Automated Nginx Reverse Proxy for Docker

Mar 25, 2014 · 4 minute read · 96 Comments

docker **nginx** **service** **golang** **docker-gen**

A reverse proxy server is a server that typically sits in front of other web servers in order to provide additional functionality that the web servers may not provide themselves.

For example, a reverse proxy can provide SSL termination, load balancing, request routing, caching, compression or even A/B testing.

When running web services in docker containers, it can be useful to run a reverse proxy in front of the containers to simplify depoyment.

Why Use A Reverse Proxy With Docker

Docker containers are assigned random IPs and ports which makes addressing them much more complicated from a client perspsective. By default, the IPs and ports are private to the host and cannot be accessed externally unless they are bound to the host.

Binding the container to the hosts port can prevent multiple containers from running on the same host. For example, only one container can bind to port 80 at a time. This also complicates rolling out new versions of the container without downtime since the old container must be stopped before the new one is started.

A reverse proxy can help with these issues as well as improve availabilty by facilitating zero-downtime deployments.

Generating Reverse Proxy Configs

Setting up a reverse proxy configuration can be complicated when containers are started and stopped. Typically the configuration needs to be updated manually which is error prone and time consuming.

Fortunately, Docker provides a remote API to inspect containers and access their IP, Ports and other configuration meta-data. In addition, it also provides a real-time events API that can be used for notifications when containers are started and stopped. These APIs can be used to generate a reverse proxy config automatically.

docker-gen is a small utility that uses these APIs and exposes container meta-data to templates. Templates are rendered and an optional notification command can be run to restart the service.

Using docker-gen, we can generate Nginx config files automatically and reload nginx when they change. The same approach can also be used for docker log management.

Nginx Reverse Proxy for Docker

This example nginx template can be used to generate a reverse proxy configuration for docker containers using virtual hosts for routing. The template is implemented using the golang text/template package. It uses a custom `groupBy` template function to group the running containers by their `VIRTUAL_HOST` environment variable. This simplifies iterating over the containers to generate a load-balanced backend and also enables zero-downtime deployments.

```
{{ range $host, $containers := groupBy $ "Env.VIRTUAL_HOST" }}
upstream {{ $host }} {
```

```
{{ range $index, $value := $containers }}
    {{ with $address := index $value.Addresses 0 }}
    server {{ $address.IP }}:{{ $address.Port }};
    {{ end }}
{{ end }}

}

server {
    #ssl_certificate /etc/nginx/certs/demo.pem;
    #ssl_certificate_key /etc/nginx/certs/demo.key;

    gzip_types text/plain text/css application/json application/x-javascript
               text/xml application/xml application/xml+rss text/javascript;

    server_name {{ $host }};

    location / {
        proxy_pass http://{{ $host }};
        include /etc/nginx/proxy_params;
    }
}
{{ end }}
```

The template can be run with docker-gen using:

```
docker-gen -only-exposed -watch -notify "/etc/init.d/nginx reload"
templates/nginx.tmpl /etc/nginx/sites-enabled/default
```

- `-only-exposed` - Only use containers that have exposed ports.
- `-watch` - After starting up, watch for docker container events and regenerate the template.
- `-notify "/etc/init.d/nginx reload"` - Reload the nginx config after the template is generated.
- `templates/nginx.tmpl` - The nginx template.
- `/etc/nginx/sites-enabled/default` - Destination file.

This is the rendered template with two containers configured with `VIRTUAL_HOST=demo1.localhost` and one with `VIRTUAL_HOST=demo2.localhost`

```
upstream demo1.localhost {
    server 172.17.0.4:5000;
    server 172.17.0.3:5000;
}

server {
    #ssl_certificate /etc/nginx/certs/demo.pem;
    #ssl_certificate_key /etc/nginx/certs/demo.key;

    gzip_types text/plain text/css application/json application/x-javascript
               text/xml application/xml application/xml+rss text/javascript;

    server_name demo1.localhost;

    location / {
        proxy_pass http://demo.localhost;
        include /etc/nginx/proxy_params;
    }
}

upstream demo2.localhost {
    server 172.17.0.5:5000;
}

server {
    #ssl_certificate /etc/nginx/certs/demo.pem;
    #ssl_certificate_key /etc/nginx/certs/demo.key;

    gzip_types text/plain text/css application/json application/x-javascript
               text/xml application/xml application/xml+rss text/javascript;

    server_name demo2.localhost;
```

```
    location / {
        proxy_pass http://demo2.localhost;
        include /etc/nginx/proxy_params;
    }
}
```

