Práctica Linux Containers (LXD)

Elaborado por: Prof. Oscar Mondragón

1. Objetivos

1.1. Comprender el funcionamiento de Linux Containers
1.2. Instalar y configurar el demonio LXD

2. Herramientas a utilizar

- Máquinas virtuales Ubuntu 20.04 en Vagrant
- Linux Containers - LXD

3. Desarrollo de la Práctica

3.1. Configuración de Vagrant

Esta práctica la desarrollaremos usando boxes de Ubuntu 20.04 en Vagrant. El Vagrantfile que usaremos es el que se configuró al inicio del curso:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

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

 if Vagrant.has_plugin? "vagrant-vbguest"
  config.vbguest.no_install  = true
  config.vbguest.auto_update = false
  config.vbguest.no_remote   = true
 end

 config.vm.define :clienteUbuntu do |clienteUbuntu|
  clienteUbuntu.vm.box = "bento/ubuntu-20.04"
  clienteUbuntu.vm.network :private_network, ip: "192.168.100.2"
  clienteUbuntu.vm.hostname = "clienteUbuntu"
 end

 config.vm.define :servidorUbuntu do |servidorUbuntu|
```

```
  servidorUbuntu.vm.box = "bento/ubuntu-20.04"
  servidorUbuntu.vm.network :private_network, ip: "192.168.100.3"
  servidorUbuntu.vm.hostname = "servidorUbuntu"
 end
end
```

3.3. Instalación y utilización de LXD

Instalar LXD

La manera más simple de probar LXD es usando su herramienta de línea de comandos.

Abre una sesión SSH con servidorUbuntu

```
vagrant ssh servidorUbuntu
```

Para instalar LXD en Ubuntu ejecuta (puede tardar varios minutos):

```
sudo apt-get install lxd -y
```

Escoge LXD 4.0

Luego, ejecuta el siguiente comando para loguearte en el nuevo grupo creado:

```
newgrp lxd
```

Inicia lxd

```
lxd init --auto
```

Usar imágenes desde un servidor remoto

Desde este punto ya podrás interactuar con el demonio LXD instalado. Se puede, por ejemplo, experimentar con servidores remotos, agregando el servidor LXD público en https://images.linuxcontainers.org:8443

Ese servidor es de sólo lectura de imágenes, por lo que todo lo que puedes hacer con él es listar imágenes, copiarlas o iniciar contenedores.

Tendrás que hacer lo siguiente para: añadir el servidor*:

```
lxc remote add images images.linuxcontainers.org
```

(*Generalmente el servidor images.linuxcontainers.org ya viene instalado por lo que le generará un mensaje diciendo que ya existe.

Mi primer contenedor

LXD es basado en imágenes, sin embargo, por defecto ninguna imagen se carga localmente en el almacén de imágenes como se puede ver con:

```
lxc image list
```

LXD sabe acerca de 3 servidores de imagen predeterminados:

- ubuntu: (imágenes estables de Ubuntu)
- ubuntu-daily: (imágenes diarias de Ubuntu)
- images: (otras distribuciones)

Las imágenes estables de Ubuntu pueden ser listadas con:

```
lxc image list ubuntu: | less
```

Para lanzar un primer contenedor llamado "first" usando la imagen de Ubuntu 20.04, utiliza:

```
lxc launch ubuntu:20.04 first
```

Tu nuevo contenedor ahora será visible en:

```
lxc list
```

Los detalles del estado de ejecución y la configuración se pueden consultar con:

```
lxc info first
lxc config show first
```

Limitando recursos

Por defecto, tu contenedor no tiene limitación de recursos y hereda todos los recursos disponibles de su entorno principal (host). Puedes confirmarlo con:

```
free -m
lxc exec first -- free -m
```

Para aplicar un límite de memoria a tu contenedor, haz lo siguiente:

```
lxc config set first limits.memory 64MB
```

Y confirma que se ha aplicado con:

```
lxc exec first -- free -m
```

Snapshots

LXD es compatible con snapshots y restauración de contenedores.

Antes de hacer una instantánea, permite realizar algunos cambios en el contenedor, por ejemplo, instalando un paquete:

```
lxc exec first -- apt-get install net-tools -y
```

Entremos a la consola del contenedor y verifiquemos su IP

```
lxc exec first -- bash
```

```
root@first:~# ifconfig
```

Ahora que el contenedor está modificado, hagamos una instantánea llamada "clean":

```
lxc snapshot first clean
```

Dañemos nuestro contenedor de alguna forma:

```
lxc exec first -- rm -Rf /etc /usr
```

Confirmemos el daño de la siguiente manera (tratando de entrar a la consola del contenedor):

```
lxc exec first -- bash
```

Restauremos todo al estado del snapshot que realizamos:

