Integrantes: Joel Cuascota y Roberth Lara

Computación en la nube

Deber 2

**Experimento 1**

**Pasos para el cliente**

* Crear una carpeta para guardar el vagrantfile
* Abrimos la carpeta en powershell
* Hacemos un **vagrant init**
* Se crea un vagrant file, abrimos con un editor y colocamos el siguiente código:

Vagrant.configure("2") do |config|

# Usar Ubuntu 18.04 como base

config.vm.box = "ubuntu/bionic64"

# Configurar la red en modo puente para que la VM esté en la misma red que la máquina física

config.vm.network "public\_network", ip:”192.168.93.11”

# Configurar recursos de la VM

config.vm.provider "virtualbox" do |vb|

vb.memory = "1024" # Asignar 1GB de RAM

vb.cpus = 2 # Asignar 2 CPUs

end

# Instalar unzip, wrk y sus dependencias

config.vm.provision "shell", inline: <<-SHELL

sudo apt-get update

sudo apt-get install -y unzip build-essential libssl-dev git

git clone https://github.com/wg/wrk.git

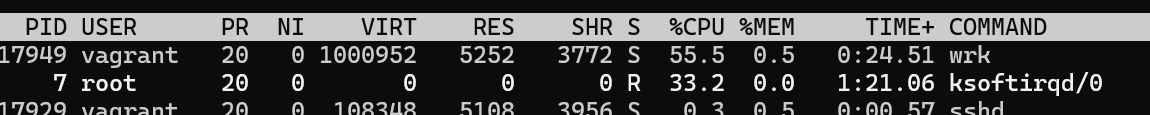
cd wrk && make

sudo cp wrk /usr/local/bin/

SHELL

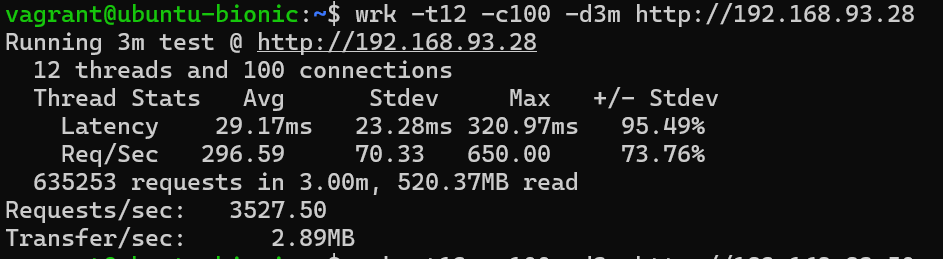
End

* Cerrar el documento
* Ejecutar “**vagrant up**”
* Ejecutar “**vagrant ssh**” para ingresar a la consola de la máquina virtual
* Ejecutamos **wrk -t12 -c100 -d3m** [**http://192.168.93.10**](http://192.168.93.10)



Pantalla de computadora con letras

Descripción generada automáticamente con confianza media



**Pasos para el servidor**

* **Crear un directorio:**

mkdir vm\_server

cd vm\_server

* **Crea el archivo Vagrantfile dentro de vm\_server:**

touch Vagrantfile

nano Vagrantfile

* Configurar el Vagrantfile para la máquina del servidor:

Vagrant.configure("2") do |config|

config.vm.box = "ubuntu/bionic64" # Utiliza Ubuntu 18.04 como sistema base

config.vm.network "public\_network", ip: "192.168.93.10" # Configurar IP pública para la VM del servidor

config.vm.provider "virtualbox" do |vb|

vb.memory = "1024" # Asigna 1 GB de RAM

vb.cpus = 2 # Asigna 2 CPUs

end

config.vm.provision "shell", inline: <<-SHELL

sudo apt-get update

sudo apt-get install -y nginx # Instalar NGINX

echo "<h1>Bienvenido al Servidor VM</h1>" | sudo tee /var/www/html/index.html # Página de bienvenida

