

**UNIVERSIDAD TECNOLÓGICA NACIONAL**  
**FACULTAD REGIONAL TUCUMÁN**



**Trabajo final**

**Virtualización y consolidación de  
servidores**

**2023**

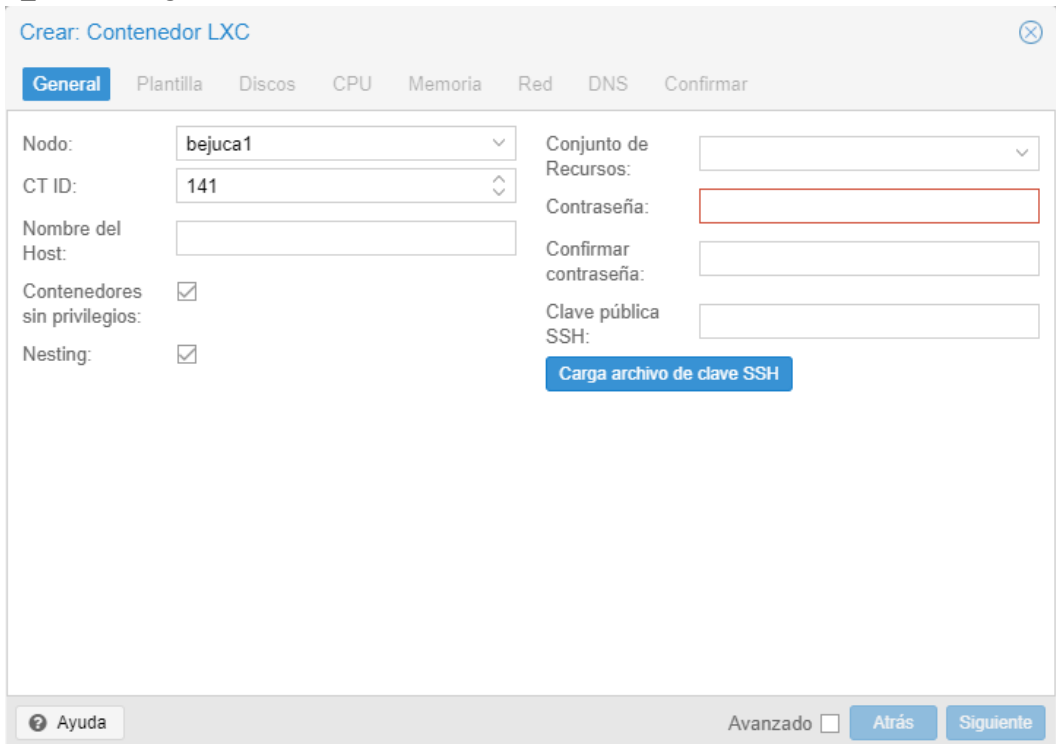
**Profesor: Ing. Luis María Carriles**

**Alumno: Javier Martín Roza**

**Legajo: 42274**

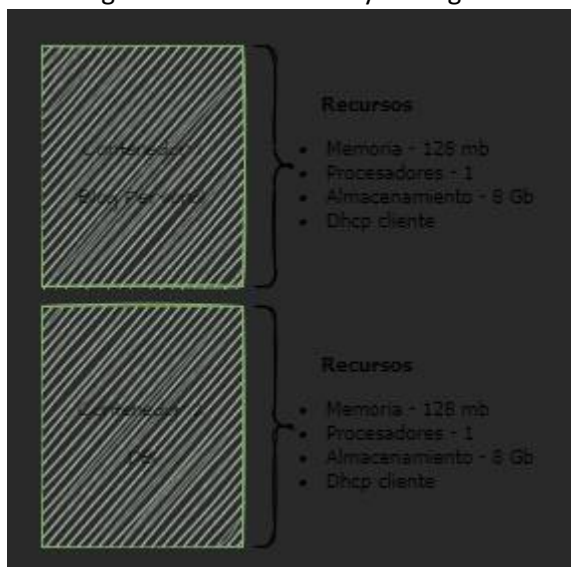
En este informe se detalla el procedimiento para la creación de los contenedores en la plataforma de proxmox y todo lo necesario para implementar una pagina web alojado en uno de ellos y a su vez este se conecta a otro contenedor donde esta alojada la base de datos.

1. En primer lugar, se solicitó la creación del usuario para ingresar a la plataforma proxmox. Con el usuario y contraseña provisto se ingreso a través del link <https://319e02b588a6.sn.mynetname.net:9991/>.
2. El siguiente paso fue la creación del factor de doble autenticación. En mi caso utilice TOTP. Descargando la aplicación en el celular, el servicio provee una clave temporal con lo que se podrá acceder como segunda medida de seguridad luego de la contraseña.
3. Una vez logueados en la plataforma se procede a crear los 2 contenedores en el nodo bejuca1. Ambos son creados con la plantilla de ubuntu-20.04-standard\_20.04-1\_amd64.tar.gz

