

VIRTUALIZACIÓN - CONSOLIDACIÓN DE SERVIDORES 2023

Trabajo Final Integrador

Comisión: 5K3

Fecha de Presentación: 31/08/2023

Docente:

Prof. Carriles, Maria Luis

Alumno: Siales, Eduardo

Legajo: 44670

INTRODUCCIÓN

En este trabajo, se utilizará la plataforma de Proxmox, una plataforma de virtualización de código abierto, para poder crear dos contenedores. Lo que busca este trabajo es configurar un alojamiento web para un blog personal. Para ello, el contenedor donde estará alojado el blog personal usará Nginx. Mientras que el de la base de datos contará con servicio de MariaDB. Finalmente para la creación del blog, el mismo poseerá scripts realizados en HTML y CSS.

En relación a que es Proxmox

Proxmox Virtual Environment (Proxmox VE) es una plataforma de virtualización de código abierto que combina la virtualización de servidores y el manejo de contenedores en un único entorno integrado. Proxmox VE permite crear y gestionar máquinas virtuales (VMs) y contenedores en un entorno web fácil de usar.

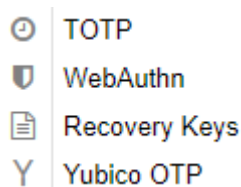
Características principales de Proxmox VE:

1. Virtualización: Proxmox VE admite la virtualización de VM para la virtualización completa y LXC (Linux Containers) para la virtualización ligera.
2. Gestión de Contenedores: Los contenedores ofrecen un enfoque más liviano para la virtualización al compartir recursos del sistema operativo anfitrión.
3. Alta Disponibilidad: Proxmox VE admite la configuración de clústeres y alta disponibilidad. Esto significa que se puede crear una agrupación de servidores y compartir recursos para garantizar la continuidad del servicio incluso si un nodo falla.
4. Respallos y Restauraciones: Proxmox VE incluye capacidades integradas para realizar copias de seguridad y restaurar VMs y contenedores. Esto es fundamental para la recuperación ante desastres y la migración de VM.
5. Almacenamiento Integrado: Proxmox VE admite una variedad de tipos de almacenamiento, incluidos sistemas de archivos locales, almacenamiento en red (NFS), almacenamiento compartido, entre otros.
6. Migración en Vivo: La migración en vivo permite mover máquinas virtuales en funcionamiento de un nodo a otro sin interrupciones para los usuarios. Esto es especialmente útil para la administración de carga y la actualización de hardware.

Autenticación de Dos Factores (TFA)

La autenticación de dos factores (TFA) es un método de seguridad que requiere dos formas diferentes de verificar la identidad del usuario antes de permitir el acceso a una cuenta o sistema. Para esta tarea, se añade una capa adicional de protección más allá de la simple autenticación mediante una contraseña.

Se puede mencionar 4 tipos de autenticación de dos factores



TOTP (Time-Based One-Time Password)

El TOTP es un protocolo de generación de contraseñas de un solo uso basado en el tiempo. Funciona generando códigos que cambian en intervalos de tiempo regulares (generalmente 30 segundos). Estos códigos se generan en función de una clave secreta compartida entre el servicio y el usuario. El usuario ingresa el código actual junto con su contraseña para autenticarse.

WebAuthn (Web Authentication)

WebAuthn es un estándar web para la autenticación fuerte y sin contraseñas. Permite la autenticación mediante factores como huellas dactilares, reconocimiento facial o hardware de seguridad. WebAuthn se basa en criptografía asimétrica, y en lugar de enviar contraseñas, se utilizan claves públicas y privadas para autenticar al usuario.

Recovery Keys (Claves de Recuperación)

Las claves de recuperación son códigos especiales proporcionados al usuario durante la configuración de TFA. Estas claves se almacenan en un lugar seguro y se utilizan como método de respaldo para acceder a la cuenta en caso de que se pierda o no se pueda acceder al dispositivo de autenticación principal. Deben guardarse con seguridad, ya que son similares a tener acceso completo a la cuenta.

