NGINX Reverse Proxy y Granja de Servidores IPv6 Only: Conectividad Web Eficiente

Autor: Alejandro Acosta
Coordinación/revisión: Guillermo Cicileo, Carlos MArtínez
Edición: Área de Comunicaciones
Área: Área de Tecnología

Julio 2023

lacnic
www.lacnic.net

Introducción

En el presente trabajo presentaremos una manera muy sencilla de ofrecer acceso Web Dual Stack a una granja de servidores IPv6 Only utilizando NGINX. Con el crecimiento continuo de la red y la adopción gradual del protocolo IPv6, es esencial garantizar la conectividad y accesibilidad para aquellos clientes que utilizan tanto IPv4 como IPv6.

Explicaremos cómo configurar NGINX para admitir acceso web Dual Stack; veremos cómo configurar NGINX como un proxy inverso que escucha tanto en direcciones IPv4 como IPv6, y cómo direccionar correctamente las solicitudes entrantes a los servidores backend que solo tienen direcciones IPv6. Por cierto, lo que estudiaremos en el siguiente artículo es un importante paso para lograr el ansiado ahorro de direcciones IPv4, entre muchos otros beneficios.

¿Qué es un reverse proxy?

Cloudflare define en [1] un Servidor Proxy Inverso o Reverso como:

“Un proxy inverso es un servidor que se sitúa delante de los servidores web y reenvía las solicitudes del cliente (por ejemplo, el navegador web) a esos servidores web. Los proxies inversos suelen implementarse para ayudar a aumentar la seguridad, el rendimiento y la fiabilidad. Para entender mejor cómo funciona un proxy inverso y las ventajas que puede aportar, definamos primero qué es un servidor proxy.”

¿Qué es un servidor proxy?

Nuevamente Cloudflare define en [1] un Servidor Proxy como:

“Un proxy de reenvío, con frecuencia conocido como proxy, servidor proxy o proxy web, es un servidor que se sitúa delante de un grupo de máquinas cliente. Cuando esos ordenadores realizan solicitudes a sitios y servicios en Internet, el servidor proxy intercepta esas peticiones y luego se comunica con los servidores web en nombre de esos clientes, como un intermediario.”

¿Cuáles son los beneficios de un Reverse Proxy?

- Ofrecer IPv4 o IPv6 transparente a clientes provenientes desde Internet servidos desde una granja de servidores IPv6 Only (en esto nos enfocaremos)

- **Escalabilidad:** Al utilizar un proxy inverso, es posible agregar o eliminar servidores backend según sea necesario sin afectar a los usuarios finales. Esto facilita la escalabilidad horizontal de las aplicaciones, lo que permite manejar un mayor número de solicitudes y usuarios simultáneos.

- **Cacheo de contenido estático:** NGINX puede almacenar en caché contenido estático como imágenes, archivos CSS y JavaScript, lo que reduce la carga en los servidores backend y acelera la entrega de contenido a los usuarios. Esto mejora el tiempo de carga de las páginas y reduce el ancho de banda necesario.

- **Seguridad:** NGINX actúa como un punto de entrada a la aplicación, lo que proporciona una capa adicional de seguridad. Puede realizar funciones como filtrado de solicitudes, prevención de ataques DDoS, protección contra inyecciones SQL y autenticación de clientes. Además, NGINX puede habilitar el uso de SSL/TLS para cifrar la comunicación entre los clientes y el servidor backend.