Try It Out

I created a trusted build with this setup to make it easier to try it out:

Run nginx-proxy container:

```
$ docker run -d -p 80:80 -v /var/run/docker.sock:/tmp/docker.sock -t jwilder/nginx-proxy
```

Start your containers with a `VIRTUAL_HOST` environment variables:

```
$ docker run -e VIRTUAL_HOST=foo.bar.com -t ...
```

If you need HTTPS, would like to run `docker-gen` in a separate container from nginx, Websocket support or other features, take a look at the github project for more information.

Conclusion

Generating nginx reverse proxy configs for docker containers can be automated using the Docker APIs and some basic templating. This can simplify deployments as well as improve availability.

While this works well for containers running on a single host, generating configs for remote hosts requires service discovery. Take a look at docker service discovery for a solution to that problem.

Update: There's a few other posts with similar ideas and variations that are worth checking out:

- Using Nginx, Confd, and Docker for Zero-Downtime Web Update - Brian Ketelsen
- Docker Events - Michael Crosby
- Haproxy As A Static Reverse Proxy for Docker Containers - Oskar Hane

samir Patel • 3 years ago

Instead of a sub-domain VHOST can you use a regex match on the URL? For our API we would like all v2 endpoints to go to one container and all v3 to the other.

For example:

http://api.com/v2/users/123 -> Goes to container1 on port 3001

http://api.com/v3/users/123 -> Goes to container2 on port 3002

5 • Reply • Share ›

ismail yenigül ➚ samir Patel • 2 years ago

Hi Samir

Did you find a solution about the following mapping?

http://api.com/v2/users/123 -> Goes to container1 on port 3001

You should use openresty with lua to parse uri and forward to the docker destination.
You can use redis key value to store key=path value=container1IP:Port

• Reply • Share ›

mhawila Jr. ➚ samir Patel • 2 years ago

Did you find a way to do this?

• Reply • Share ›

samir Patel ➚ mhawila Jr. • 2 years ago

Nope I'm still trying to figure this out, it's on the backburner so any help is appreciated

• Reply • Share ›

mhawila Jr. ➚ samir Patel • 2 years ago

I had to resort to using subdomains

• Reply • Share ›

Klevin Cyan ➚ samir Patel • 2 years ago

Maybe use the native nginx image? It supports such functions.

• Reply • Share ›

bketelsen • 5 years ago

BTW - we're looking at integrating SkyDNS with Etcd to help solve service discovery in Docker in a prettier way.

3 • Reply • Share ›

carlivar ➚ bketelsen • 5 years ago

http://www.consul.io/ could conceivably cover both in one shot.

2 • Reply • Share ›

Jason Wilder Mod ➚ bketelsen • 5 years ago

Cool. Definitely a pain point w/ docker right now. Are you referring to Skydock or something else?

• Reply • Share ›

bketelsen ➚ Jason Wilder • 5 years ago

The idea is that services that register using a pre-configured format (using SkyDock or other things like Serf, etc) would be available via DNS queries. It simplifies the discovery part quite a bit. Even removes most of the need for Docker links.

• Reply • Share ›

Ludovic Roguet • 3 years ago

This was a great post. For those interested in running the proxy on a Raspberry Pi I made a version of it just for that: https://fourteenislands.io/... & https://github.com/lroguet/...

2 • Reply • Share ›

jr1 • 2 years ago

no live upstreams while connecting to upstream
the service is confirmed up, the gerated config has no port and simply says "down" after the ip.

1 • Reply • Share ›

chris strutton • 3 years ago

I struggled for a while trying to use the seperate container idea with docker-compose. Because docker-compose mangles the container names I needed to change the command a bit to use enviroment variables. See this SO question which points out what I missed (http://stackoverflow.com/qu.... Thanks again for such a simple solution.

1 • Reply • Share ›

Aaron J Ban • 3 years ago

Wow. Wow, wow, wow. This is excellent. Just gave it a spin and I'm very happy to see that you can run multiple containers under the same `VIRTUAL_HOST` name and the Nginx container just automagically round-robin balances them.

Excellent work yet again Jason, you truly earn the respect the community has for you.

