correct tool for your operating system.

##### For Windows & macOS Hosts: VirtualBox

###### 1. Download & Install:

- Go to the [VirtualBox downloads page](#) and download the installer for your OS.
- Also, download the **VirtualBox Extension Pack** from the same page.
- Run the main installer, accepting the defaults. On macOS, you must **Allow** the Oracle system extension in System Settings > Privacy & Security.
- Double-click the downloaded Extension Pack file to install it.

###### 2. How to Use VirtualBox:

- You'll use the graphical interface to create and manage VMs.
- Click "**New**" to start a wizard for a new VM.
- Modify settings by selecting a VM and clicking "**Settings**".

##### For Linux Hosts: QEMU

###### 1. Download & Install:

- QEMU and KVM (for hardware acceleration) are in most standard repositories. On Debian/Ubuntu, open a terminal and run:  

```
$ sudo apt update  
$ sudo apt install qemu-system-x86 qemu-system-i386 bridge-utils
```
- Add your user to the kvm group to run VMs without sudo. You must log out and back in for this to take effect.  

```
$ sudo usermod -a -G kvm $USER
```

###### 2. How to Use QEMU:

- QEMU is command-line driven. You'll create disks with `qemu-img` and launch VMs with `qemu-system-x86_64`.
- A typical launch command looks like this, with flags specifying resources:  
`$ qemu-system-x86_64 -m 1G -hda disk_image.qcow2 -cdrom installer.iso`

---

#### Part 2: Lightweight Emulation with FreeDOS

Here, we'll explore a simple, non-multitasking OS to understand basic machine emulation.

## 1. Download Resources:

- Download the **FreeDOS 1.3 Live CD** from the [official site](#). You'll need the FD14-LiveCD.zip file. Unzip it to get the .iso file.
- Download a classic shareware DOS game, like the first episode of **DOOM** (doom19s.zip), from a trusted [archive](#). Unzip it into a folder named doom.

## 2. Create the FreeDOS VM:

### • VirtualBox:

1. Click “**New**”. Name: FreeDOS, Type: Other, Version: DOS.
2. Memory: 64 MB.

3. Hard Disk: Create a new VDI, 128 MB, fixed size.

4. In **Settings > Storage**, select the empty CD drive, click the CD icon on the right, and **Choose a disk file...** to select your FD14LIVE.iso.

### • QEMU:

1. Create a 128M hard disk image.

```
$ qemu-img create -f qcow2 freedos.qcow2 500M
```

2. Launch the VM with the Live CD.

```
$ qemu-system-i386 -machine accel=kvm:tcg -m 128 -cpu host \
-k pt-pt -rtc base=localtime -device adlib -device sb16 \
-device cirrus-vga -display gtk -hda $DISK \
-cdrom /tmp/freedos/FD14LIVE.iso -boot d
```

## 3. Install and Set Up FreeDOS:

- Boot the VM and select “Install to harddisk”.
- Follow the on-screen prompts. It will ask to partition and format the drive (C:). Proceed with the default options.
- Once installation is complete, shut down the VM. In VirtualBox, remove the ISO from the virtual CD drive. In QEMU, remove the –cdrom and –boot d flags for the next launch.

## 4. Get The Game into the VM:

We will create a second CD image containing the game.

- **On Linux:** Qemu can make a FAT drive from a folder.

- **On Windows/macOS:** Use a free tool like AnyBurn to “Create ISO from files/folders”.

- **Attach the game ISO:**

- **VirtualBox:** Go to **Settings > Storage**. Click the “Add Optical Drive” icon on the IDE Controller, then add your doom.iso.

- **QEMU:** Add a second drive to your launch command: –drive file=fat:rw:/tmp/games/doom,form

- **Start FreeDOS.** Your game CD will likely appear as the D: drive. Type D: to switch to it, then run INSTALL.BAT or the game’s .EXE file.

## 5. Play the Game

---

## Part 3: Lightweight Virtualization with Alpine Linux Alpine

Let's install a modern, minimal Linux distribution that is the foundation for many containers.

## 1. Download Alpine:

- Go to the [Alpine Linux downloads page](#) and get the **STANDARD** version for your architecture (usually either x86\_64 or aarch64 ISO).

## 2. Install Alpine:

### • VirtualBox:

1. Create a new VM. Name: Alpine, Type: Linux, Version: Linux 2.6 / 3.x / 4.x (64-bit).

2. Memory: 1G. Hard Disk: 8 GB.

3. Attach the Alpine ISO in **Settings > Storage**.

### • QEMU:

```
$ qemu-img create -f qcow2 alpine.qcow2 8G
$ qemu-system-x86_64 -m 1G -hda alpine.qcow2 -cdrom path/to/alpine.iso -boot d
```

- Boot the VM and log in as `root` (no password). Run `setup-alpine` and follow the prompts. A "sys" install to `sda` is a good choice. When done, reboot and detach the ISO.

### 3. Explore Network Types:

- **NAT (Default):** With the default network setting, start the VM and check its IP address.

```
# Inside Alpine VM
$ ip addr show
```

You will see an IP like `10.0.2.15`. You can reach the internet (e.g., `ping google.com`), but you can't easily reach the VM from your host.

