Práctica 1.10.b

Implantación de Aplicaciones Web

José Juan Sánchez Hernández

Curso 2024/2025

Índice

1 Balanceador de carga con Nginx **1**

1.1 Infraestructura necesaria 1
1.2 Breve descripción de los requisitos 1
1.3 Arquitectura de una aplicación web en tres niveles 3
1.4 Instalación de Nginx 4
1.5 Cómo configurar Nginx como balanceador de carga 4
1.6 Métodos de balanceo de carga 5
1.7 Entregables 6
1.7.1 Documento técnico 6
1.7.2 Scripts de Bash 6

2 Referencias **8**

3 Licencia **9**

Índice de figuras

Índice de cuadros

1 Balanceador de carga con Nginx

En esta práctica vamos a configurar Nginx para que trabaje como un balanceador de carga. Se recomienda la lectura del artículo Nginx Load Balancing - HTTP Load Balancer.

1.1 Infraestructura necesaria

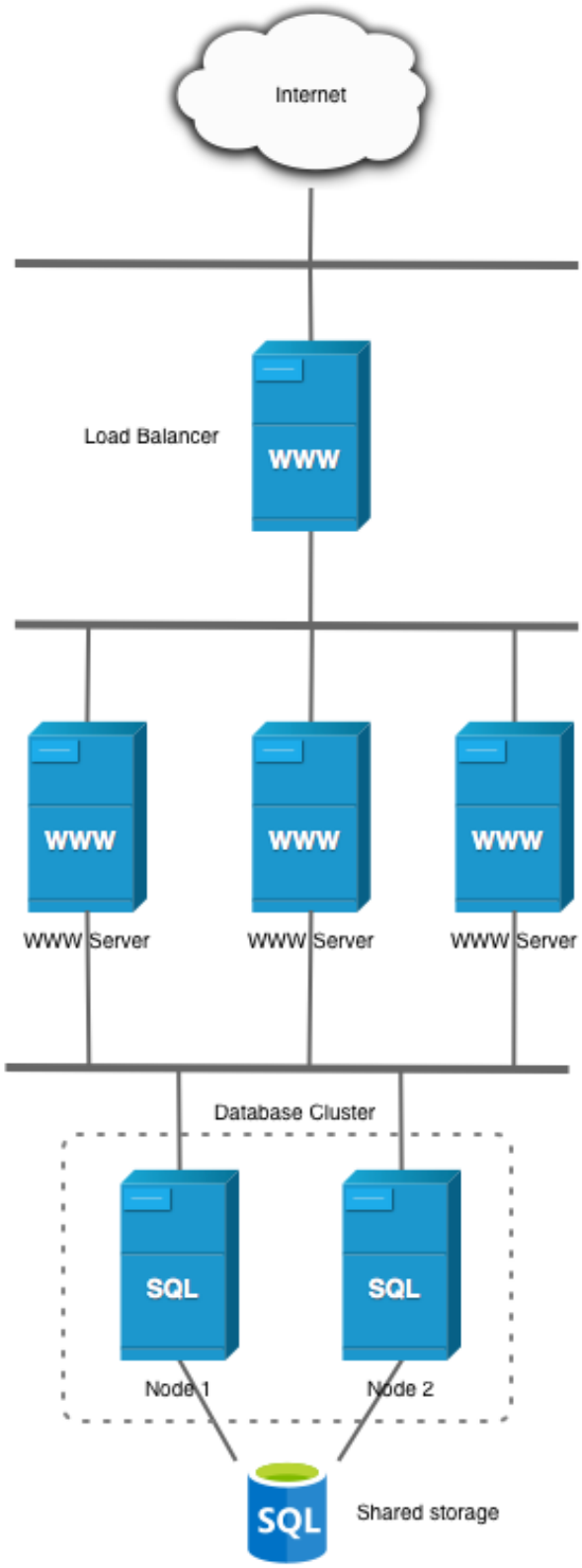
La infraestructura propuesta consta de cuatro máquinas virtuales, una para el balanceador de carga (*Load Balancer*), dos como frontales web (*Front-End*) y una como servidor de base de datos MySQL (*Back-End*).

- Balanceador.
- Frontal Web 1.
- Frontal Web 2.
- Servidor de Base de Datos MySQL.

1.2 Breve descripción de los requisitos

- Tiene que utilizar **máquinas virtuales EC2** de **Amazon Web Services (AWS)**, con la última versión de **Ubuntu Server**.
- El balanceador tendrá instalado el servidor Nginx.
- Las dos máquinas que actúan como frontales web tendrán que estar configuradas para servir páginas PHP con acceso a una base de datos MySQL.
- La instancia EC2 que hace de balanceador tiene que utilizar una dirección IP elástica de AWS.
- Registre un nombre de dominio en algún proveedor de nombres de dominio gratuito. Por ejemplo, puede hacer uso de Freenom o No-IP.
- Configure los registros DNS del proveedor de nombres de dominio para que el nombre de dominio de ha registrado pueda resolver hacia la dirección IP elástica que ha asociado con su instancia EC2 de AWS.
- Instale y configure el cliente ACME Certbot en su instancia EC2 de AWS que hace balanceador para que pueda aceptar tráfico HTTPS. Siga los pasos de la documentación oficial.