Yubico OTP

Yubico OTP es un protocolo desarrollado por Yubico, una empresa de seguridad. Implica el uso de dispositivos USB llamados llaves YubiKey. Estas llaves generan códigos únicos en cada uso. Cuando se inserta en una computadora o dispositivo móvil, se genera un código OTP que se puede usar junto con la contraseña para autenticación.

Para el uso de la autenticación de dos factores en la plataforma Proxmox se empleó TOTP

Login a Proxmox VE

Nombre de Usuario:

40530706

Contraseña:

.....|

Ambito:

Proxmox VE authentication server

Idioma:

Spanish

Guardar nombre de usuario:

☒

Login

Y aqui esta la solicitud del doble factor al loguear:

Second login factor required

WebAuthn

TOTP App

Recovery Key

Please enter your TOTP verification code:

Confirm Second Factor

Instalación y configuración del contenedor 1 (Aplicación para el blog)

1. Creación del contenedor en la plataforma de Proxmox.

Creamos un contenedor en Proxmox. Luego le colocamos un nombre y contraseña. Siendo el nombre del contenedor, el dni + A

The screenshot shows the 'General' tab of the Proxmox VE container creation interface. The 'Nodo' (Node) is set to 'bejuca1'. The 'CT ID' (Container ID) is '100'. The 'Nombre del Host' (Host Name) field is empty. The 'Contenedores sin privilegios' (Unprivileged containers) checkbox is checked. The 'Nesting' checkbox is also checked. On the right side, there are fields for 'Conjunto de Recursos' (Resource Set), 'Contraseña' (Password), 'Confirmar contraseña' (Confirm Password), and 'Clave pública SSH' (SSH Public Key). A blue button labeled 'Carga archivo de clave SSH' (Load SSH key file) is located below the SSH key field.

Elegimos la plantilla, para este caso usamos la de Ubuntu 20.04

The screenshot shows the 'Plantilla' (Template) tab. The 'Almacenamiento' (Storage) is set to 'local'. The 'Plantilla' (Template) dropdown menu is open, showing a list of templates. The list has columns for 'Nombre' (Name), 'For...' (Format), and 'Tamaño' (Size). The selected template is 'ubuntu-20.04-standard_20.04-1_amd64.tar.gz' with a size of 214.20 MB.

Nombre	For...	Tamaño
debian-11-standard_11.6-1_amd64.tar.zst	tzst	123.19 MB
ubuntu-20.04-standard_20.04-1_amd64.tar.gz	tgz	214.20 MB

Luego asignamos el tamaño de almacenamiento que usará el contenedor, que será de 8GB

The screenshot shows the 'Discos' (Disks) tab. A disk named 'rootfs' is listed. The 'Almacenamiento' (Storage) is set to 'local-lvm'. The 'Tamaño de disco (GiB)' (Disk size in GiB) is set to '8'. There is an 'Agregar' (Add) button at the bottom.

Luego asignamos la cantidad de núcleos que usará el contenedor, que será de 1.

The screenshot shows the 'CPU' tab. The 'Núcleos' (Cores) field is set to '1'.

Colocamos la cantidad de memoria RAM a emplear en el contenedor, que sera de 128MB

General	Plantilla	Discos	CPU	Memoria	Red	DNS	Confirmar
Memoria (MiB):	<input type="text" value="128"/>						
Swap (MiB):	<input type="text" value="128"/>						

Luego establecemos la red como DHCP y confirmamos

General	Plantilla	Discos	CPU	Memoria	Red	DNS	Confirmar
Nombre:	<input type="text" value="eth0"/>			IPv4:	<input type="radio"/> Estático <input checked="" type="radio"/> DHCP		
Dirección MAC:	<input type="text" value="auto"/>			IPv4/CIDR:	<input type="text"/>		
Puente:	<input type="text" value="vbr0"/>			Puerta de enlace (IPv4):	<input type="text"/>		
Etiqueta VLAN:	<input type="text" value="no VLAN"/>			IPv6:	<input checked="" type="radio"/> Estático <input type="radio"/> DHCP <input type="radio"/> SLAAC		
Tasa límite (MB/s):	<input type="text" value="unlimited"/>			IPv6/CIDR:	<input type="text" value="None"/>		
Cortafuego:	<input checked="" type="checkbox"/>			Puerta de enlace (IPv6):	<input type="text"/>		