```
lxc restore first clean
```

Confirmemos que todo está de vuelta a su estado normal y luego salimos del contenedor (exit):

```
lxc exec first -- bash
```

Sal de la consola del contenedor con exit:

```
#exit
```

Para listar los snapshots del contenedor first ejecuta:

```
lxc info first --verbose
```

Investiga: Qué efecto tiene la bandera –stateful en la creación de un snapshot.

Eliminación de una imagen

El comando para eliminar una imagen del repositorio local es*:

```
lxc image delete <alias or fingerprint>
```

*Por ahora no ejecutaremos este comando

El directorio de imágenes se encuentra en var/lib/lxd/image o /var/snap/lxd/common/lxd/images y es manejado internamente por LXD. Se recomienda no manipular directamente este directorio ya que dañará la instalación de LXD. Cuando ejecutes el comando lxc image delete, LXD removerá los archivos de ese directorio.

Creación de imágenes

Como probablemente has notado, LXD está basado en imágenes; es decir, todos los contenedores deben crearse a partir de una copia de un contenedor existente o de una imagen.

Puedes crear nuevas imágenes desde un contenedor existente o desde un snapshot de contenedor.

Para publicar el snapshot "clean" como una nueva imagen con un alias "clean-ubuntu", ejecuta:

```
lxc publish first/clean --alias clean-ubuntu
```

Verifícala con:

```
lxc image list
```

En ese momento no vamos a necesitar más nuestro contenedor "first", así que simplemente lo eliminamos con:

```
lxc stop first
lxc delete first
```

Y por último, podemos iniciar un nuevo contenedor a partir de nuestra imagen con:

```
lxc launch clean-ubuntu second
```

Acceso a archivos desde el contenedor

Para extraer un archivo del contenedor puedes utilizar el comando "lxc file pull":

```
lxc file pull second/etc/hosts .
```

Verifica que esté local:

```
ls
```

Mira su contenido

```
cat hosts
```

Añade una entrada:

```
echo "1.2.3.4 my-example" >> hosts
```

Y cópialo de nuevo a su lugar original:

```
lxc file push hosts second/etc/hosts
```

También puedes utilizar este mecanismo para acceder a los archivos de registro:

```
lxc file pull second/var/log/syslog - | less
```

No vamos a necesitar más ese contenedor, así que lo paramos y borramos así:

```
lxc delete --force second
```

Utilizar un servidor de imágenes remoto

La herramienta cliente lxc admite múltiples "remotes". Estos pueden ser servidores de imagen de sólo lectura u otros hosts LXD. LXC upstream ejecuta un servidor de este tipo en https://images.linuxcontainers.org que sirve un conjunto de imágenes generadas automáticamente para varias distribuciones de Linux.

Viene preagregado con LXD predeterminado, pero puedes quitarlo o cambiarlo si no lo deseas.

Puedes listar las imágenes disponibles con:

```
lxc image list images: | less
```

Y generar un nuevo contenedor Centos 8 con:

```
lxc launch images:centos/8 third
```

Confirma que es, de hecho, una imagen de Centos 8 con:

```
lxc exec third -- cat /etc/redhat-release
```

y elimínalo:

```
lxc delete -f third
```

Lista todos los “remotes” configurados con:

```
lxc remote list
```

3.4. Configuración de un Servidor Web Apache

Genera un nuevo contenedor Ubuntu 20.04 que usaremos para configurar un servidor Web Apache.

```
lxc launch ubuntu:20.04 web
```

Instala Apache

```
lxc exec web -- apt-get install apache2
```

Verifica el estado del servicio

```
lxc exec web -- systemctl status apache2
```

Verifica la existencia del index.html

```
lxc exec web -- ls /var/www/html
```

Por fuera del contenedor un nuevo index.html

```
<!DOCTYPE html>
<html>
```

```
<body>
<h1>Página de prueba</h1>
<p>Bienvenidos a mi contenedor LXD</p>
</body>
</html>
```

Reemplaza el index.html en el contenedor

```
lxc file push index.html web/var/www/html/index.html
```

Verifica su contenido

```
vagrant@servidorUbuntu:~$ lxc exec web -- cat /var/www/html/index.html
<!DOCTYPE html>
<html>
<body>
<h1>Página de prueba</h1>
<p>Bienvenidos a mi contenedor LXD</p>
</body>
</html>
```

Reinicia el servicio de apache

```
lxc exec web -- systemctl restart apache2
```

Verifica la ip del contenedor

```
vagrant@servidorUbuntu:~$ lxc info web
Name: web
```

```
Location: none
Remote: unix://
Architecture: x86_64
Created: 2020/08/14 15:49 UTC
Status: Running
Type: container
Profiles: default
Pid: 19010
Ips:
  eth0:	inet	10.24.66.4	veth87294e13
  eth0:	inet6	fd42:e654:eb7b:e925:216:3eff:fe5c:5c22	veth87294e13
  eth0:	inet6	fe80::216:3eff:fe5c:5c22	veth87294e13
  lo:	inet	127.0.0.1
  lo:	inet6	::1
```

Prueba el servicio con curl (usar la ip de su contenedor)