1.3 Arquitectura de una aplicación web en tres niveles



1.4 Instalación de Nginx

En primer lugar, tendremos que instalar el servidor web Nginx.

```
sudo apt update
sudo apt install nginx -y
```

Una vez instadlo, podemos examinar los directorios donde se almacenan los archivos de configuración de los **Virtual Hosts** de Nginx:

- `/etc/nginx/sites-available/`: Virtual Hosts disponibles.
- `/etc/nginx/sites-enabled/`: Virtual Hosts habilitados. Estos archivos son enlaces simbólicos a los archivos de configuración que hay en `/etc/nginx/sites-available/`.

En la instalación por defecto sólo tendremos un archivo de configuración **`default`** en ambos directorios.

Una posible configuración para un Virtual Host que responde peticiones HTTP en el puerto 80 podría ser la siguiente:

```
server {
        listen 80;

        server_name _;

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

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

1.5 Cómo configurar Nginx como balanceador de carga

Para configurar Nginx como balanceador de carga, vamos a deshabilitar el sitio por defecto eliminando el enlace simbólico con el siguiente comando:

```
sudo unlink /etc/nginx/sites-enabled/default
```

Una vez eliminado, vamos a crear un nuevo archivo de configuración llamado `load-balancer.conf` en el directorio `/etc/nginx/sites-available/` con el siguiente contenido:

```
upstream frontend_servers {
    server IP_FRONTEND_1;
    server IP_FRONTEND_2;
}

server {
    listen 80;
    server_name _;

```

```
    location / {
        proxy_pass http://frontend_servers;
    }
}
```

Tenga en cuenta que deberá configurar **IP_FRONTEND_1** y **IP_FRONTEND_2** con los valores de las direcciones IP donde quiere que Nginx redirija las peticiones HTTP.

Habilite el nuevo Virtal Host que acaba de crear.

```
sudo ln -s /etc/nginx/sites-available/load-balancer.conf /etc/nginx/sites-enabled/
```

Recargue la configuración de Nginx.

```
sudo systemctl reload nginx
```

Nota: Si tiene algún problema al recargar el servicio de Nginx, puede ejecutar el siguiente comando para revisar la sintaxis del archivo de configuración.

```
sudo nginx -t
```

1.6 Métodos de balanceo de carga

Nginx nos permite configurar algunos métodos de balanceo de carga dentro de la directiva `upstream`.

- **Round Robin**: Es el método por defecto. Nginx distribuye las peticiones de manera equitativa entre los servidores que se hayan definido en el grupo. Ejemplo:

```
upstream frontend_servers {
  server IP_FRONTEND_1;
  server IP_FRONTEND_2;
}
```

- **Least Connections**: Nginx envía las peticiones al servidor con menos conexiones activas. Ejemplo:

```
upstream frontend_servers {
  least_conn;
  server IP_FRONTEND_1;
  server IP_FRONTEND_2;
}
```

- **IP Hash**: Nginx asigna las peticiones basándose en la dirección IP del cliente. Esto permite mantener la sesión del cliente en el mismo servidor. Es útil para mantener sesiones persistentes. Ejemplo:

```
upstream frontend_servers {
  ip_hash;
  server IP_FRONTEND_1;
  server IP_FRONTEND_2;
}
```

1.7 Entregables

Deberá crear un repositorio en **GitHub** con el nombre de la práctica y añadir al profesor como colaborador.

El repositorio debe tener el siguiente contenido:

- Un **documento técnico** con la descripción de todos los pasos que se han llevado a cabo.
- Los **scripts de Bash** que se han utilizado para automatizar la instalación y configuración de la aplicación web propuesta utilizando la arquitectura web de tres niveles.

Además del contenido anterior puede ser necesario crear otros archivos de configuración. A continuación se muestra un ejemplo de cómo puede ser la estructura del repositorio:

```
.
├── README.md
├── conf
│   ├── load-balancer.conf
│   └── 000-default.conf
└── scripts
    ├── .env
    ├── setup_load_balancer.sh
    ├── setup_letsencrypt_https.sh
    ├── install_lamp_frontend.sh
    ├── install_lamp_backend.sh
    ├── deploy_frontend.sh
    └── deploy_backend.sh
```

1.7.1 Documento técnico

El documento técnico `README.md` tiene que estar escrito en [Markdown][6] y debe incluir **como mínimo** los siguientes contenidos:

- Descripción del proceso de instalación de la instalación de [la aplicación web propuesta][30].

1.7.2 Scripts de Bash

El directorio `scripts` debe incluir los siguientes archivos:

- `.env`: Este archivo contiene todas las variables de configuración que se utilizarán en los scripts de Bash.
- `setup_load_balancer.sh`: Script de Bash con la automatización del proceso de instalación del servidor web Nginx como balanceador de carga.
- `setup_letsencrypt_https.sh`: Script de Bash con la automatización del proceso de solicitar un certificado SSL/TLS de Let's Encrypt y configurarlo en el servidor web Apache que hace de balanceador de carga.
- `install_lamp_frontend.sh`: Script de Bash con la automatización del proceso de instalación de la pila LAMP en las máquinas de frontend.

- `install_lamp_backend.sh`: Script de Bash con la automatización del proceso de instalación de la pila LAMP en la máquina de backend.
- `deploy_frontend`: Script de Bash con la automatización del proceso de instalación de la aplicación web propuesta en el frontend.
- `deploy_backend`: Script de Bash con la automatización del proceso de instalación de la aplicación web propuesta en el backend.

2 Referencias

- Balanceador de carga en Wikipedia
- Nginx Load Balancing - HTTP Load Balancer
- Simple Load Balancing
- Load Balancing Techniques and Optimizations. Jason Potter.

3 Licencia