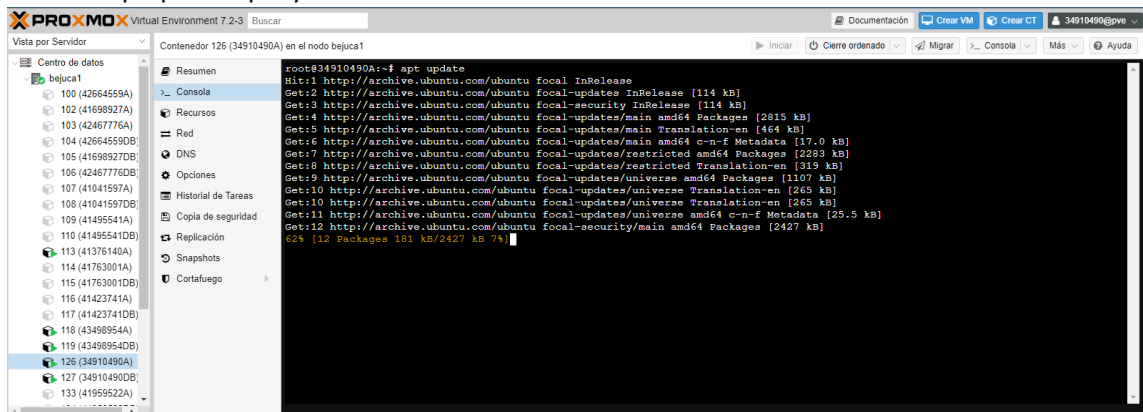


The screenshot shows the 'Crear: Contenedor LXC' window in Proxmox. The 'General' tab is selected. The 'Nodo' is set to 'bejuca1' and 'CT ID' is '141'. The 'Nombre del Host' field is empty. The 'Contenedores sin privilegios' checkbox is checked, and the 'Nesting' checkbox is also checked. On the right, the 'Conjunto de Recursos' dropdown is empty, and the 'Contraseña', 'Confirmar contraseña', and 'Clave pública SSH' fields are empty. A blue button labeled 'Carga archivo de clave SSH' is present. At the bottom, there is an 'Ayuda' button, an 'Avanzado' checkbox, and 'Atrás' and 'Siguiente' buttons.

Se configuran todos los datos y se asignan los recursos según el enunciado:



4. Comenzamos trabajando con el primer contenedor que será el que tendrá el front-end. Una vez logueados ejecutamos los comandos necesarios para poder tener instalados los componentes que nos harán falta para trabajar. se ejecuta “apt update” que nos permite actualizar los paquetes disponibles en los repositorios de soft del sistema, es decir verifica en los servidores si existen actualizaciones disponibles para los paquetes instalados o que podrían ser instalados. Solo actualiza la información sobre las versiones disponibles. A continuación ejecutamos “apt upgrade” para instalar las actualizaciones disponibles sobre los paquetes que ya tenemos instalados.

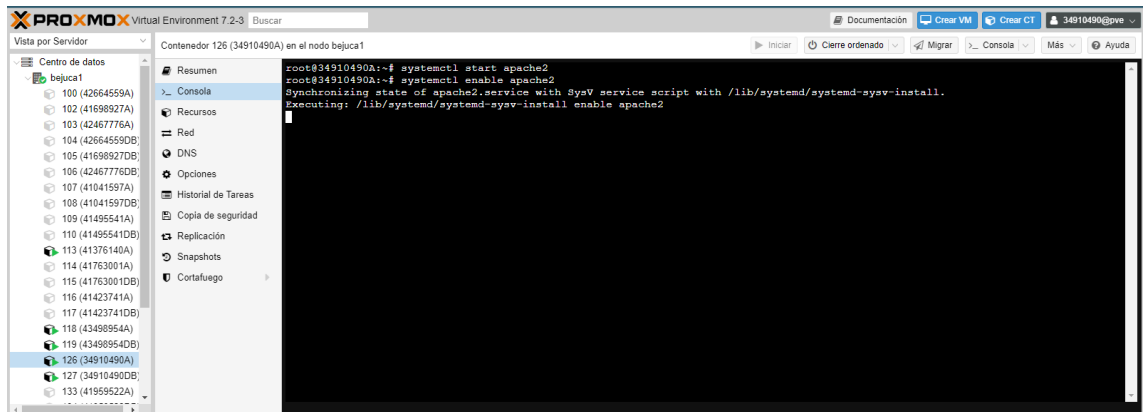


The screenshot shows the Proxmox Virtual Environment 7.2-3 interface. On the left, a sidebar lists various data centers and containers. The main window displays the terminal output for container 126 (34910490A) on node bejca1. The terminal shows the execution of the 'apt update' command, which updates the package lists from the Ubuntu repositories. The output includes the following lines:

```
root@34910490A:~# apt update
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [2815 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates/main Translation-en [464 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [17.0 kB]
Get:7 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [2283 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [319 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1107 kB]
Get:10 http://archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [265 kB]
Get:11 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [25.5 kB]
Get:12 http://archive.ubuntu.com/ubuntu focal-security/main amd64 Packages [2427 kB]
debconf: delaying package configuration, since apt-utils is not installed
OK:12 Packages 151 kB/2427 kB %
```

5. Se ejecuta el comando “apt install apache2”. Esto nos instalara en nuestro nodo el software apache, un soft de servidor web que recibe las requests de los visitantes de la pagina y les devuelve la información que solicitan en forma de página web. Esta a cargo básicamente de la conexión entre el servidor donde esta alojado el sitio y el dispositivo desde el cual se accede al mismo.