```
vagrant@servidorUbuntu:~$ curl 10.24.66.4
<!DOCTYPE html>
<html>
<body>
<h1>Página de prueba</h1>
<p>Bienvenidos a mi contenedor LXD</p>
</body>
</html>
```

Reenvío de puertos

Para poder hacer pruebas desde la red local (red del computador host) es necesario hacer reenvío de puertos desde el contenedor a la máquina virtual de vagrant, de la siguiente forma:

```
lxc config device add web myport80 proxy listen=tcp:192.168.100.3:5080 connect=tcp:127.0.0.1:80

Device myport80 added to web
```

Lo que hicimos aquí es:

1. Agregar un dispositivo proxy en el contenedor web, dándole un nombre arbitrario (myport80).

2. Configurar para escuchar en la interfaz de red (192.168.100.3) en el host (máquina Vagrant), puerto 5080.

3. Configurar para hacer conexiones al contenedor web en la interfaz 127.0.0.1 (localhost) en el puerto 80. En algunas guías sugieren usar localhost en lugar de 127.0.0.1 pero de esa forma no funciona en algunos casos, por favor usar 127.0.0.1 en su lugar.

Verificamos que la máquina vagrant esté escuchando en el puerto 5080

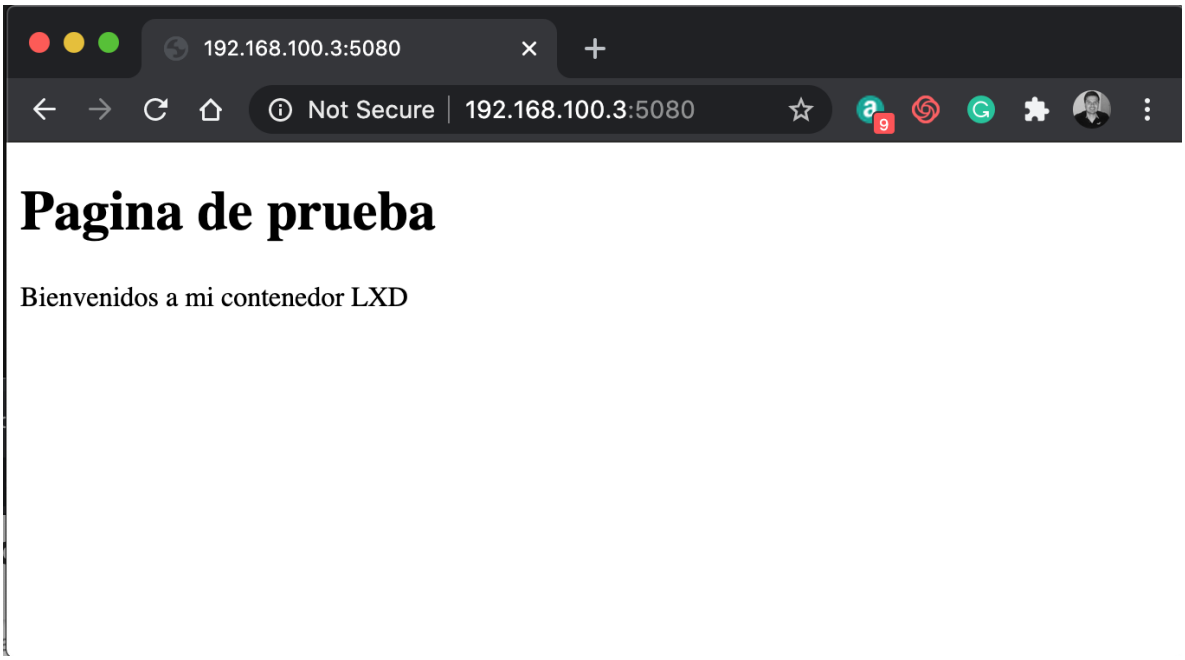
```
sudo lsof -i -n | grep 5080
```

También puedes usar el siguiente comando para ver los dispositivos creados:

```
lxc config device show web
```

Comprueba el servicio por fuera de la máquina vagrant

Comprueba el servicio desde el navegador usando la IP de la máquina virtual vagrant y el puerto de reenvío 5080.



3.5. Configuración Servidor SSH en el Contenedor

1. **Habilitar autenticación por password en SSH en el contenedor**

Es necesario habilitar autenticación por password en SSH en el contenedor para poder copiar luego la clave pública ssh.

