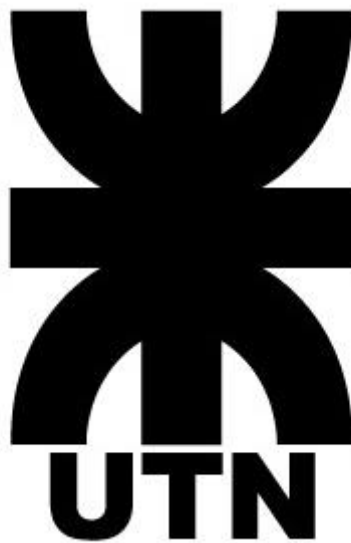


VIRTUALIZACION - CONSOLIDACION DE SERVIDORES

2023



Trabajo Final Integrador

Comisión 5K3

Docente: Luis Maria Carriles

Estudiante: Juan Marcos Alonso Lieb

Este es mi proyecto final de la materia Virtualización Consolidación de Servidores el cual tuvimos como objetivo crear dos contenedores en Proxmox y montar un Blog personal, personalmente lo hice con HTML y CCS.

Proxmox VE es una plataforma de virtualización de código abierto que proporciona una solución completa para la virtualización de servidores, permitiendo a los administradores de sistemas gestionar recursos de manera eficiente y desplegar aplicaciones en entornos virtualizados. Es una opción popular en entornos empresariales y de servidores para consolidación de servidores, implementaciones de nube privada y más.

Algunas de las características clave de Proxmox VE incluyen:

1. Virtualización basada en contenedores: Proxmox utiliza LXC (Linux Containers) para proporcionar una virtualización a nivel de sistema operativo que es liviana y eficiente. Los contenedores comparten el mismo núcleo del sistema operativo subyacente, lo que los hace más rápidos y eficientes en términos de recursos en comparación con las máquinas virtuales tradicionales.
2. Virtualización basada en hipervisor: Proxmox también admite la virtualización de hardware completa mediante KVM (Kernel-based Virtual Machine) y QEMU. Esto permite ejecutar máquinas virtuales con diferentes sistemas operativos en el mismo servidor.
3. Interfaz de usuario web: Proxmox ofrece una interfaz web fácil de usar llamada Proxmox Virtual Environment (PVE) que permite a los administradores de sistemas gestionar máquinas virtuales, contenedores, almacenamiento y redes de manera centralizada.
4. Almacenamiento definido por software: Proxmox VE incluye herramientas para gestionar el almacenamiento de forma eficiente, incluyendo soporte para almacenamiento local, SAN (Storage Area Network), NFS (Network File System), Ceph y más.
5. Alta disponibilidad: Proxmox VE permite la configuración de clústeres de servidores para lograr alta disponibilidad y redundancia.
6. Copias de seguridad y restauración: Ofrece herramientas integradas para realizar copias de seguridad de máquinas virtuales y contenedores, lo que facilita la recuperación en caso de fallos o pérdida de datos.

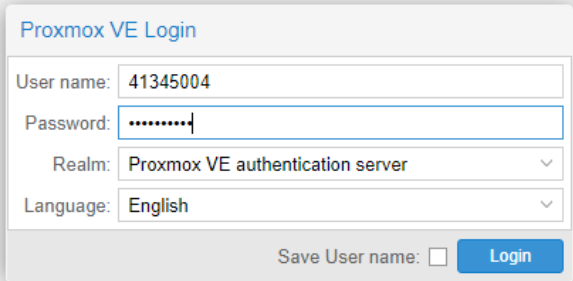
De esta manera creé dos contenedores con Nginx y lo configure de forma tal que pueda alojar una página web para poder mostrar un blog personal. Un contenedor tendrá al blog personal y otro contendrá la base de datos.

Características de los contenedores LXC creados en Proxmox:

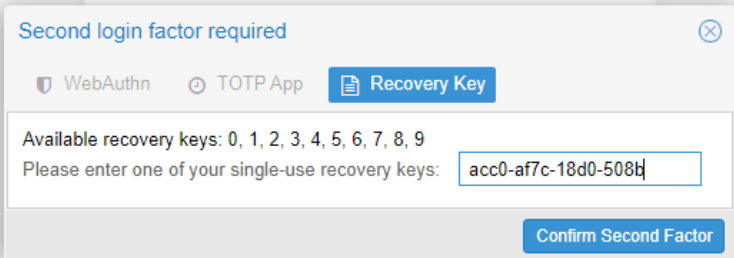
- Nombre del contenedor: “DNI + DB” (Para backend), y “DNI + A” (Para frontend).
- Memoria RAM: 128 MB.
- Almacenamiento: 8GB.
- Procesador: 1 núcleo.
- Red: DHCP.