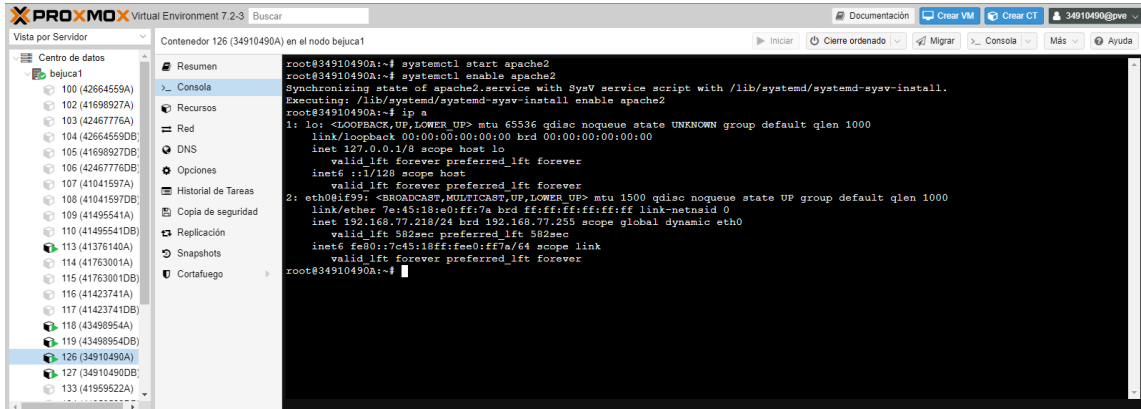
Una vez instalado se ejecutan los comandos “systemctl start apache2” y “systemctl enable apache2”. El primero inicia el servicio del servidor web apache. El segundo comando habilita el inicio automático del servicio de servidor web cuando se inicia el contenedor.



The screenshot shows the Proxmox Virtual Environment 7.2-3 interface. On the left, a sidebar lists various data centers and containers. The main window displays the terminal output for container 126 (34910490A) on node bejca1. The terminal shows the execution of the 'systemctl start apache2' and 'systemctl enable apache2' commands. The output includes the following lines:

```
root@34910490A:~# systemctl start apache2
root@34910490A:~# systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
```

6. Procedemos a verificar el numero de ip asignado por el servicio de dhcp al contenedor A:



Utilizamos el comando “ip a”. Vemos que nos arroja la ip 192.168.77.218/24

El puerto utilizado por apache en un SO Ubuntu es el puerto numero 80 para conexiones HTTP y el numero 43 para las conexiones de tipo HTTPS. En nuestro caso será HTTP.

7. A continuación se instalara un firewall para filtrar el tráfico de red entrante y saliente y proteger el sistema contra amenazas y ataques.

Instalaremos UFW. Se instala como el resto de los paquetes con el comando “apt install ufw”.

UFW es una interfaz de línea de comandos que simplifica la configuración y administración de firewall en sistemas basados en Debian.

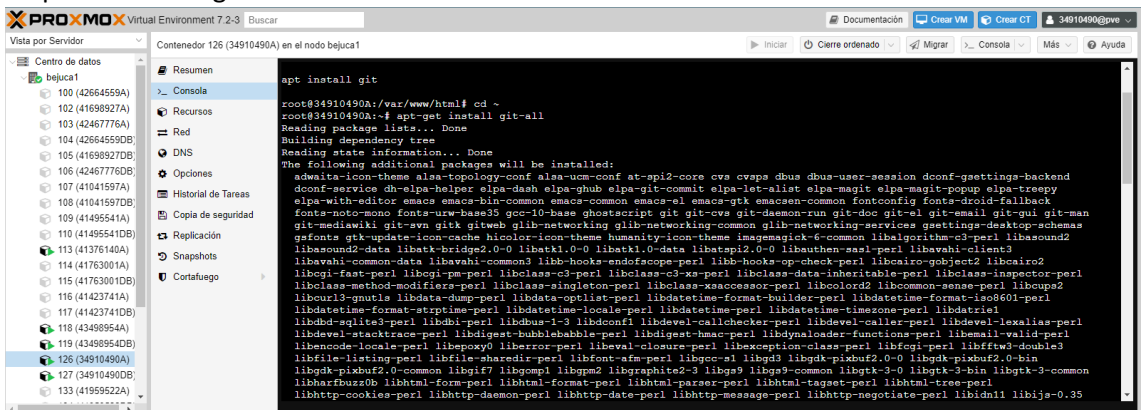
Una vez instalado se ejecuta el comando “ufw allow 80/tcp” para permitir el tráfico entrante en el puerto 80 utilizado por el protocolo TCP a través del firewall.