- **Bridge:** Shut down the VM.

- **VirtualBox:** Go to **Settings > Network**. Change **Attached to:** from NAT to Bridged Adapter.

- **QEMU:** Modify your launch command to use a bridge. This is more complex and system-dependent. Here is a sample code:

```
echo -e "Setup Bridge Interface"
sudo /sbin/ip link add virtbr0 type bridge
sudo /sbin/ip link set dev $INTERFACE master virtbr0
sudo /sbin/ip addr flush dev $INTERFACE
sudo /sbin/dhclient virtbr0
sudo /sbin/ip link set dev $INTERFACE up
sudo /sbin/ip link set dev virtbr0 up

echo -e "Start Alpine (BRIDGE)"
sudo qemu-system-x86_64 -machine accel=kvm:tcg -m 4G -smp 4 -cpu host \
-k pt-pt -rtc base=localtime -display gtk -hda $DISK \
-netdev bridge,id=net0,br=virtbr0 -device virtio-net-pci,netdev=net0

echo -e "Clean Bridge Interface"
sudo /sbin/ip link set virtbr0 down
sudo /sbin/ip link del virtbr0
sudo /sbin/dhclient $INTERFACE
```

- Start the VM again and run `ip addr show`. You should now see an IP address from your local home network (e.g., `192.168.1.123`).

### 4. Set Up a Web Server:

- Alpine uses `busybox httpd`. Install the package for extra features.

```
# Inside Alpine VM
$ doas apk add busybox-extras
```

- Create a directory for your web page.

```
$ doas mkdir -p /var/www/localhost/htdocs
```

- Create a simple HTML page.

```
$ echo '<h1>Hello from Alpine Linux!</h1>' > /var/www/localhost/htdocs/index.html
```

- Start the web server.

```
$ doas httpd -f -p 80 -h /var/www/localhost/htdocs
```

- From your **host machine's web browser**, navigate to the Alpine VM's IP address. You should see your message!

## Part 4: Server Management with Proxmox VE

Let's virtualize the virtualizer! We will install Proxmox, a bare-metal hypervisor, inside a VM.

**⚠ CAUTION: Nested Virtualization** You are about to run a hypervisor (Proxmox) inside another hypervisor (VirtualBox/QEMU). This is called **nested virtualization**. It is very demanding on your CPU and will be slow. This exercise is for demonstration purposes only.

#### 1. Download Proxmox:

- Go to the [Proxmox VE downloads page](#) and get the latest ISO Installer.

#### 2. Create the Proxmox VM: This VM needs more resources.

##### • VirtualBox:

1. Create a VM. Name: Proxmox, Type: Linux, Version: Debian (64-bit).
2. Memory: 4096 MB or more. Processors: 2 or more.
3. Hard Disk: 32 GB or more.
4. In **Settings > System > Processor**, check **Enable Nested VT-x/AMD-V**.
5. In **Settings > Network**, set it to **NAT**.
6. Configure port forwarding to redirect Host port 8006 to Guest port 8006.
7. Attach the Proxmox ISO.

##### • QEMU:

```
$ qemu-img create -f qcow2 proxmox.qcow2 32G
# The '-cpu host' flag is critical for passing through virtualization capabilities
$ qemu-system-x86_64 -m 4096 -smp 2 -cpu host -hda proxmox.qcow2 -cdrom proxmox.iso
```

#### 3. Install and Configure:

- Boot the VM and follow the Proxmox installation steps. It's a standard graphical installer.
- For networking, provide a static IP on your home network (e.g., 192.168.1.200).
- After installation, reboot and detach the ISO.

#### 4. Access the Web Portal:

- On the Proxmox console, it will display the URL to access. From your **host machine's web browser**, navigate to <https://localhost:8006>.
- You will see a security warning about the certificate; it's safe to proceed.
- Log in with **root** and the password you set during installation.

#### 5. Launch a Guest VM in Proxmox:

- Inside the Proxmox web portal, you can now create a new VM or container.
- Challenge: Try to create a new **Alpine Linux VM** inside Proxmox by uploading the Alpine ISO to the Proxmox server and following the web interface prompts.

---

## Part 5: Bonus Exercise - Emulating Android OS

The best way to emulate Android is using the official tools from Google.

#### 1. Download & Install Android Studio:

- Go to the [Android Studio download page](#) and get the installer for your OS.
- The installation process is a standard wizard. It will download many components, so it will take time.

#### 2. Use the AVD Manager:

- Open Android Studio. You don't need to create a project.
- From the welcome screen or the **Tools** menu, select **AVD Manager** (Android Virtual Device Manager).

#### 3. Create a Virtual Device:

- Click **Create Virtual Device...**.
- Choose a phone hardware profile (e.g., Pixel 7).
- Select a system image (a version of Android) to download.
- Give your AVD a name and click **Finish**.

#### 4. Launch the Emulator:

- Back in the AVD Manager, click the “Play” icon next to your new virtual device.
- A new window will open, booting a full, emulated Android operating system. Explore the interface, open apps, and use the browser, just like a real phone.