A continuación, haremos un paso a paso de cómo crear los mismos:

1. Logearnos en Proxmox con la cuenta que previamente nos brindó el profesor y luego configuramos nosotros con la forma de autenticación que seleccionamos para acceder a la plataforma.



The image shows the Proxmox VE Login interface. It includes fields for User name (41345004), Password (masked with dots), Realm (Proxmox VE authentication server), and Language (English). There is a checkbox for 'Save User name' and a 'Login' button.



The image shows a 'Second login factor required' dialog box. It offers three authentication methods: WebAuthn, TOTP App, and Recovery Key. The Recovery Key method is selected, showing a list of available recovery keys (0-9) and a text input field where a key (acc0-af7c-18d0-508b) has been entered. A 'Confirm Second Factor' button is at the bottom right.

2. Debemos hacer click en el botón “Crear CT” en la parte superior derecha.



3. Ingresamos el nombre del contenedor y le asignamos una contraseña. Hacemos click en siguiente.

Create: LXC Container

General Template Disks CPU Memory Network DNS Confirm

Node: bejuca2 Resource Pool:
CT ID: 139 Password:
Hostname: 41345004A Confirm password:
Unprivileged container: ☒ SSH public key:
Nesting: ☒ [Load SSH Key File](#)

Help Advanced ☐ Back Next

4. Seleccionamos la plantilla y hacemos click en siguiente.

Create: LXC Container

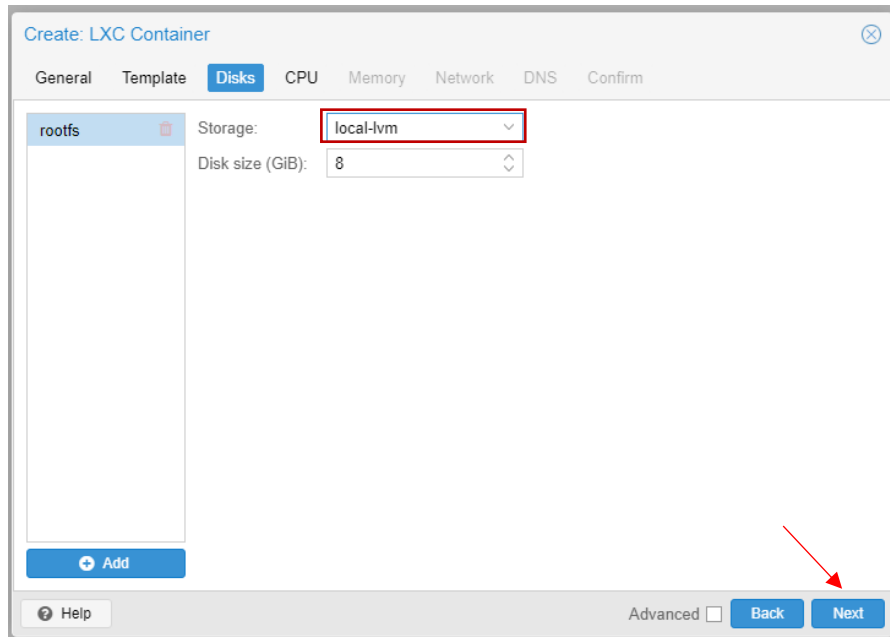
General Template Disks CPU Memory Network DNS Confirm

Storage: local Template: debian-11-turnkey-ansible_17.1-1_

Name	For...	Size
debian-10-standard_10.7-1_amd64.tar.gz	tgz	231.06 MB
debian-11-turnkey-ansible_17.1-1_amd64.tar.gz	tgz	551.59 MB
ubuntu-20.04-standard_20.04-1_amd64.tar.gz	tgz	214.20 MB

Help Advanced ☐ Back Next

5. En este paso vamos a determinar el tamaño del disco, el mismo debe ser de 8Gb. Luego de eso hacemos click en siguiente.