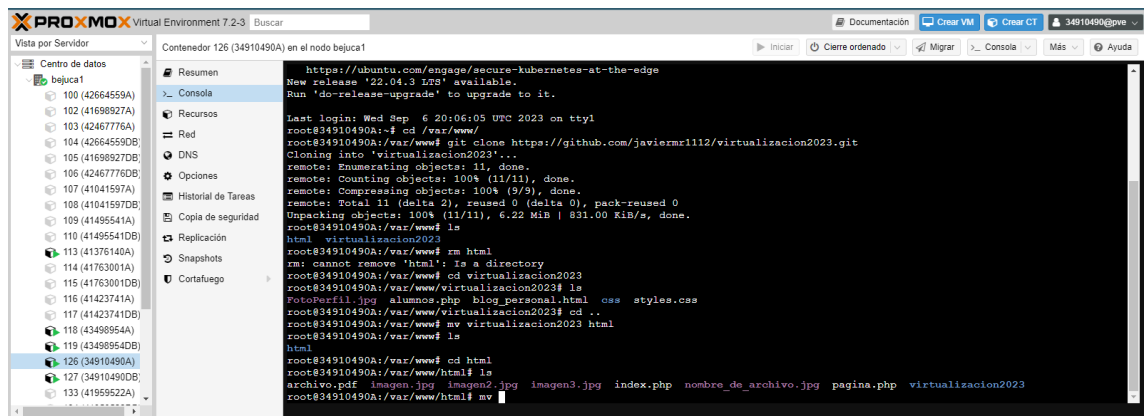
Por ultimo activamos el firewall mediante el comando “ufw enable”

8. La programación del blog se hizo externamente al contenedor y para llevarlo al mismo se utilizo github. Para poder trabajar con los repositorios de github se deben instalar los paquetes necesarios. Se ejecutaron los comandos “apt-get install git-all”

Una vez que lo tenemos instalado podemos clonar nuestro repositorio en la carpeta html generada por apache (/var/www/html). Todo lo que queramos que apache ponga a disposición de los visitantes del blog debe estar en esta carpeta.

Nuestro archivo traído de github puede tener el nombre que le hayamos puesto previamente pero cuando queremos mostrarlo como una pagina debemos modifícalo a index.php para que sea accedido desde la url que apunta al contenedor. Debe ser .php para que el archivo sea dinámico y su código se ejecute en el servidor antes de enviar la respuesta al navegador.

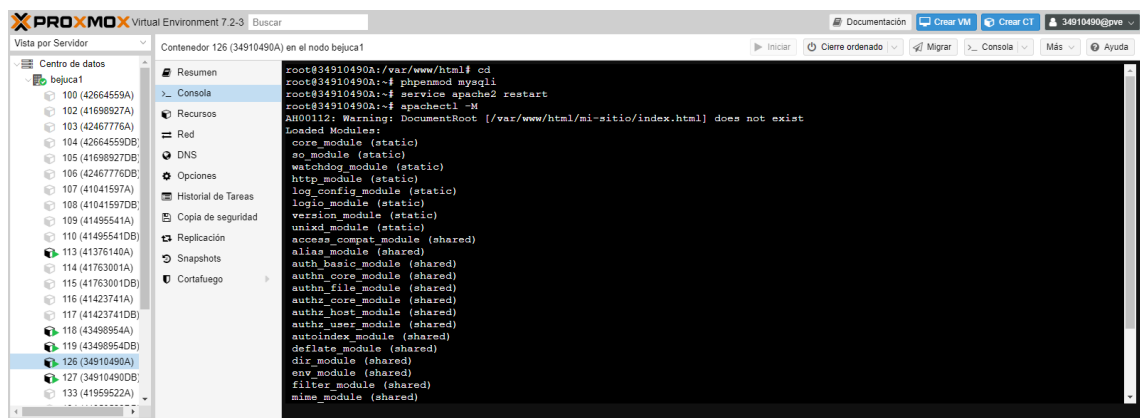
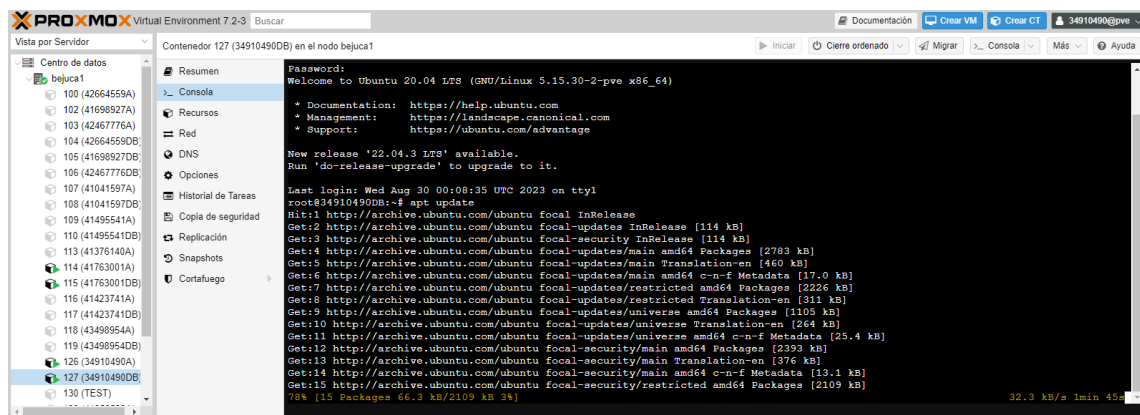




- 9.
10. Instalamos php con el comando “apt install php libapache2-mod-php php7.4-mysql”
11. Se habilita el paquete mysql mediante el comando “phpenmod mysql” y se reinicia apache2 usando “service apache2 restart”
12. Verificamos que se encuentre activo el modulo de php con el comando “apachectl -M”.