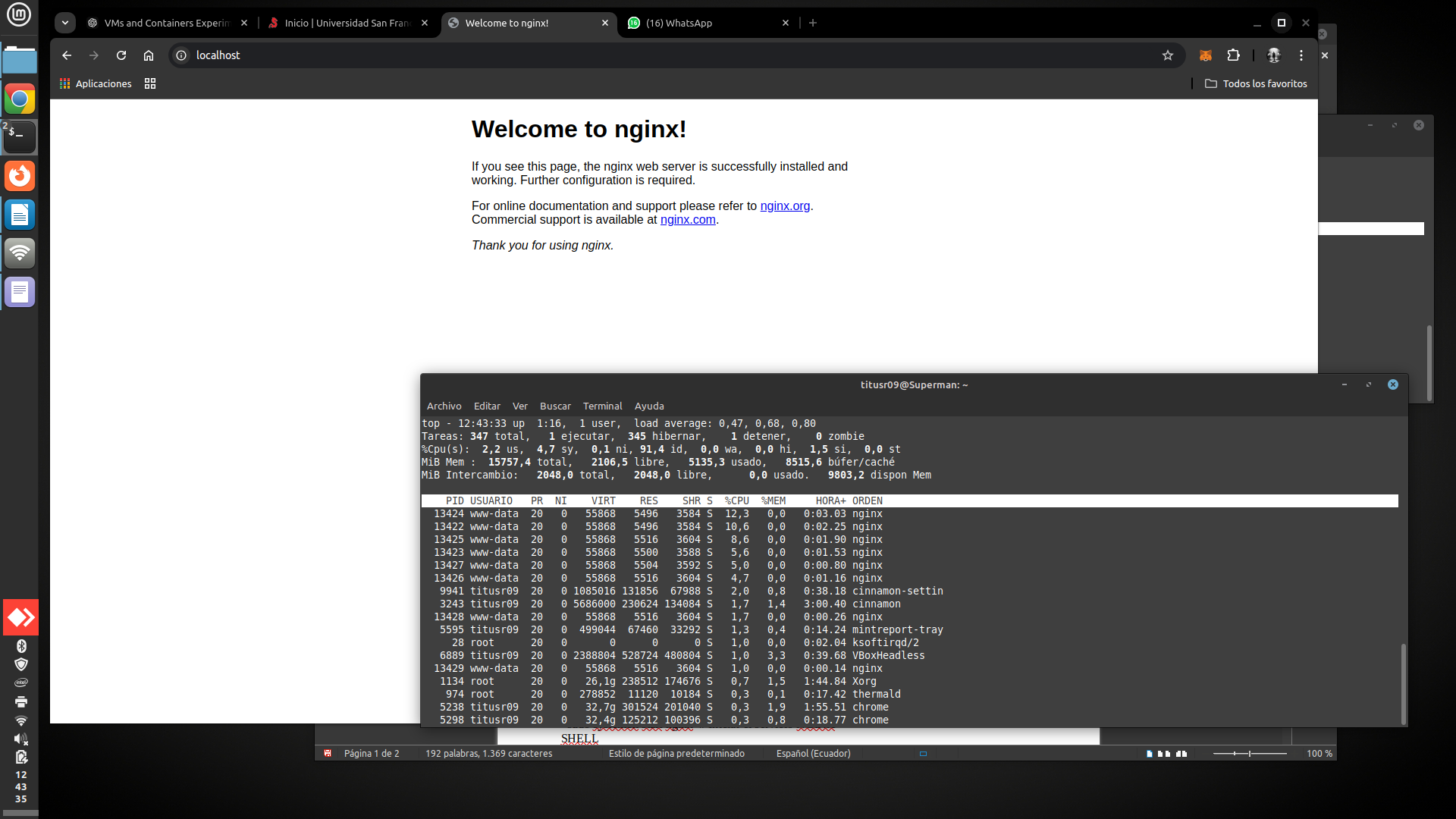
sudo systemctl start nginx # Iniciar el servicio NGINX

SHELL

end

* Levantar la VM del servidor:

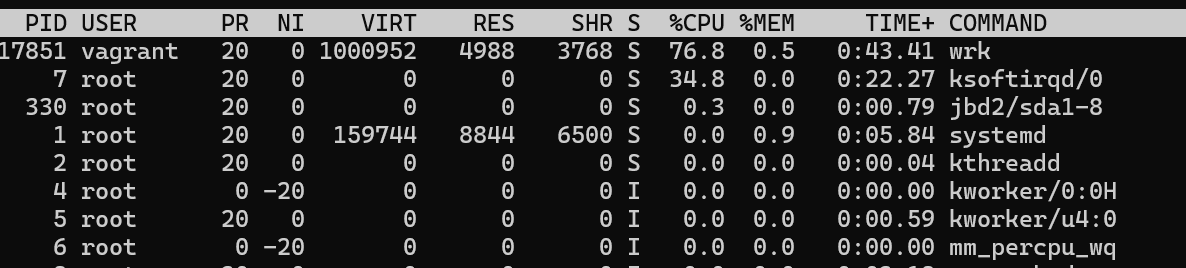
vagrant up



**Experimento 2**

**Pasos para el cliente**

* Son los mismos pasos, no hay cambios, a excepción que al momento de hacer wrk en la VM debemos colocar la ip del Docker, en este caso  **http://192.168.70.1**

