LXD - CONTAINERS

Listar los Container y Virtual-Machine

```
$ lxc image list images:
```

Listar los Container y Virtual-Machine por filtro

```
$ lxc image list images:ubuntu
$ lxc image list images:ubuntu arch=amd64
```

Crear container

```
$ lxc launchg images:ubuntu/focal apache2
```

Vereficar que existen los conatiners

```
$ lxc list
```

Acceder al container

```
$ lxc exec apache2 bash
```

Ver los nucleos que ocupa el container

```
root@apache2:~# nproc
```

Ver la memoria RAM que ocuapa el container

```
root@apache2:~# free –mega
```

Crear container limitando CPU

```
$ lxc launch images:debian/11 debian11 -c limits.cpu=1
```

Crear container limitando CPU + memoria RAM

```
$ lxc launch images:debian/11 debian11-limits -c limits.cpu=1 -c limits.memory=128MB
```

Limitar CPU de un container en caliente (es decir que ya lo tengo creado y lo estoy editando)

```
$ lxc config set debian11-limits limits.cpu 2
```

Limitar CPU + menoria RAM en caliente (es decir que ya lo tengo creado y lo estoy editando)

```
$ lxc config set debian11-limits limits.memory 2GB
```

Redireccionar puertos

- Los siguientes comandos permiten redireccionar el acceso de un container a través del host.

```
$ lxc config device add <container_name> <name_config> proxy
listen=<tcp_udp>:<host>:<host_port> connect=<tcp_udp>:<container_ip>:<container_port>
```

container_name = 10.182.10.13:4444 → host = 0.0.0.0:10000

Ver las configuraciones que tiene nuestro container

```
$ lxc config show apache2
```

Copiar del host al conatiner un archivo

```
$ lxc file push archivo.zip apache2/home/ubuntu/imagenes/
```

Copiar del host al container un directorio

```
$ lxc file push -r directorio/ apache2/ubuntu/imagenes/
```

Copiar del container al host

```
$ lxc file pull apache2/home/ubuntu/imagenes/archivo.zip .
```

Requisitos

- Servidor Ubuntu 21.10
- Instalar lxd
- Para el segundo ejemplo vamos a necesitar instalar nodejs
- Vamos a crear una carpeta compartida entre el servidor y el container

Fuentes

- Para la instalación de lxc se siguieron los pasos del video Taller Creat tu Home Server con LXD – (Felix Apaza) del canal Nucleo Linux Bolivia.
- Para la aplicación webSocket se siguieron los pasos del video Socket.io | Curso Práctico de WebSockets, con Socket.io y Nodejs | Chat con Socket.io del canal Fazt.

Observaciones

➔ Cuando descarge el binario de nodejs lo instale en el servidor y el container, cuando agrege la variable de entorno al servidor me lo reconece, mientras que en el container no por lo que tengo que ejecutar el archivo .profile con el comando `$ . ~/.profile` , y recien puedo utilizar los comandos de nodejs.

➔ Cree el volumen porque no podia acceder directamente al container desde mi editor de texto, de esta forma solucione el problema de tener que copiar de mi equipo al container y probar mi codigo.

➔ Para que corra el proyecto debo ubicarme en la carpeta del proyecto dentro del container (/var/www/html/socket_chat) y ejecutar el comando `$ npm run dev` ,de lo contrario no va a correr. Al principio creia que por estar en ese directorio debia correr sin embargo no es así, el proyecto utiliza nodejs es por eso que debemos ejecutar el comando ya mencionado de esta manera se inicializa el servidor y habilita el puerto 3000 el cual se especifica en el archivo index.js.

Para el primer ejemplo vamos a utilizar al container apache2

1. Ingresamos a container apache2
   ```
   $ lxc exec apache2 bash
   ```

2. Actualizamos la lista y instalamos apache2
   ```
   root@apache2:~# apt update
   root@apache2:~# apt install apache2
   ```

3. Verificar que apache2 este levantado
   ```
   root@apache2:~# ss -nltp
   ```

4. Vemos que esta ocupando el puerto 80

5. Debemos obtener la ip de nuestro equipo (host) en mi caso seria
   ```
   192.168.0.115
   ```

6. Luego debemos obtener la ip del container apache2
   ```
   10.216.156.163
   ```

7. Como prueba podemos colocar la ip de apache2 en un navegador y veremos que no accede al servicio.

8. Configuramos la redireccion de puertos
   ```
   $ lxc config device add apache2 config1 proxy listen=tcp:0.0.0.0:8080
   connect=tcp:10.216.156.163:80
   ```

9. Ahora abrimos un navegador y colocamos nuestra ip 192.168.0.115:8080.

10. Lo que quiere decir esto es que nos esta redireccionando de 192.168.0.115:8080 al container 10.216.156.163:80 con el servicio de apache2

Para este segundo ejemplo vamos a utilizar apache2

1. Lo primero es descargar nodejs node-v14.18.1-linux-x64.tar.xz. Yo lo descarge desde mi equipo Ubuntu con interfas y lo pase por scp al servidor y del servidor al container. Para ejecutar el comando debo estar en Downloads.
   ```
   $ scp node-v14.18.1-linux-x64.tar.xz simon@192.168.0.115:/home/simon/node
   ```

2. Ahora vamos a copiar del host al container nos debemos ubicar en el directorio /home/simon/node.
   ```
   $ lxc file push node-v14.18.1-linux-x64.tar.xz apache2/home/ubuntu
   ```

3. Debemos seguir los pasos de instalación de How to install Node.js via binary archive on linux? De la siguiente dirección: https://github.com/nodejs/help/wiki/Installation

4. Debemos redireccionar el puerto. Como en nuestro archivo index.js especificamos que el servidor opere en el puerto 3000.
   ```
   $ lxc config device add apache2 node proxy listen=tcp:0.0.0.0:3030
   connect=tcp:10.216.156.163:3000
   ```

5. Luego debemos ver el video de la aplicación webSocket.

6. Y para probar nuestra aplicación abrinos un navegador y colocamos nuestra ip 192.168.0.115:3030. Claro antes de debemos iniciar el servicio con el comando `$ npm run dev`

Anexo

Crear carpeta compartida entre el servidor y el container.

1. Debemos crear una carpeta en el servidor /home/simon/proyecto que va estar compartida con /var/www/html

2. Debemos verificar UID del usuario
   ```
   $ id -u simon
   1000
   ```
   or
   ```
   $ cat /etc/passwd
   ```

3. El container debe estar detenido
   ```
   $ lxc stop apache2
   ```

4. Con el siguiente comando creamos la correspondencia entre el directorio local y el directorio */var/www/html* del contenedor:
   ```
   $ lxc config device add apache2 www disk \
   source=/home/simon/proyecto \
   path=/var/www/html
   Device www added to apache2
   ```

5. Y por último, con este otro comando establecemos la correspondencia tanto del *UID* como del *GID* de nuestro sistema operativo anfitrión, 1000, con el *UID* y el *GID* del sistema operativo huésped del contenedor, 33. Este valor es el que tiene asignado el usuario y grupo *www-data* que utiliza el servidor *web*.
   ```
   $ lxc config set apache2 raw.idmap “both 1000 33”
   ```

6. Ahora podemos comenzar a realizar nuestros proyectos utilizando nuestro editor de texto favorito, en este caso utilice VSCode ya que cuenta con acceso remoto a través de ssh. El cual especifico la carpeta en donde voy a trabajar (ssh simon@192.168.0.115:/home/simon/proyecto).