Pasamos a trabajar con el contenedor B.

1. En esta parte se ejecutan los mismos pasos que realizamos al inicio del contenedor A. “apt update” y “apt upgrade”





5. Para poder acceder desde el contenedor A a la base de datos que estará alojada en este contenedor necesitamos saber la ip asignada. Lo hacemos nuevamente mediante el comando “ip a”.
6. Con el comando “apt install mariadb-server” instalamos mariaDB para gestionar la base de datos.
7. Al igual que con apache ejecutamos los comandos “systemctl start mariadb” y “systemctl enable mariadb” para que mariadb comience junto con el contenedor.
8. Se ejecuta el comando “mysql\_secure\_installation” para completar la configuración de mariaDB. La configuración se completa según se nos consulta en consola.
9. Debemos verificar que puerto utiliza mysql para poder habilitarlo. Utilizamos el comando “ss -tlnp”. Se observa que es el puerto número 3306.
10. Se procede a la configuración del firewall para abrir este puerto. Ejecutamos el comando “ufw allow 3306” que nos creará la regla que permita la comunicación a través del puerto 3306. Luego ejecutamos “ufw enable” para habilitar el firewall.
11. Debemos verificar en el contenedor A si tenemos acceso a través del puerto 3306 del puerto B. Instalamos la herramienta nmap con el comando “apt install nmap”. Escaneamos los puertos abiertos en la dirección ip del contenedor B y corroboramos que se encuentra cerrado.

The screenshot shows the Proxmox Virtual Environment interface. On the left, a list of virtual machines is visible, with VM 126 (34910490A) selected. The main panel displays the terminal output for container 126. The output shows the installation of nmap and a scan of IP 192.168.77.217. The scan results indicate that the host is up and that port 3306 is closed.

