Práctica 2.5. Balanceo de carga con Nginx

LA PRÁCTICA 2.3 DEBE ESTAR REALIZADA

1. Introducción

Los servidores proxy inversos y los balanceadores de carga son componentes de una arquitectura informática cliente-servidor. Ambos actúan como intermediarios en la comunicación entre los clientes y los servidores, realizando funciones que mejoran la eficiencia.

Las definiciones básicas son simples:

- Un proxy inverso acepta una solicitud de un cliente, la reenvía a un servidor que puede cumplirla y devuelve la respuesta del servidor al cliente.
- Un balanceador de carga distribuye las solicitudes entrantes del cliente entre un grupo de servidores, en cada caso devolviendo la respuesta del servidor seleccionado al cliente apropiado.

Suenan bastante similares, ¿verdad? Ambos tipos de aplicaciones se ubican entre clientes y servidores, aceptando solicitudes del primero y entregando respuestas del segundo. No es de extrañar que haya confusión sobre qué es un proxy inverso y un balanceador de carga. Para ayudar a diferenciarlos, exploremos cuándo y por qué normalmente se implementan en un sitio web.

1.1 Balanceadores de carga

Los balanceadores de carga se implementan con mayor frecuencia cuando un sitio necesita varios servidores porque el volumen de solicitudes es demasiado para que un solo servidor lo maneje de manera eficiente.

La implementación de varios servidores también elimina un solo punto de fallo, lo que hace que el sitio web sea más confiable. Por lo general, todos los servidores alojan el mismo contenido, y el trabajo del balanceador de carga es distribuir la carga de trabajo de manera que se haga el mejor uso de la capacidad de cada servidor, evite la sobrecarga en cualquiera de ellos y dé como resultado la respuesta más rápida posible al cliente.

Un balanceador de carga también puede mejorar la experiencia del usuario al reducir la cantidad de respuestas de error que ve el cliente. Lo hace detectando cuándo los servidores caen y desviando las solicitudes de ellos a los otros servidores del grupo. En la implementación más simple, el balanceador de carga detecta el estado del servidor al interceptar las respuestas de error a las solicitudes regulares.

En esta práctica tendremos el escenario donde Nginx hará tanto de proxy inverso como de balanceador de carga al mismo tiempo.

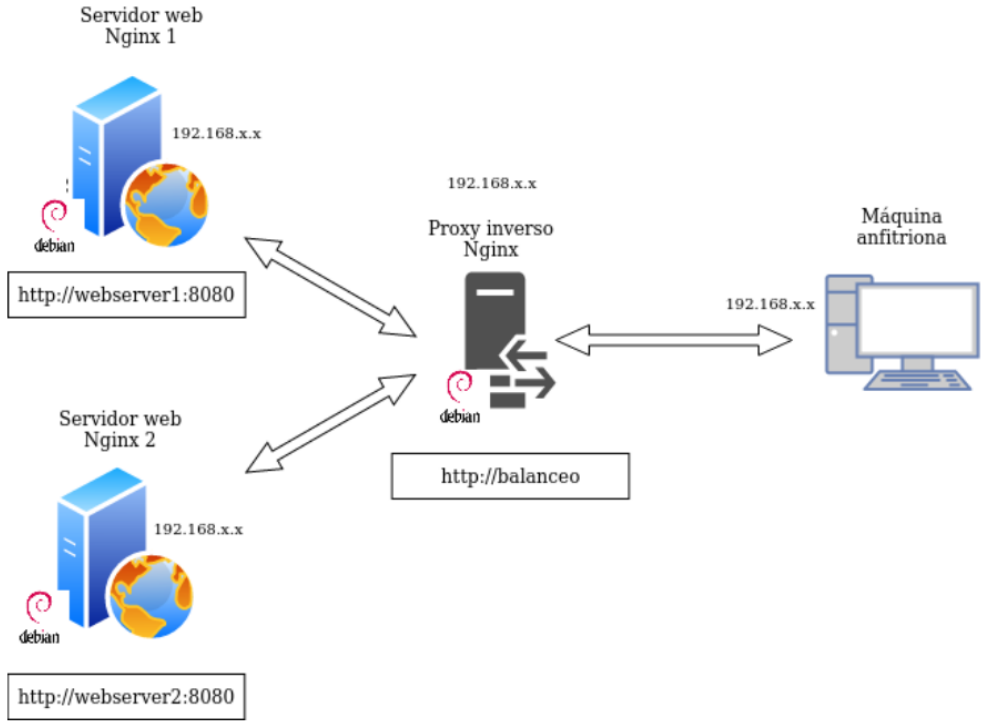
En esta práctica tendremos un escenario donde Nginx hará tanto de proxy inverso como de balanceador de carga al mismo tiempo

2. Tarea

Vamos a configurar **dos servidores web Nginx con dos máquinas Debian**, además de **reutilizar el proxy inverso Nginx configurado en la práctica anterior (EN TOTAL DEBEN HABER 3 MÁQUINAS DEBIAN)**. Partiremos por tanto de la configuración de la práctica anterior, añadiendo lo necesario:

- Cada servidor web presentará un sitio web específico para esta práctica (realmente podéis usar el mismo sitio web, pero cambiad algo del **html** principal)
- El proxy inverso que ya teníamos configurado; ahora habrá que configurarlo para que realice el balanceo de carga que deseamos
- Realizaremos las peticiones HTTP desde el navegador web de nuestra máquina anfitriona.