1 • Reply • Share ›

Jason Wilder Mod → Aaron J Ban • 3 years ago

Thanks Aaron!

• Reply • Share ›

Robclancy • 3 years ago

I never comment but have to just for this, this is great! Been trying to set this stuff up and here I just had to run a few commands :D

1 • Reply • Share ›

mbo • 4 years ago

Any chance to run the trusted build with some other user than "root"?

1 • Reply • Share ›

Gilles QUERRET • 4 years ago

Congratulations for this tool, really useful ! Just a question : I've tried this tool with the public Jenkins docker image, which have 2 exposed ports (50000 and 8080). However, it seems that docker-gen only generates the nginx file for one port (50000) and not for the second one. Is there a way to choose which port will be opened ? Or did I miss something ?

1 • Reply • Share ›

Jason Wilder Mod → Gilles QUERRET • 4 years ago

Yes. You can set VIRTUAL_PORT=8080 to specify which port. Only one port can be used currently though.

1 • Reply • Share ›

the genius → Jason Wilder • a year ago

I'd like to know as well if multiple ports is supported now

• Reply • Share ›

Matthew Groves → Jason Wilder • 2 years ago

Is it still the case (2 years later) that only one port can be used?

• Reply • Share ›

Gilles QUERRET → Jason Wilder • 4 years ago

Excellent, thanks for your answer !

• Reply • Share ›

Andrew • 4 years ago

This is awesome. Great job. Gotta love all these blog posts these days that start of with "Here's how to do this fairly complicated thing" and end with "But here's a Docker container that will set it up for you!".

1 • Reply • Share ›

Jason Wilder Mod → Andrew • 4 years ago

Thanks!

• Reply • Share ›

Sunil • 3 months ago

How to use this for following scenario,

Firstly,I've already started nginx proxy container.

I've created 2 service by using docker service in swarm manager

docker service create -e VIRTUAL_HOST=app1.domain.com --name app1 -v ~/app/1.html:/usr/share/nginx/html nginx:latest

docker service create -e VIRTUAL_HOST=app2.domain.com --name app2 -v ~/app/2.html:/usr/share/nginx/html nginx:latest

app1.domain.com >> app1
app2.domain.com >> app2

When try to hit above subdomain it's not working , but it works when I run container not a service

docker run -d -e VIRTUAL_HOST=app1.domain.com --name app1 -v ~/app/1.html:/usr/share/nginx/html

nginx:latest

• Reply • Share ›

Federico Pereira • 3 months ago

Hi, I'm trying to make it work in compose v3. but I declare nothing of version and service works to me. but I can not use the advantages of version 3.

Any suggestions.

cat docker-compose.yml
nginx-proxy:
image: jwilder/nginx-proxy
ports:
- "80:80"
- "443:443"
volumes:
- /var/run/docker.sock:/tmp/docker.sock

---
version: '3.6'
services:
mariadb:
image: mariadb:latest

see more

• Reply • Share ›

Nitin Vavdiya • 6 months ago

Can you please suggest solution for https://stackoverflow.com/q...

• Reply • Share ›

Bruno Seixas • a year ago

Thanks. It was something that me and a friend were trying to understand.

• Reply • Share ›

aboettger • a year ago

There is a typo:

> When running web services in docker containers, it can be useful to run a reverse proxy in front of the containers to simplify depoyment.

`deployment` not `depoyment`

• Reply • Share ›

Ivo • a year ago

Hi, is there a way to specify different "location" directive for particular VIRTUAL_HOST?

• Reply • Share ›

Frans Thamura • 2 years ago

can we mix with the old conf model?

• Reply • Share ›

Anshul Sahu • 2 years ago

i am using macbook m reverse proxy server is working fine but when i run other container using environmental variable virtual host my other container is not running it gives status THE SITE CANT BE REACHED. Please help me fix the problem

• Reply • Share ›

Flower7C3 • 2 years ago

I use Your proxy in my docker compose. With xip.io it is great for local development. Enjoy https://github.com/Flower7C...