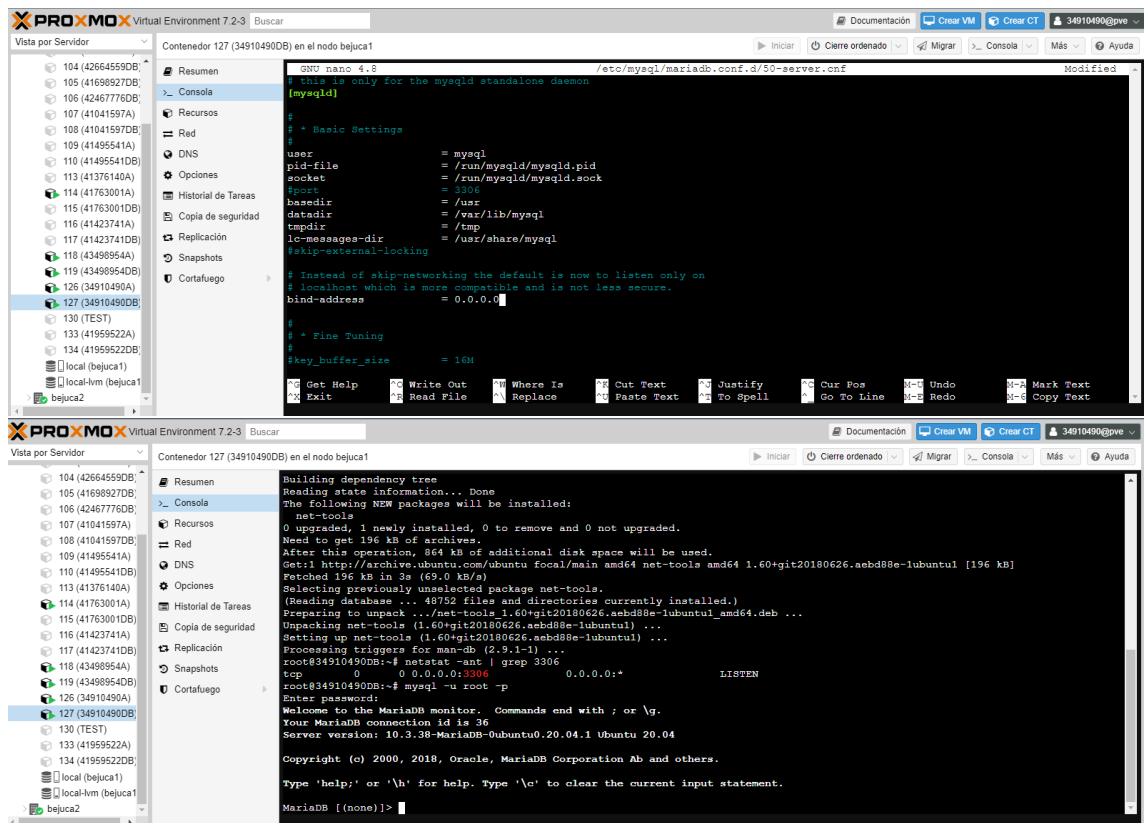
```

Unpacking lua-lpeg:amd64 (1.0.2-1) ...
Selecting previously unselected package nmap-common.
Preparing to unpack .../4-nmap-common_7.80+dfsg1-2build1_all.deb ...
Unpacking nmap-common (7.80+dfsg1-2build1) ...
Selecting previously unselected package nmap.
Preparing to unpack .../5-nmap_7.80+dfsg1-2build1_amd64.deb ...
Unpacking nmap (7.80+dfsg1-2build1) ...
Setting up lua-lpeg:amd64 (1.0.2-1) ...
Setting up libblas3:amd64 (3.9.0-1build1) ...
update-alternatives: using /usr/lib/x86_64-linux-gnu/blas/libblas.so.3 to provide /usr/lib/x86_64-linux-gnu/libblas.so.3 (libblas.so.3-x86_64-linux-gnu) in auto mode
Setting up nmap-common (7.80+dfsg1-2build1) ...
Setting up liblua5.3-0:amd64 (5.3.3-1ubuntu2) ...
Setting up liblinear4:amd64 (2.3.0+dfsg-3build1) ...
Setting up nmap (7.80+dfsg1-2build1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9) ...
root@34910490A:~# nmap 192.168.77.217
Starting Nmap 7.80 ( https://nmap.org ) at 2023-08-30 21:47 UTC
Nmap scan report for 192.168.77.217
Host is up (0.00013s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
3306/tcp   closed mysql
NMC Address: FA:0E:4D:41:88:B2 (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 6.05 seconds
root@34910490A:~#

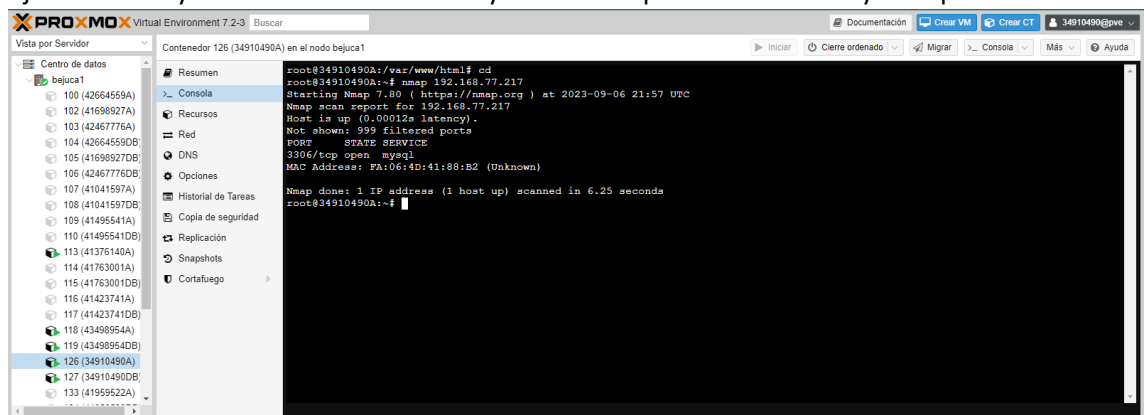
```

12. En el contenedor B debemos solucionar este asunto con el siguiente comando “nano /etc/mysql/mariadb.conf.d/50-server.cnf”. Esto nos abrirá el editor del archivo y debemos editar el parámetro “bind-address”. La configuración por defecto es 127.0.0.1 y debemos cambiarlo a 0.0.0.0 para que el puerto 3306 quede abierto. Guardamos y salimos.

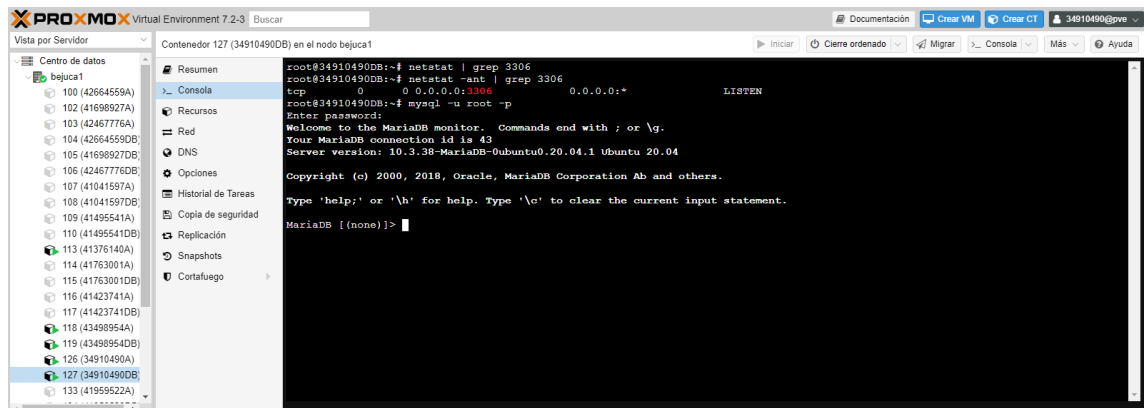




13. Ejecutamos “systemctl restart mariadb” y revisamos que los cambios hayan impactado.



14. Accedemos al gestor de mariaDB con el comando “mysql -u root -p”. Se ingresa la contraseña establecida durante la instalación de mariadb.