Para nuestro caso se emplean los contenedores 111 y 112 en la plataforma de Proxmox

	lxc	111 (40530706A)
	lxc	112 (40530706DB)

2. Instalación de Nginx

- Actualizamos los paquetes del controlador e instalamos Nginx junto con PHP-FPM usando:

```
sudo apt update
sudo apt install nginx php-fpm
```

- Creamos un archivo de instalación mi-blog.conf:

```
sudo nano /etc/nginx/sites-available/mi-blog.conf
```

- Para poder configurar el archivo mi-blog.conf debemos conocer el ip del contenedor con `ip addr show`

```
root@40530706A:~# 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@if193: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 5a:33:2d:54:79:c3 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.77.212/24 brd 192.168.77.255 scope global dynamic eth0
        valid_lft 516sec preferred_lft 516sec
    inet6 fe80::5833:2dff:fe54:79c3/64 scope link
        valid_lft forever preferred_lft forever
```

- Aplicamos una configuracion basica de nginx y php-fpm en el archivo

```
GNU nano 4.8 /etc/nginx/sites-available/mi-blog.conf
server {
    listen 80;
    server_name 192.168.77.212;

    root /var/www/html/MiBlog-main;

    index index.html;

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

Nota: se añadió MiBlog-main; para poder usar el archivo index del repositorio que se hablará más adelante

En caso que solo se use html sin poner MiBlog-main, nos saldrá solo la página de bienvenida a nginx al usar la redirección de puertos

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

- e. Habilitamos el archivo de configuración:

```
sudo ln -s /etc/nginx/sites-available/mi-blog.conf /etc/nginx/sites-enabled/
```

- f. Reiniciamos Nginx y PHP-FPM

```
sudo service nginx restart
sudo service php7.4-fpm restart
```

Instalacion y configuracion del contenedor 2 (Base de datos)

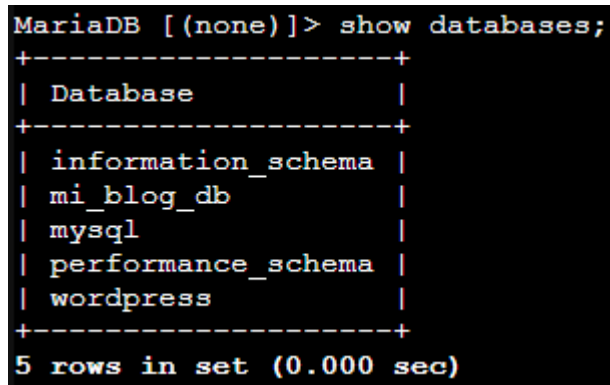
1. Para la instalación del contenedor de la DB, repetimos el paso 1 del contenedor anterior, solo que al nombre, en vez de usar DNI + A, empleamos DNI + DB
2. Instalamos MariaDB server:
`sudo apt update`
`sudo apt install mariadb-server`

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 nombredelaDB;  
CREATE USER 'usuario'@'localhost' IDENTIFIED BY 'contraseña';  
GRANT ALL PRIVILEGES ON nombredelaDB.* TO 'usuario'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;
```

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



```
MariaDB [(none)]> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mi_blog_db |  
| mysql |  
| performance_schema |  
| wordpress |  
+-----+  
5 rows in set (0.000 sec)
```

Si bien el contenido está de forma estática al descomprimir los archivos, los usuarios para la base de datos ya poseen los permisos necesarios.

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/lalito14/MiBlog.git>
2. Nos posicionamos en la siguiente locación usando este comando:
`cd /var/www/html`
3. Y procedemos a descargar el archivo de github
`wget https://github.com/usuario/nombre-repositorio/archive/main.zip`
4. Y descomprimos el archivo zip
`unzip main.zip`

5. Procedemos a reiniciar el servicio de nginx:
`sudo service nginx restart`
6. 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:

<http://319e02b588a6.sn.mynetname.net:8013/>