• Reply • Share ›

Victor S. Recio • 2 years ago

Hi bro! I'm using docker-gen and i'm trying to assign virtual host to my nginx file but doesn't work, because I can't see the application, for example mydomain.com/app1 and mydomain.com/app2 this is my template file:
{{ range $host, $containers := groupBy $ "Env.VIRTUAL_HOST" }}
upstream {{ $host }} {

{{ range $index, $value := $containers }}
{{ with $address := index $value.Addresses 0 }}
server {{ $address.IP }}:{{ $address.Port }};

{{ end }}
{{ end }}

}

server {
server_name {{ $host }};
{{ range $application, $name := groupBy $containers "Env.NAME_APP" }}
{{ with $name := $application }}
location /{{ $name }} {

see more

• Reply • Share ›

Ivo ➚ Victor S. Recio • a year ago

hi, did you manage to resolve this issue? I am facing the very same problem currently...

• Reply • Share ›

Victor S. Recio ➚ Ivo • a year ago

Sure, I decided for using subdomains!

• Reply • Share ›

Mirko • 2 years ago

HI nginx-proxy using the --net option works great for network overlay. But when I start to separate the container into nginx and docker-gen it is not clear to me how the network overlay should work. I added --net to both (nginx + docker-gen) but the necessary upstream info in not writen into config:

with nginx-proxy:

upstream 18022a7879d6d732e6513ec5406c215cb174615b {
Can be connect with "isolated_nw" network
grafana
server 172.18.0.4:3000;
}

with nginx + docker-gen

upstream 18022a7879d6d732e6513ec5406c215cb174615b {
}

• Reply • Share ›

Nikola Milojević • 2 years ago

Heyo, thanks for this tool!

Is there any possiblity that when I run multiple containers with VIRTUAL_HOST directive it only uses the ip of the last one started?

Thanks

• Reply • Share ›

George Hartzell • 2 years ago

@jason, this looks like a great writeup and a very useful tool.

Just to double check my understanding, do you have a typo in the rendered template example above? I think that this:

proxy_pass http://demo.localhost;

should be this:

proxy_pass http://demo1.localhost;

given the template source and the way that demo2 rendered.

• Reply • Share ›

dan • 2 years ago

Hi,
How I can change the algorithm that is used to a sticky session algorithm?
Thank You

• Reply • Share ›

Ivor Scott • 2 years ago

nice

• Reply • Share ›

Jacob Hämmerle • 2 years ago

Do I understand right:
docker-gen runs on the host. But how does the generated nginx config go inside the nginx container?

^ v • Reply • Share ›

Jacob Hämmerle ➚ Jacob Hämmerle • 2 years ago

I just looked at the dockerfile of the jwilder/nginx-Proxy
Seems like docker-gen runs INSIDE the container. How can a program access the docker API and get all the container Infos of the other containers from inside a container?

^ v • Reply • Share ›

Chris ➚ Jacob Hämmerle • 2 years ago

-v /var/run/docker.sock:/tmp/docker.sock makes the nginx-proxy container can access the docker API.

3 ^ v • Reply • Share ›

ⓥ ⓘ ⓝ ⓒ ⓔ ⓣ ⓢ ⓔ • 3 years ago

Thanks so much for this post, Jason. It really saved my day when I tried to deploy an application with Docker. I needed zero-downtime deployments, and figured out how to do a close approximation of it with Docker Compose that I thought I should share in case someone finds it useful.

https://github.com/vincetse...

katopz ➚ ⓥ ⓘ ⓝ ⓒ ⓔ ⓣ ⓢ ⓔ • 3 years ago

Thanks man!

^ v • Reply • Share ›

廖煜 • 3 years ago

If I only want to monitor create and remove of container, how to change the docker-gen option?

^ v • Reply • Share ›

Santiago García Gil • 3 years ago

Works thank you very much

but if the two containers are different as nginx to know which should go

^ v • Reply • Share ›

Juan Antonio González Cano • 3 years ago

hi,

I runnig docker run -d -p 80:80 -v /var/run/docker.sock:/tmp/docker.sock -t jwilder/nginx-proxy

and next

I'm try to run multiple containers "xcgd/libreoffice"

docker -d -p 8992:8997 - e VIRTUAL_HOST=libreoffice.com xcgd/libreoffice

but

when I try (telnet libreoffie.com 8992) the result es OK but when I try telnet libreoffie.com 80 the result not is OK.

what am I doing wrong?

^ v • Reply • Share ›

Ludovic Roguet ➚ Juan Antonio González Cano • 3 years ago

I know this is 5 months old but what if you try to run your containers like this instead:
`docker -d - e VIRTUAL_HOST=libreoffice.com xcgd/libreoffice`

^ v • Reply • Share ›