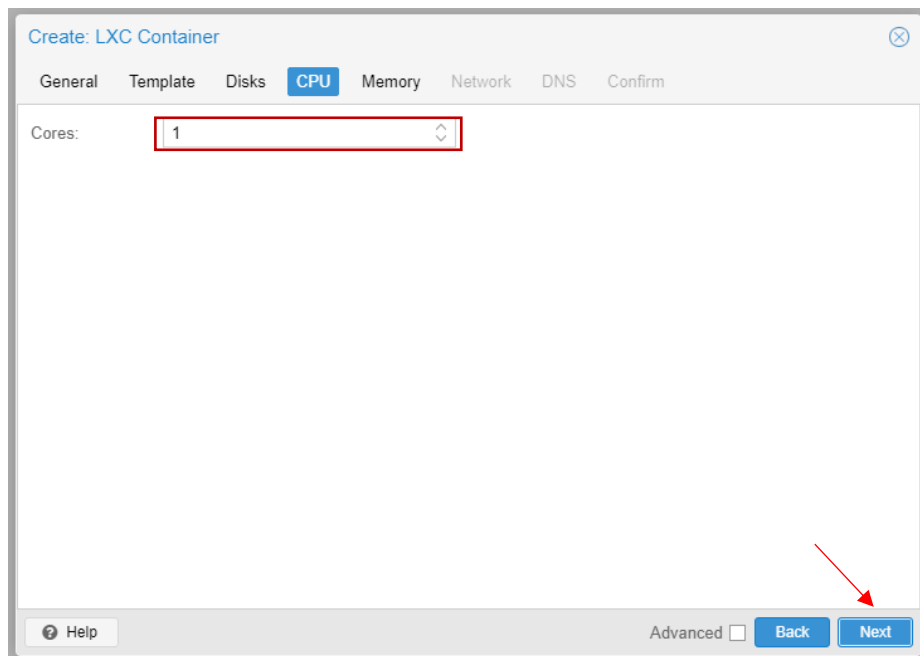


The screenshot shows the 'Create: LXC Container' dialog box with the 'Disks' tab selected. The 'Storage' dropdown is set to 'local-lvm' and the 'Disk size (GiB)' is set to '8'. A red box highlights the 'Storage' dropdown. A red arrow points to the 'Next' button at the bottom right.

Storage: local-lvm
Disk size (GiB): 8

Back Next

6. Hacemos lo mismo que en el paso número 5, solo que en este caso seleccionamos la cantidad de núcleos que va a tener. El mismo debe ser de 1 núcleo. Luego de eso hacemos click en siguiente.

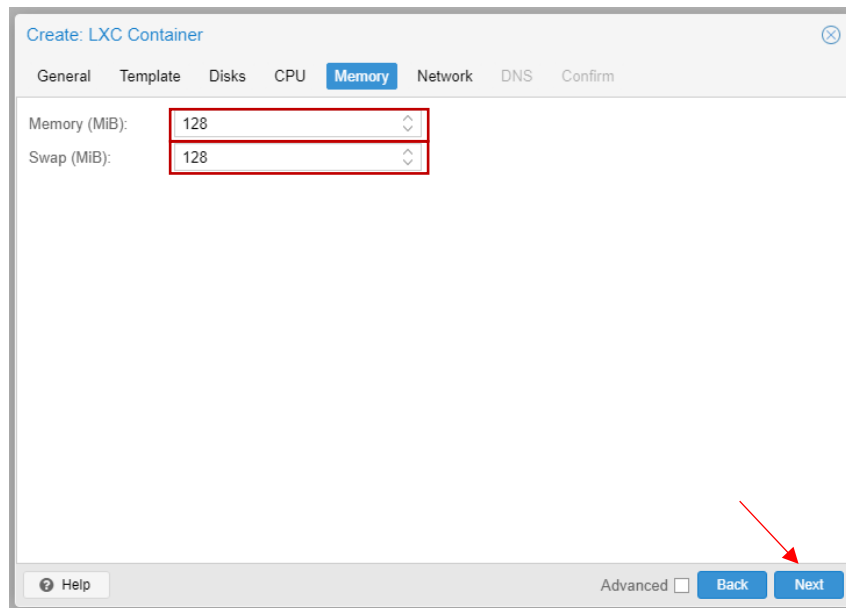


The screenshot shows the 'Create: LXC Container' dialog box with the 'CPU' tab selected. The 'Cores' dropdown is set to '1'. A red box highlights the 'Cores' dropdown. A red arrow points to the 'Next' button at the bottom right.