15. Se consultan las bases de datos existentes mediante el comando “show databases;”. Se crea la base de datos “alumnos” mediante el comando “create database alumnos;” Se



selecciona la base de datos “alumnos” mediante el comando “use alumnos;”. Se crea una tabla “alumnos” mediante el comando “CREATE TABLE alumnos (...)”. Se agregan elementos a la tabla “alumnos” mediante el comando “INSERT INTO alumnos VALUES (...)”. Finalmente, comprobamos que los elementos hayan sido agregados a la tabla “alumnos” de la base de datos mediante el comando “SELECT \* FROM alumnos;”

The screenshot shows a terminal window in Proxmox VE. The user is in a container named 'bejuca1' and has run the command 'cd /etc/mysql'. They then run 'mysql -u root -p', which results in a syntax error: 'ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'cd' at line 1'. After exiting MySQL, they run 'netstat -tlnp', which displays the listening ports and processes for the system.

```

MariaDB [alumnos]> cd
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'cd' at line 1
MariaDB [alumnos]> Bye
root@34910490DB:~# netstat -tlnp
State      Recv-Q     Send-Q       Local Address:Port       Peer Address:Port
Process
LISTEN     0          4096        127.0.0.0:53              0.0.0.0:*
users: (("systemd-resolve",pid=142,fd=13))
LISTEN     0          100        127.0.0.1:25              0.0.0.0:*
users: (("master",pid=431,fd=13))
LISTEN     0          80         0.0.0.0:3306              0.0.0.0:*
users: (("mysqld",pid=324,fd=22))
LISTEN     0          511        *:80                      *:80
users: (("apache2",pid=475,fd=4), ("apache2",pid=474,fd=4), ("apache2",pid=473,fd=4), ("apache2",pid=472,fd=4), ("apache2",pid=471,fd=4), ("apache2",pid=454,fd=4))
LISTEN     0          4096        *:22                      *:22
users: (("systemd",pid=1,fd=36))
LISTEN     0          100        [::]:25                   [::]:*
users: (("master",pid=431,fd=14))
root@34910490DB:~#

```

The screenshot shows a terminal window in Proxmox VE. The user is in a container named 'bejuca1' and has run a script to install MariaDB and git. The script output shows the installation of git-commit-2.90.1, git-2.90.1, and the MariaDB server. It also shows the configuration of the root password and the creation of the 'test' database.

```

Install/git-commit-2.90.1: byte-compiling for emacs
Install/elpa-ghub for emacs
Install/ghub-3.3.0: Handling install of emacs flavor emacs
Install/ghub-3.3.0: byte-compiling for emacs
Install/elpa-magit for emacs
Install/magit-2.90.1: Handling install of emacs flavor emacs
Install/magit-2.90.1: byte-compiling for emacs
Setting up git-el (1:2.25.1-lubuntu3.11) ...
Install git for emacs
Install git for emacs
Setting up emacs (1:26.3+1-lubuntu2) ...
Setting up git-all (1:2.25.1-lubuntu3.11) ...
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
root@34910490DB:~# systemctl start mariadb
root@34910490DB:~# systemctl enable mariadb
root@34910490DB:~# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):