****

**Pantalla de computadora con letras

Descripción generada automáticamente con confianza baja**

**Pantalla de computadora con letras

Descripción generada automáticamente con confianza media**

**Texto

Descripción generada automáticamente**

**Pasos para el servidor**

* **Activar los servicios de docker**

sudo systemctl start docker

* **Crear un directorio**

mkdir docker\_server

cd docker\_server

* **Creamos la carpeta del docker**

touch Dockerfile

* **Configuramos el Dockerfile**

FROM nginx:latest

RUN echo "<h1>Servidor NGINX en Docker</h1>" > /usr/share/nginx/html/index.html

EXPOSE 80

* **Construimos la imagen**

docker build -t my-nginx-server .

* **Ejecutamos el contenedor NGINX**

**Captura de pantalla de computadora

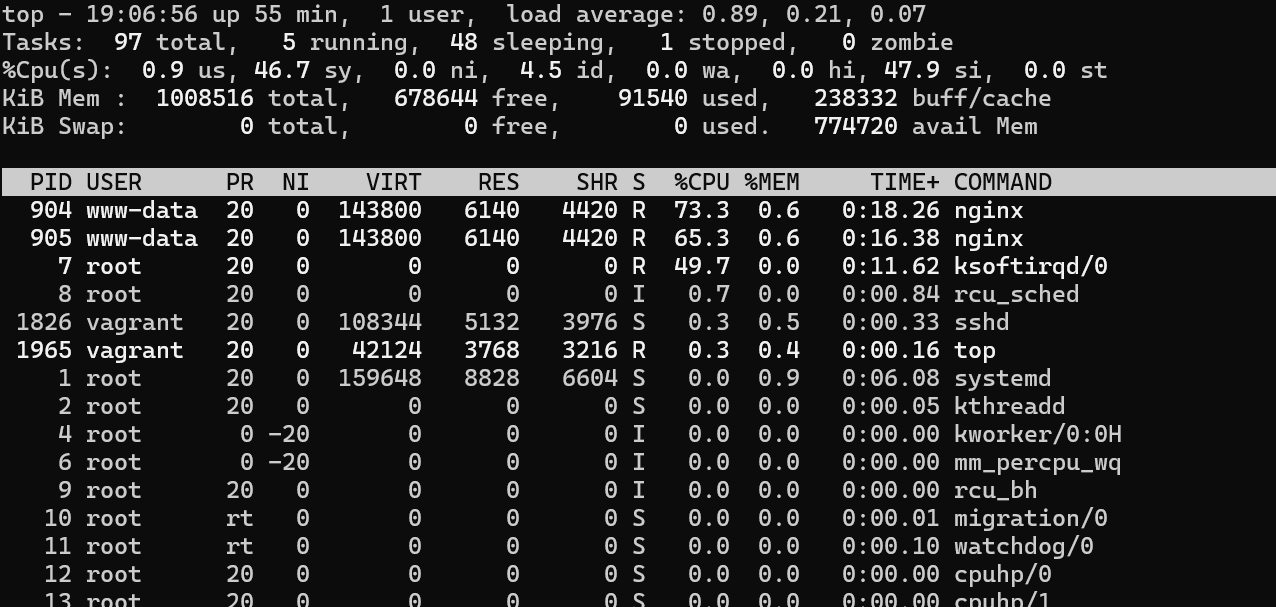
Descripción generada automáticamente**docker run -d -p 80:80 --name nginx-server my-nginx-server

**Análisis del servidor Vagrant y el servidor Docker**

El contenedor es más eficiente en términos de uso de CPU y memoria. Al no tener que emular hardware ni ejecutar sistemas operativos completos, el overhead (costo adicional) es mínimo, lo que permite un uso más eficiente de los recursos del host.

La máquina virtual ofrece un mayor nivel de aislamiento y flexibilidad en términos de ejecutar diferentes sistemas operativos, pero tiene un coste mayor en el uso de CPU y memoria debido a la virtualización completa.

Esta es una imagen del consumo del servidor vagrant con ngnix



Este es una imagen del consumo del servidor Docker

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

Aquí podemos apreciar lo antes mencionado