Balanceamento de carga e failover com servidor Nginx na borda e um farm de servidores Web só IPv6

Autor: Alejandro Acosta
Coordenação e revisão: Guillermo Cicileo, Carlos MArtínez
Edição: Área de Comunicações
Área: Área de Tecnologia

Agosto 2023

lacnic
www.lacnic.net

Introdução

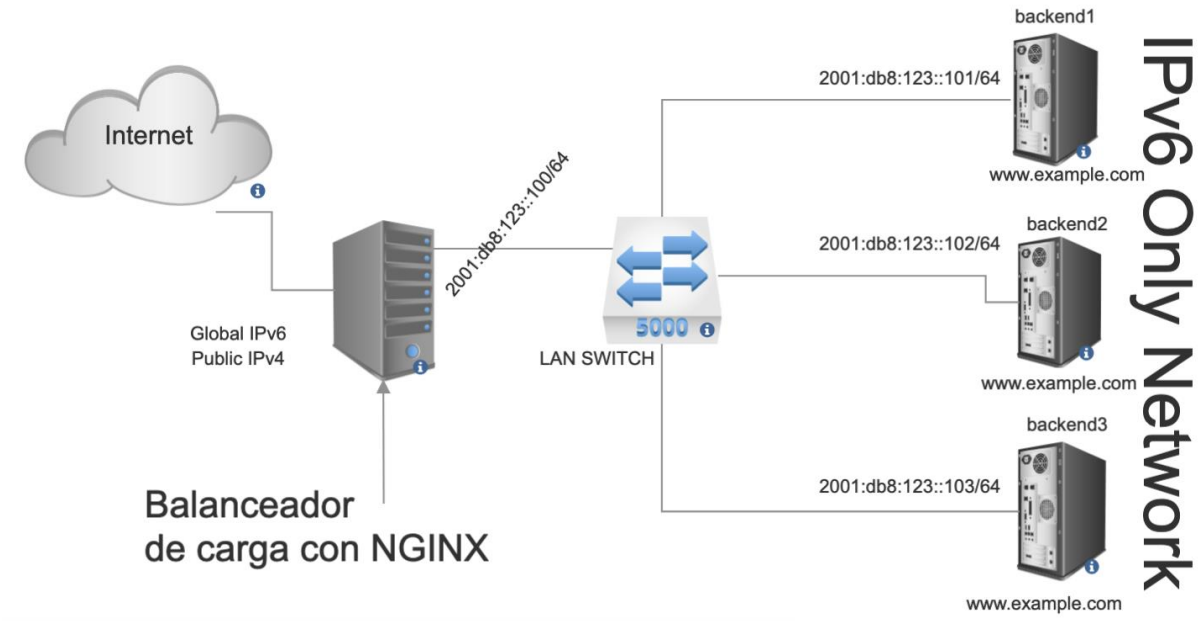
Este trabalho é a continuação do documento NGINX Reverse Proxy e Farm de servidores só IPv6: Conectividade Web Eficiente. Nele, estivemos configurando um Proxy Reverso Nginx no qual, com um servidor, pudemos fornecer acesso à web Dual Stack (IPv4 e IPv6) a partir de um farm de servidores só IPv6. Uma forma muito interessante de economizar endereços IPv4 e obter outro grande número de benefícios.

Neste artigo, vamos explorar como você pode implementar recursos de balanceamento de carga usando um servidor Nginx na borda e um farm de servidores web que operam exclusivamente no IPv6. Vamos descobrir os benefícios dessa configuração e as etapas necessárias para obter uma arquitetura robusta e confiável, bem como os diferentes métodos de implementação.

Por que usar Nginx na borda?

O servidor Nginx é conhecido por seu desempenho, escalabilidade e recursos avançados de balanceamento de carga. Colocar o Nginx na borda da rede permite que você tenha um único ponto de entrada para seus serviços Web, pudendo assim gerenciar e distribuir o tráfego com eficiência para seu farm de servidores só IPv6.

Topologia



Métodos do NGINX para balanceamento de carga

O Nginx tem vários métodos para o balanceamento de carga; a seguir, explicamos cada um deles:

- IP-hash: este método usa um algoritmo que pega o endereço IP de origem e destino do cliente e o servidor para gerar uma chave de hash exclusiva. Isso permite a persistência da sessão.
- Round-robin (padrão): este é o método padrão para o balanceamento de carga. Diz ao balanceador de carga que volte para o topo da lista e se repita de novo.
- "O menos conectado" (least_conn): este método usa um algoritmo de balanceamento de carga dinâmico. Redistribui as conexões para o membro do pool que administra o menor número de conexões abertas no momento em que a nova solicitação de conexão é recebida.

