PUBLIC | AUTOMATED BUILD

jderusse (/u/jderusse/)/dns-gen (/r/jderusse/dns-gen/) ☆

Last pushed: 2 years ago

Short Description

Short description is empty for this repo.

Full Description

Docker DNS-gen

dns-gen sets up a container running Dnsmasq and docker-gen (https://github.com/jwilder/docker-gen).
docker-gen generates a configuration for Dnsmasq and reloads it when containers are
started and stopped.

By default it will provide thoses hosts: `containername.docker` and `servicename.projectfolder.docker`
pointing to the corresponding container.

Simple usage

First, you have to know the IP of your `docker0` interface. It may be
`172.17.42.1` but it could be something else. To known your IP, run the
following command:

```
$ /sbin/ifconfig docker0 | grep "inet" | head -n1 | awk '{ print $2}' | cut -d: -f2
```

Now, you can start the `dns-gen` container:

```
$ docker run --detach \
  --name dns-gen \
  --publish 172.17.42.1:53:53/udp \
  --volume /var/run/docker.sock:/var/run/docker.sock \
  jderusse/dns-gen
```

Last thing: Register you new DnsServer in you resolv.conf

```
$ echo "nameserver 172.17.42.1" | sudo tee --append /etc/resolvconf/resolv.conf.d/head
$ sudo resolvconf -u
```

This is it. You can now start your containers and retrieve their IP:

```
$ docker run --name my_app --detach nginx
$ dig my_app.docker
$ dig sub.my_app.docker
```

You can customize the DNS name by providing an environment variable, like this:
`DOMAIN_NAME=subdomain.youdomain.com`

```
$ docker run --env DOMAIN_NAME=foo.docker --detach nginx
$ dig foo.docker
$ dig sub.foo.docker
$ docker run --env DOMAIN_NAME=bar.docker,baz.docker --detach nginx
$ dig bar.docker
$ dig baz.docker
```

Start the container automatically after reboot

You can tell docker (version >= 1.2) to automatically start the DNS container
after booting, by passing the option `--restart always` to your `run` command.

```
$ docker run -d --name dns-gen \
  --restart always \
  --publish 172.17.42.1:53:53/udp \
  --volume /var/run/docker.sock:/var/run/docker.sock \
  jderusse/dns-gen
```

beware! When your host will restart, it may change the IP address of
the `docker0` interface.
This small change will prevent docker to start your dns-gen container. Indeed,
remember our container is configured to forward port 53 to the previous
`docker0` interface which may not exist after reboot. Your container just will
not start, you will have to re-create it. To solve this drawback, force docker
to always use the same IP range by editing the default configuration of the docker
daemon (sometimes located in `/etc/default/docker` but may change regarding
your distribution). You have to restart the docker service to take the changes
into account. Sometimes the interface is not updated, you will have to restart
your host.

```
 # For systemd users (Fedora and recent Ubuntu versions) :
$ vim /lib/systemd/system/docker.service
# append the --bip="172.17.42.1/24" option to the ExecStart line
# then
$ sudo systemctl daemon-reload

# For other users
$ vim /etc/default/docker

DOCKER_OPTS="--bip=172.17.42.1/24"

# In any cases
$ sudo service docker restart
```

One more thing When you start your host, the docker service is not fully
loaded.
Until this daemon is loaded, the dns container will not be automatically started
and you will notice bad performance when your host will try to resolve DNS.
The service is not fully loaded, because it uses a feature of systemd called
socket activation (http://0pointer.de/blog/projects/socket-activation.html): The first access to the docker socket will trigger the
start of the true service.
To skip this feature, you simply have to activate the docker service.

```
$ sudo update-rc.d docker enable
```

Et voila, now, docker will really start with your host, it will always
use the same range of IP addresses and will always start/restart the container
dns-gen.

Advanced usage

The previous method is simple to use, but suffer from drawback:

- Containers won't resolve DNS from other containers
- External DNS resolution may be slower
- Some VPN changes the /etc/resolv.conf file which remove your configuration

Instead of using the `docker0` interface and modifying `/etc/resolv.conf`,
an other solution is to install localy a dnsmasq server (some distribs like
ubuntu or debian are now using it by default) and forward requests to the
dns-gen container and configure containers to use it.

step 1 Configure the local dnsmasq to forward request to `127.0.0.1:54`
And listen to interfaces `lo` and `docker0`.

```
$ sudo vim /etc/NetworkManager/dnsmasq.d/01_docker`
bind-interfaces
interface=lo
interface=docker0

server=/docker/127.0.0.1#54

$ sudo systemctl status NetworkManager
```

step 2 Run dns-gen and bind port `53` to the `54` 's host

```
$ docker run --daemon --name dns-gen \
  --restart always \
  --publish 54:53/udp \
  --volume /var/run/docker.sock:/var/run/docker.sock \
  jderusse/dns-gen -R
```

> the option `-R` just tell dns-gen to not fallback to the default resolver
> which avoid an infinity loop of resolution

step 3 Configure docker to use the `docker0` as DNS server

```
# For systemd users (Fedora and recent Ubuntu versions) :

$ vim /etc/systemd/system/docker.service

# append
[Service]
ExecStart=
ExecStart=/usr/bin/docker daemon -H fd:// --bip=172.17.42.1/24 --dns=172.17.42.1

# then
$ sudo systemctl daemon-reload

# For other users
$ vim /etc/default/docker
DOCKER_OPTS="--dns=172.17.42.1 --bip=172.17.42.1/24"

# In any cases
$ sudo service docker restart
```

Thank to this configuration the resolution workflow is now:

- the host want to resolve `google.com` : `host -> dnsmasq -> external dns`
- the host want to resolve `foo.docker` : `host -> dnsmasq -> 127.0.0.1:54 -> dns-gen`
- a container want to resolve `google.com` : `container -> 172.17.42.1 -> dnsmasq -> external dns`
- a container want to resolve `foo.docker` : `container -> 172.17.42.1 -> dnsmasq -> 127.0.0.1:54 -> dns-gen`

Docker Pull Command

```
docker pull jderusse/dns-gen
```

Owner

jderusse

Source Repository

jeremy-derusse/docker-dns-gen (https://github.com/jeremy-derusse/docker-dns-gen)