

Entorno de testing y pruebas

1. Instalamos VirtualBox y Vagrant con las configuraciones por defecto, disponibles en:
 - a. <https://www.virtualbox.org>
 - b. <https://www.vagrantup.com>
2. Obtenemos descargándolo manualmente el fichero Vagrantfile del repositorio, lo dejamos en una carpeta

<https://github.com/desarrollo-seguro/introduccion-ds>

3. Realizamos un primer arranque:

vagrant up

El primer arranque tarda en descargar la máquina y también tarda en configurar la interfaz de usuario y se quedará bloqueado en el punto que dice:

default: Setting up gdm3 (3.36.3-0ubuntu0.20.04.4) ...

lo paramos con Control+C, volvemos a la línea de comandos con un Enter.

Paramos la máquina con

vagrant halt

si se queda en ejecución se puede parar con la interfaz gráfica de virtualbox.

4. Arrancamos nuevamente la máquina virtual

vagrant up

5. Tras llegar a la pantalla de login:
 - a. Accedemos con PASSWORD:

vagrant

- b. Desde “Activities” abrimos “Settings” -> “Date & Time”.
 - i. Unlock y establecer “Time Zone” CEST (Madrid).
 - ii. Cerrar la ventana de “Time Zone”

- c. “Region & Language”
 - i. Añadir en input Sources Spanish
 - ii. Eliminar English

- d. Cerrar Settings

6. Abrimos un terminal
 - a. Completamos la instalación, al lanzar el comando en la interfaz gráfica aceptamos el valor por defecto

sudo dpkg --configure -a

- b. Nos colocamos en la carpeta compartida entre la máquina virtual entre la máquina física y la virtual

cd /vagrant

Bloque 1 – Introducción al desarrollo seguro

- c. Clonamos el proyecto de github

git clone https://github.com/desarrollo-seguro/introduccion-ds

- d. Entramos en la carpeta introduccion-ds

cd introduccion-ds

- e. Entramos en la carpeta scripts

cd scripts

- f. Comprobamos que la ruta actual es /vagrant/introducción-ds/scripts y que hay varios ficheros

ll

- g. Vemos el contenido del fichero install-varios

cat instalar-varios

- h. Lo ejecutamos

./instalar-varios

- i. Realizamos la misma operación con instalar-firefox, y comprobamos que tras la instalación tenemos disponible desde “Activities” el navegador

./instalar-firefox

- j. Realizamos la instalación de Docker

./instalar-docker

- k. Cerramos el terminal

7. Mediante vagrant paramos la máquina

vagrant halt

8. Volvemos a arrancar

vagrant up

9. Volvemos a un terminal

- a. Volvemos a la carpeta donde nos encontrábamos

cd /vagrant/introducción-ds/scripts/

- b. Probamos la instalación de Docker

docker --help

docker images

docker ps

docker ps -a

Bloque 1 – Introducción al desarrollo seguro

- c. Descargamos el container “hello-world”

docker pull hello-world

- d. Comprobamos que hay una nueva imagen y no hay containers en ejecución

docker images

- e. Ejecutamos el container “hello-world”

docker run hello-world

- f. Comprobamos que no hay container en ejecución, pero que lo hubo

docker ps

docker ps -a

- g. Descargamos las imágenes de las aplicaciones

./instalar-imagenes

- h. Comprobamos que están presentes

docker images

- 10. Atención los siguientes pasos van a levantar aplicaciones web con agujeros de seguridad, en primer lugar webgoat:

- a. **DESACTIVAMOS LA CONEXIÓN DE RED DE LA MAQUINA VIRTUAL**

- b. **Compramos que no hay conectividad a internet**

- i. Ejecutando un ping:

ping 8.8.8.8

- ii. Comprobamos que recibimos el mensaje:

ping: connect: **Network is unreachable**

- c. Vamos a lanzar y probar la aplicación webgoat

- i. Lanzamos y veremos que deja bloqueada la consola con las trazas de la aplicación

./webgoat

- d. Abrimos el browser y accedemos a la aplicación en:

- i. <http://localhost:8080/WebGoat>

- e. Cuando terminemos, podemos parar la aplicación desde un segundo terminal

- i. Comprobamos el “containerId” o el “Name del container” indistintamente

docker ps

- ii. Detenemos el container

docker stop “containerId”

Bloque 1 – Introducción al desarrollo seguro

- f. Verificamos de nuevo que no queden containers en ejecución:

docker ps

- g. Restablecemos la conexión de red, ya que es necesaria para poder cerrar posteriormente la máquina con normalidad de cara al siguiente arranque.

- 11. Juice Shop igualmente presenta agujeros de seguridad, por lo que por el mismo motivo lo haremos sin conexión

- a. **DESACTIVAMOS LA CONEXIÓN DE RED DE LA MAQUINA VIRTUAL**

- b. **Compramos que no hay conectividad a internet**

- i. Ejecutando un ping:

ping 8.8.8.8

- ii. Comprobamos que recibimos el mensaje:

ping: connect: **Network is unreachable**

- c. Vamos a lanzar y probar la aplicación webgoat

- i. Lanzamos y veremos que deja bloqueada la consola con las trazas de la aplicación

./juiceshop

- d. Abrimos el browser y accedemos a la aplicación en:

- i. <http://localhost:3000>

- e. Cuando terminemos, podemos parar la aplicación desde un segundo terminal

- i. Comprobamos el "containerId" o el "Name del container" indistintamente

docker ps

- ii. Detenemos el container

docker stop "containerId"

- f. Verificamos de nuevo que no queden containers en ejecución:

docker ps

- g. Restablecemos la conexión de red, ya que es necesaria para poder cerrar posteriormente la máquina con normalidad de cara al siguiente arranque.

- 12. Entorno de testing

- a. Instalamos node

./instalar-node

- b. Verificamos la disponibilidad y versiones de node y npm

node --version

npm --version

- c. Instalamos cypress

Bloque 1 – Introducción al desarrollo seguro

`./instalar-cypress`

- d. Instalamos VSCode

`./instalar-chrome`

- e. Instalamos VSCode

`./instalar-vscode`

- f. Creamos en una carpeta de la máquina virtual un nuevo proyecto node con los valores por defecto (pero no en /vagrant)

`npm init`

`npm install cypress --save-dev`