El diagrama de red quedaría así:



Haremos las peticiones web desde el navegador al proxy inverso, que las repartirá entre los dos servidores web que tenemos.

Accederemos a `http://balanceo` y debemos observar que las peticiones, efectivamente, se van repartiendo entre el servidor 1 y el 2.

2.1 Configuraciones

Ya no vamos a utilizar los sitios web que hemos configurado en las prácticas anteriores. Por ello, para evitarnos una serie de problemas que pueden surgir, vamos a desactivarlos.

Dentro de la carpeta /etc/nginx/sites-enabled debemos ejecutar unlink nombre_archivo para cada uno de los archivos de los sitios web que tenemos.

Si no hacéis esto obtendréis errores en todas las prácticas que quedan de este tema.

Nginx servidor web 1

El primer servidor web será el servidor principal que hemos venido utilizando hasta ahora durante el curso, el original, donde tenemos instalado ya el servicio Web.

Debemos configurar este servidor web para que sirva el siguiente index.html que debéis crear dentro de la carpeta **/var/www/webserver1/html**:

```
<head>
    <title> Prueba de balanceo de carga con Nginx</title>
</head>
<body>
    <h2>Este es el servidor web 1</h2>
    <p>Comprueba el balanceo de carga con Nginx recargando esta página</p>
</body>
</html>
```

- El nombre del sitio web que debéis utilizar en los archivos correspondientes (*sites-available…*) que debéis crear para Nginx es *webserver1*, así como en sus configuraciones. Fijaos en las configuraciones que hicisteis en prácticas anteriores a modo de referencia.
- El sitio web debe escuchar en el puerto 8080.
- Debéis añadir una cabecera que se llame `Serv_Web1_vuestronombre`.

Nginx servidor web 2

Debe ser una máquina Debian, clon del servidor web 1.

En este servidor web debemos realizar una configuración idéntica al servidor web 1 pero cambiando *webserver1* por *webserver2* (también en el *index.html*), así como el nombre de la cabecera añadida, que será `Serv_Web2_vuestronombre`

Nginx Proxy Inverso

Ya disponemos de los dos servidores web entre los que se van a repartir las peticiones que realice el cliente desde el navegador.

Vamos, por tanto, a configurar el proxy inverso para que realice este reparto de peticiones:

- En sites-available debéis crear el archivo de configuración con el nombre *balanceo*

Este archivo tendrá el siguiente formato:

```
upstream backend_hosts {
            random;
            server ________:____;
            server ________:____;
}
        server {
            listen 80;
            server_name ________;
            location / {
                proxy_pass http://backend_hosts;
            }
        }
```

Donde:

- El bloque *upstream* → son los servidores entre los que se va a repartir la carga, que son los dos que hemos configurado anteriormente.
- Si miráis el diagrama y tenéis en cuenta la configuración que habéis hecho hasta ahora, aquí deberéis colocar la IP de cada servidor, así como el puerto donde está escuchando las peticiones web.
- A este grupo de servidores le ponemos un nombre, que es *backend_hosts*
- El parámetro *random* lo que hace es repartir las peticiones HTTP que llegan al proxy inverso de forma completamente aleatoria entre el grupo de servidores que se haya definido en el bloque *upstream* (en nuestro caso sólo hay dos).
    - Pondremos *random* porque es lo más fácil para comprobar que todo funciona bien en la práctica, pero hay diferentes formas de repartir la carga (las peticiones HTTP). Existen otras opciones como: `round-robin`, `least_conn`, `ip_hash`, etc.

2.2 Comprobaciones

Si accedéis a vuestro sitio web, debéis poder seguir accediendo sin problemas.

- Comprobad dándole repetidamente a F5, que accedéis cada vez a uno de los servidores. Se os mostrará el contenido del *index.html* del servidor correspondiente cada vez.
    - Para una doble comprobación, utilizando las herramientas de desarrollador y mostrad que la web que se os muestra coincide con la cabecera que ha añadido el servidor web en la respuesta HTTP.

RECORDAD DESACTIVAR CACHÉ

2.2.1 Comprobación del balanceo de carga cuando cae un servidor

Nuestro balanceador de carga está constantemente monitorizando “la salud” de los servidores web. De esta forma, si uno deja de funcionar por cualquier razón, siempre enviará las solicitudes a los que queden “vivos”. Vamos a comprobarlo:

- Para el servicio Nginx en el servidor web 1 y comprueba, de la misma forma que en el apartado anterior, que todas las solicitudes se envían ahora al servidor web 2
- Tras iniciar de nuevo Nginx en el servidor web 1, repite el proceso con el servidor web 2.

ACTIVIDADES

1. Para esta actividad, es importante que en el vídeo mostréis las 3 máquinas virtuales, así como peticiones para comprobar el funcionamiento del balanceo de carga.

2. Busca información de qué otros métodos de balanceo se pueden aplicar con Nginx y describe al menos 3 de ellos.

3. Si quiero añadir 2 servidores web más al balanceo de carga, describe detalladamente qué configuración habría que añadir y dónde.