Cores: 1

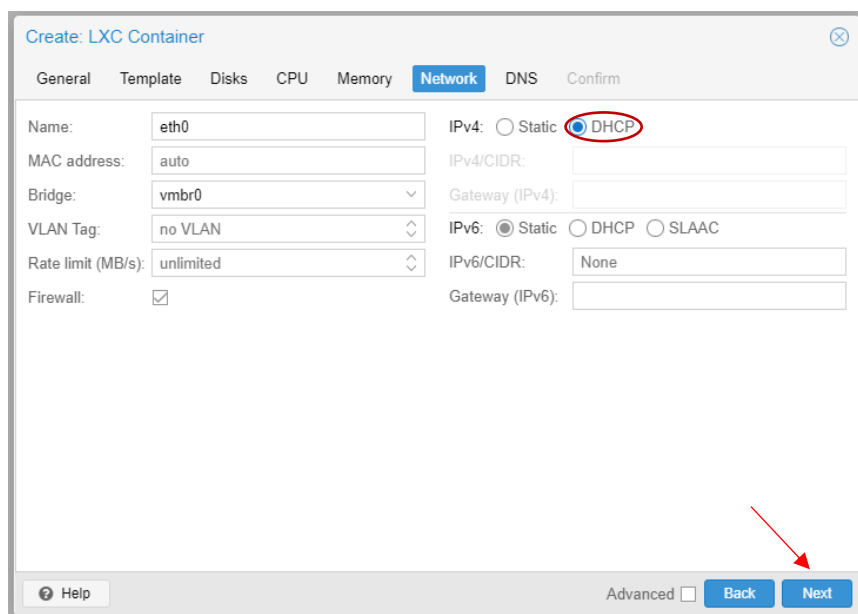
Back Next

7. Al igual que en el paso número 5 y 6 vamos a indicar cuanto memoria RAM le vamos a asignar al contenedor en este caso serán 128Mb. Hacemos click en siguiente.



The screenshot shows the 'Create: LXC Container' dialog box with the 'Memory' tab selected. The 'Memory (MiB)' field is set to 128 and the 'Swap (MiB)' field is also set to 128. Both fields are highlighted with a red rectangle. At the bottom right, a red arrow points to the 'Next' button. The 'Advanced' checkbox is unchecked.

8. Configuramos la red seleccionando DHCP. Hacemos click en siguiente.



The screenshot shows the 'Create: LXC Container' dialog box with the 'Network' tab selected. The 'IPv4' section has the 'DHCP' radio button selected, which is circled in red. The 'IPv6' section has the 'Static' radio button selected. The 'Name' field is 'eth0', 'MAC address' is 'auto', 'Bridge' is 'vmbro', 'VLAN Tag' is 'no VLAN', 'Rate limit (MB/s)' is 'unlimited', and 'Firewall' is checked. At the bottom right, a red arrow points to the 'Next' button. The 'Advanced' checkbox is unchecked.

9. No colocamos nada en la configuración de DNS y hacemos click en siguiente.
10. Por último nos mostrara en la pestaña de "Confirmar" Toda la configuración que definimos para el contenedor de esta manera podemos corroborar que todas las configuraciones que hicimos quedaron de la forma correcta y si no es así se puede arreglar. Luego de ver que todo esté en orden hacemos click en Finalizar.

11. Esperamos que se cree el contenedor y al finalizar nos dirá “TASK OK”, de esta manera sabremos que creamos de forma exitosa nuestro contenedor. Luego de eso cerramos la ventana.
12. Se realiza el mismo procedimiento para el CT de la base de datos.

Instalación de Nginx

a. Actualizamos los paquetes del controlador e instalamos Nginx usando:

apt update

apt upgrade

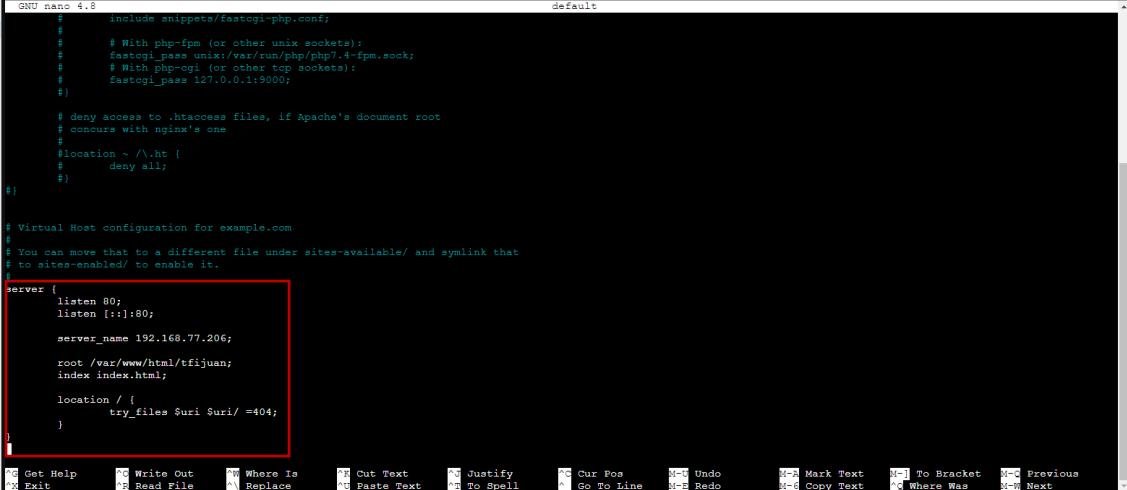
apt install nginx

b. Usamos el archivo default.conf

c. Para poder configurar el archivo default.conf debemos conocer el ip del contenedor con ip addr show

```
root@41345004A:/etc/nginx/sites-enabled# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0@if33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 82:0f:db:b9:66:a0 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.77.206/24 brd 192.168.77.255 scope global dynamic eth0
        valid_lft 493sec preferred_lft 493sec
    inet6 fe80::800f:dbff:feb9:66a0/64 scope link
        valid_lft forever preferred_lft forever
```

d. Aplicamos una configuracion basica de nginx en el archivo



```
GNU nano 4.8 default
#
#   include snippets/fastcgi-php.conf;
#
#   # With php-fpm (or other unix sockets):
#   fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
#   # With php-cgi (or other tcp sockets):
#   fastcgi_pass 127.0.0.1:9000;
#
# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
#location ~ /\.ht {
#    deny all;
#}

# Virtual Host configuration for example.com
#
# You can move that to a different file under sites-available/ and symlink that
# to sites-enabled/ to enable it.
#
server {
    listen 80;
    listen [::]:80;

    server_name 192.168.77.206;

    root /var/www/html/tfjuaan;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Para descargar los archivos del blog

1. Creamos un repositorio en github con los archivos para poder usar los archivos en el servidor de proxmox y copiamos el link del repositorio:

<https://github.com/juanmarcosal/tfijuanm>

2. Nos posicionamos en la siguiente locación usando este comando:

```
cd /var/www/html
```

3. Y procedemos a descargar el archivo de github

```
git clone https://github.com/juanmarcosal/tfijuanm
```

4. Procedemos a reiniciar el servicio de nginx:

```
systemctl restart nginx
```

5. Una vez cargados los archivos, usamos el link con la redirección de puertos correspondiente para visualizar el blog desde la plataforma de proxmox

Siendo el link:

319e02b588a6.sn.mynetname.net:8012

Instalacion y configuracion del contenedor 2 (Base de datos)

1. Para la instalación del contenedor de la DB, repetimos el paso 1 del contenedor

anterior, solo que, al nombre, en vez de usar DNI + A, empleamos DNI + DB

2. Instalamos MariaDB server:

```
sudo apt update
```

```
sudo apt install mariadb-server
```

Durante la instalación, nos pedirá configurar la contraseña del root de MariaDB

3. Creamos una base de datos, accediendo a MariaDB

```
sudo mysql -u root -p
```

4. Creamos una nueva base de datos y un usuario con sus respectivos permisos

```
CREATE DATABASE DBtfi;
```

```
CREATE USER 'juan'@'localhost' IDENTIFIED BY '123456';
```

```
GRANT ALL PRIVILEGES ON DBtfi.* TO 'juan'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
EXIT;
```


5. Podemos revisar las bases de datos creadas con SHOW DATABASES;

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| DBtfti   |
| information_schema |
| mysql     |
| performance_schema |
+-----+
4 rows in set (0.075 sec)

MariaDB [(none)]> 
```