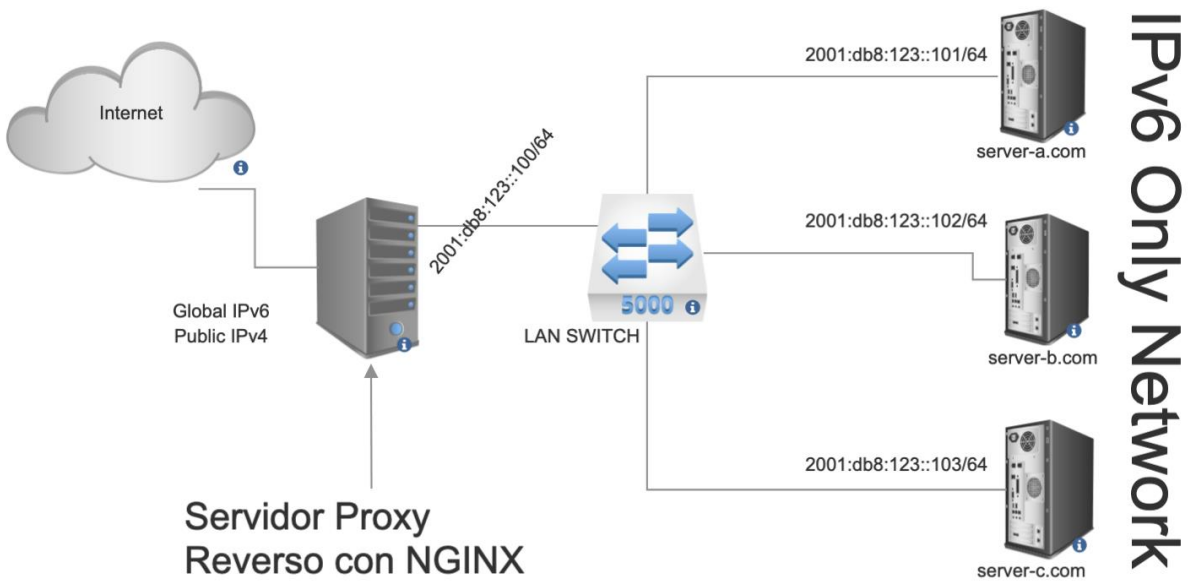
- **Enrutamiento avanzado:** Un proxy inverso permite realizar enrutamiento avanzado según diferentes criterios, como el nombre de dominio, la URL o las cabeceras HTTP. Esto es útil en casos donde se necesite dirigir el tráfico a diferentes servidores backend según las características de la solicitud.

- **Consolidación de servicios:** NGINX puede actuar como un punto de entrada único para varios servicios backend. Esto simplifica la infraestructura al consolidar múltiples servicios en un solo servidor, lo que facilita la administración y el mantenimiento.

- **Mejora del rendimiento**: NGINX está diseñado para ser liviano y eficiente en el uso de recursos. Su arquitectura optimizada y su capacidad para manejar grandes cantidades de conexiones simultáneas lo convierten en una opción popular para mejorar el rendimiento de las aplicaciones web.

- **Balanceo de carga:** Un proxy reverso como NGINX puede distribuir el tráfico entrante a través de varios servidores backend. Esto ayuda a equilibrar la carga de trabajo y garantiza que ningún servidor esté sobrecargado, lo que mejora el rendimiento y la capacidad de respuesta de la aplicación.

Topología que vamos a usar



¿Qué vamos a lograr el día de hoy?

El servidor en el borde (Servidor Proxy Reverso) va a ser capaz de recibir peticiones HTTP en IPv4 e IPv6, y dependiendo del sitio Web que se desea visitar (dominio) re-enviará la consulta al servidor correcto. En el ejemplo actual ocurrirá lo siguiente:

El Cliente visita		Petición enviada a:
server-a.com	→	2001:db8:123::1
server-b.com	→	2001:db8:123::2
server-c.com	→	2001:db8:123::3
server-a.com	→	2001:db8:123::101
server-b.com	→	2001:db8:123::102
server-c.com	→	2001:db8:123::103

Prerrequisitos

- Linux con Nginx en el Servidor Proxy Reverso
- Acceso super usuario
- Servidor Web en cada uno de los servidores de la granja
- Conectividad a Internet IPv4 e IPv6
- Conectividad interna en IPv6

Manos a la obra

1) Instalar nginx en todos los servidores

```
#apt update
#apt install nginx
```