```
$ sudo lxc exec web bash

# vim /etc/ssh/sshd_config
```

Busca el parámetro PasswordAuthentication y configúralo como yes:

```
PasswordAuthentication yes
```

Luego

```
# service sshd restart
```

Crea un usuario con usuario **user** y password **user**

```
# adduser user
```

Salir del contenedor

```
$ exit
```

Verifique que tiene acceso SSH con password

```
ssh user@<ip_del_contenedor>
```

2. **Crear par de claves SSH**

En la máquina virtual de servidorUbuntu crear una clave ssh

```
$ ssh-keygen
```

Dar enter al siguiente mensaje

```
Generating public/private rsa key pair.

Enter file in which to save the key (/home/username/.ssh/id_rsa):
```

Al solicitar un “paraphrase” dar enter (las dos veces)

```
Created directory '/home/username/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

El par de claves SSH (pública y privada) se generarán en /home/vagrant/.ssh/ con los nombres **id_rsa** (clave privada) y **id_rsa.pub** (clave publica)

```
Your identification has been saved in /home/username/.ssh/id_rsa.
Your public key has been saved in /home/username/.ssh/id_rsa.pub.
The key fingerprint is:
a9:49:2e:2a:5e:33:3e:a9:de:4e:77:11:58:b6:90:26 username@remote_host
The key's randomart image is:
+--[ RSA 2048]----+
|     ..o         |
|   E o= .        |
|    o. o         |
```

```
|       ..      |
|     ..S       |
|     o o.      |
|   =o.+.       |
|. =++..        |
|o=++.          |
+-----------------+
```

3. **Copiar clave pública en el contenedor**

```
vagrant@servidorUbuntu:~$ ssh-copy-id user@10.24.66.4

/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
"/home/vagrant/.ssh/id_rsa.pub"The authenticity of host '10.24.66.4 (10.24.66.4)' can't be
established.ECDSA key fingerprint is
SHA256:jpq7urUpWL6k3rHBgRlZEGMkfoedoAq292PsPL8uR/E.Are you sure you want to
continue connecting (yes/no/[fingerprint])?
```

Escribir **yes**

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are
already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to
install the new keys
user@10.24.66.4's password:
```

El anterior comando copiará la clave pública desde /home/vagrant/.ssh/id_rsa.pub en la máquina vagrant a /home/user/.ssh en el contenedor

10.24.66.4 corresponde a la IP del contenedor (modifíquela apropiadamente)

El password solicitado es el que se usó cuando se creó el usuario *user* en el paso 1.

4. **Iniciar sesión SSH en el contenedor usando la clave ssh**

```
ssh user@10.24.66.4
```

5. **Copiar archivos al contenedor**

```
scp archivo.txt user@<ip_contenedor>:~/
```

3.6. Ejercicio LXD

- Crea un contenedor 20.04 de 64 bits en la máquina Vagrant servidorUbuntu. El contenedor se deberá llamar **server.**
- Instala un servidor web en el contenedor (server) y verifica su funcionamiento desde la red local (red del host, ver punto 3.4)
- Instala SSH en el servidor (Punto 3.5)
- Inicia sesión SSH en el contenedor server desde la máquina Vagrant clienteUbuntu. Para esto, ten en cuenta:
    - Crear un nuevo usuario en el contenedor llamado **remoto** (adduser remoto).
    - Realizar el **reenvío de puertos** necesario: del puerto 22 del contenedor al puerto que escojas (por encima del 1023) de la maquina servidorUbuntu.
    - Para copiar la clave pública generada en clienteUbuntu al contenedor puedes usar un comando similar a este:

      ssh-copy-id -p <puerto_reenvio> remoto@192.168.100.3

      Donde puerto_reenvio es el puerto de la máquina servidorUbuntu al cual le estás haciendo reenvío de puertos desde el contenedor.
    - Iniciar sesión SSH remotamente. Usando un comando similar a este:

      ssh -p <puerto_reenvio> remoto@192.168.100.3
    - Transferir un archivo al servidor usando scp.

      scp -P <puerto_reenvio> archivo remoto@192.168.100.3:~/

      la -P es mayúscula

5. Entregables y evaluación

- Sustentación de la práctica

6. Referencias

- Practical LXC and LXD. Linux Containers for Virtualization and Orchestation. Senthil Kumaran S. Apress, 2017.
- Linux Containers: https://linuxcontainers.org/lxd/introduction/
- Servidor LXD demostrativo online: https://linuxcontainers.org/lxd/try-it/
- Port-forwarding: https://discuss.linuxcontainers.org/t/forward-port-80-and-443-from-wan-to-container/2042
- Creación clave SSH. https://www.digitalocean.com/community/tutorials/how-to-configure-ssh-key-based-authentication-on-a-linux-server
- Otra forma de copiar claves SSH (No usar para esta práctica). https://aapjeisbaas.nl/post/push-ssh-public-key-to-lxc-container/