Requisitos Configuração do Nginx para o balanceamento de carga (Servidor Proxy -borda-)

- Instalar o Nginx em seu servidor na borda e verifique que ele esteja configurado corretamente para funcionar com IPv4 e IPv6. Lembre-se de que este servidor pode escutar da rua IPv4 e IPv6, vai fazer proxy das solicitações e vai repassá-las internamente para o farm de servidores só por IPv6
- Criar um arquivo de configuração Nginx e definir o bloco *upstream* com os endereços IPv6 de seus servidores web.
- Configurar os algoritmos de balanceamento de carga, como round-robin, least_conn ou ip_hash, para distribuir as solicitações entre os servidores web do farm.
- Servidor Linux na borda com Nginx instalado e este terá um endereço IPv4 e um endereço IPv6
- É necessário que cada um dos seus servidores web no farm tenha um endereço IPv6 diferente configurado e que este seja acessível desde o servidor Proxy.

Configurações

Configuração no lado do balanceador

```
#Arquivo: /etc/nginx/sites-enabled/example.com
upstream back-end {  #O upstream do farm de servidores é chamado de upstream
    server [2001:db8:123::101]; #server backend1
    server [2001:db8:123::102]; #server backend2
    server [2001:db8:123::103]; #server backend3
}

server {  #esta já é uma diretiva conhecida do Nginx
    listen 80; #porta na qual o servidor web escuta
    server_name example.com www.example.com;  #nome de domínio

    location / {
        proxy_pass http://backend;     #note-se que back-end é o nome do
upstream
    }
}
```

Configuração do lado dos servidores do farm

Todos os servidores de back-end no farm têm a mesma configuração

```
#/etc/nginx/sites-available/default
server {
      listen [::]:80 default_server;
      root /var/www/html;
      index index.html index.htm index.nginx-debian.html;
      server_name _;
      location / {
            try_files $uri $uri/ =404;
      }
}
```

Testes e monitoramento

Depois de fazer as configurações, passamos a testar. A seguir, uma lista de possíveis testes que podem ser realizados:

a) Criar um arquivo diferente em cada servidor do farm e carregue www.example.com desde a Internet. Toda vez que carregamos e recarregamos a página, deveria mostrar uma página para cada servidor na diretiva upstream
b) Revisar os logs no Nginx em cada servidor de back-end, por exemplo, poderia ser feito: *tail -f /var/log/nginx/*.log* e rever os acessos e/ou erros
c) Rever os logs no balanceador, também poderia se usar: *tail -f /var/log/nginx/*.log*
d) Você pode executar este script muito simples para apreciar o round robin:
for ((i=1;i<=10;i++)); do curl -v "http://www.example.com"; sleep 1; done

Configuração do Nginx para o failover e outras opções

O failover no Nginx é tratado passando parâmetros para os servidores de back-end especificados no upstream.

Os diferentes parâmetros são (veja o exemplo abaixo)

weight: Esta opção permite especificar o peso relativo de cada servidor no grupo upstream. Como você já usou no seu exemplo, o peso determina a proporção de pedidos que cada servidor irá tratar em relação aos outros servidores.

max_fails: Esta opção permite especificar o número máximo de tentativas de conexão com um servidor antes de considerá-lo temporariamente não disponível. Por padrão, o valor é 1. Por exemplo, max_fails=3; indica que um servidor será marcado como não disponível após três tentativas malsucedidas de conexão consecutivas.

fail_timeout: Esta opção define o período de tempo em que um servidor será considerado não disponível após atingir o número máximo de tentativas malsucedidas especificado por max_fails. Por padrão, o valor é 10 segundos. Por exemplo, fail_timeout=30s; estabelece um tempo de espera de 30 segundos para um servidor marcado como não disponível.

backup: Esta opção indica que um servidor deve ser usado como reserva ou backup. O servidor marcado como backup só será usado se todos os outros servidores estiverem marcados como não disponíveis.

down: Esta opção marca um servidor como não disponível de forma permanente. O Nginx não enviará solicitações para um servidor marcado como "down", mesmo que todos os outros servidores estejam marcados como não disponíveis. Por exemplo, down; marca um servidor como não disponível.

```
upstream back-end {
    server [2001:db8:123::101] weight=3;
    server [2001:db8:123::102] max_fails=2 fail_timeout=10s;
    server [2001:db8:123::103] backup;
    server [2001:db8:123::104] down;
}
```

Verificar a configuração e reiniciar o servidor para atualizar as mudanças

```
# nginx -t
# systemctl restart nginx
```

Conclusão

Implementar um balanceamento de carga e failover usando um servidor Nginx na borda e um farm de servidores Web só IPv6 oferece uma arquitetura escalável e robusta. Você pode distribuir eficientemente o tráfego de entrada entre seus servidores web e garantir a alta disponibilidade de seus serviços.

Github com os arquivos de configuração usados

https://github.com/LACNIC/BlogPostHelpFiles/tree/main/2023_07_Balanceo_Carga_y_Failover_NGINX_IPv6

Referências

https://help.clouding.io/hc/es/articles/360019908839-C%C3%B3mo-configurar-un-servidor-de-balanceo-de-carga-Nginx-en-Ubuntu-20-04
https://cloud.google.com/load-balancing/docs/https
https://stackoverflow.com/questions/69285690/nginx-load-balancer-configuration-not-working