2) Crear los sitios Web en el Proxy Reverso NGINX

Archivo /etc/nginx/sites-available/server-a.com

```
server {
      listen 80;
      listen [::]:80;

    server_name server-a.com;
    location / {
        proxy_pass http://[2001:db8:123::101];
    }
}
```

Archivo /etc/nginx/sites-available/server-b.com

```
server {
      listen 80;
      listen [::]:80;

    server_name server-b.com;
    location / {
        proxy_pass http://[2001:db8:123::102];
    }
}

Archivo  /etc/nginx/sites-available/server-c.com
server {
      listen 80;
      listen [::]:80;

    server_name server-c.com;
    location / {
        proxy_pass http://[2001:db8:123::103];
    }
}
```

3) Crear links simbólicos para habilitar los sitios configurados:

```
root@ProxyReverseSRV:/etc/nginx/sites-enabled# ln -s
/etc/nginx/sites-available/server-a.com /etc/nginx/sites-
enabled/server-a.com

root@ProxyReverseSRV:/etc/nginx/sites-enabled# ln -s
/etc/nginx/sites-available/server-b.com /etc/nginx/sites-
enabled/server-b.com

root@ProxyReverseSRV:/etc/nginx/sites-enabled# ln -s
/etc/nginx/sites-available/server-c.com /etc/nginx/sites-
enabled/server-c.com
```

4) Recordemos reiniciar nginx:

```
$sudo systemctl restart nginx
```

Sobre los logs

Los registros de conexión (logs) son de suma importancia para cualquier empresa e ISP que desea realizar alguna revisión de las conexiones entrantes.

Lo que ocurre es que NGINX por defecto utilizará su propia dirección IP cuando realice las conexiones salientes, lo que trae como consecuencia que se pierde la dirección del cliente que originó la solicitud HTTP. Pero no se preocupen, NGINX tiene la solución, se llama *proxy_set_header* y la configuración se divide en el servidor final y en el servidor Proxy Reverso.

1) En el Servidor Proxy Reverso, archivo del sitio web.

```
# Ejemplo de  nginx reverse proxy que permite conservar la
dirección
# y puerto de original del cliente
location /examples {
  proxy_pass http://[2001:db8:123::103];
  proxy_buffering off;
  proxy_set_header X-Real-IP $remote_addr;
  proxy_set_header X-Forwarded-Host $host;
  proxy_set_header X-Forwarded-Port $server_port;
  proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
```

2) En el servidor final en el archivo: /etc/nginx/nginx.conf agregar en la sección http lo siguiente:

```
   set_real_ip_from 2001:db8:123::100; #sustituir la dirección
IP por la del Proxy
   real_ip_header X-Forwarded-For;
   real_ip_recursive on;
```

Ejemplo:

```
http {
    …
    set_real_ip_from 2001:db8:123::100;
    real_ip_header X-Forwarded-For;
    real_ip_recursive on;
   …
}
```

Luego de estas configuraciones el servidor final confiará en la cabecera llamada X-Forwarded-For que provenga del IP 2001:db8:123::100 y en sus registros (/var/log/nginx/access.log) se podrá apreciar la dirección origen del cliente.

Conclusiones

Se puede percibir que con el diseño propuesto podemos administrar una granja de servidores web 100% IPv6 Only con acceso al mundo tanto IPv4 e IPv6 de una manera muy sencilla, escalable y eficiente. Lo anterior trae consigo diferentes beneficios tales como: administrar solo un stack TCP/IP, simplicidad, seguridad e incluso ahorro de direcciones IPv4.

Referencias

- [1] https://www.cloudflare.com/es-es/learning/cdn/glossary/reverse-proxy/
- https://www.digitalocean.com/community/tutorials/how-to-configure-nginx-as-a-reverse-proxy-on-ubuntu-22-04
- Archivos de configuración de todo el proyecto en el Github de LACNIC: https://github.com/LACNIC/BlogPostHelpFiles/tree/main/2023_Ofreciendo_conectividad_Dual_Stack_a_servidores_Web_en_una_granja_de_servidores_100_IPv6_Only