```

The screenshot shows a terminal window in Proxmox VE. The user is in a container named 'bejuca1' and has run the script to complete the MariaDB installation. The script output shows the removal of the 'test' database, the reloading of the privilege tables, and the completion of the installation.

```

ensures that someone cannot guess at the root password from the network.
Disallow root login remotely? [Y/n] n
... skipping.

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

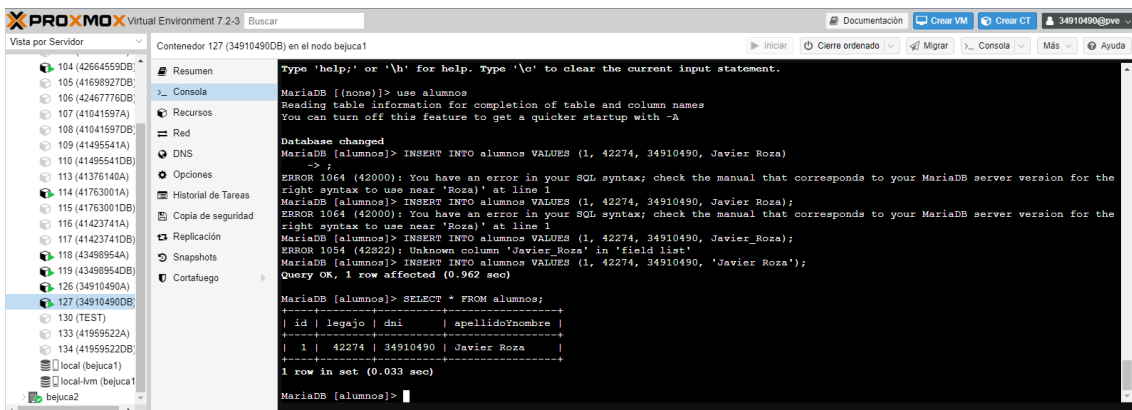
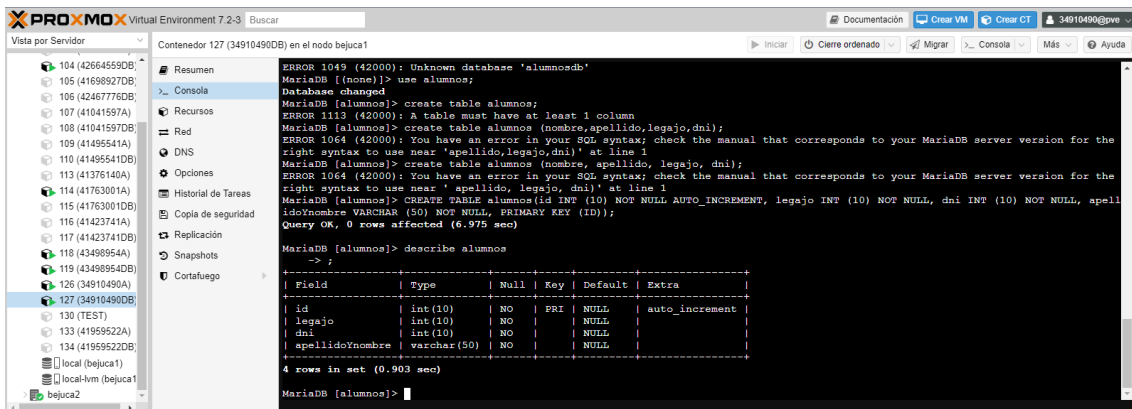
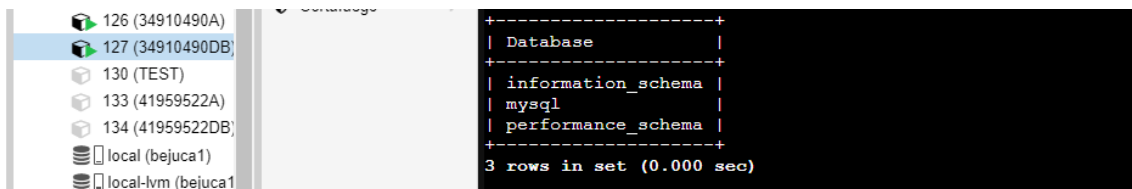
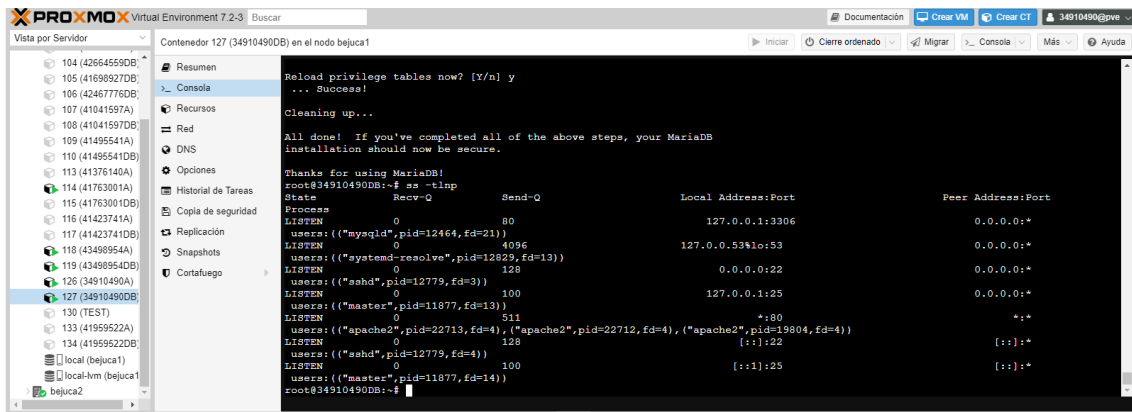
Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
root@34910490DB:~#

```



16. Se crea un usuario para acceder de forma remota a la base de datos utilizando los comandos:

- GRANT ALL ON \*.\* TO 'nombre'@'IPContenedorA' IDENTIFIED BY 'contraseña' WITH

- b. GRANT OPTION;
  - c. FLUSH PRIVILEGES;
  - d. Salimos de mysql con las teclas "Ctrl + d".
17. Instalamos php con el comando "apt install php"

Que tecnologías se utilizaron en este proyecto:

- 1. Contenedor A:
  - a. CSS
  - b. Html
  - c. Apache2
  - d. PHP
  - e. Ubuntu
- 2. Contenedor B:
  - a. PHP
  - b. MariaDB
  - c. Ubuntu

Ambos contenedores alojados en la plataforma de proxmox que a su vez se ejecuta sobre el hardware del servidor.