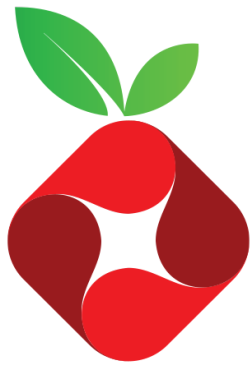
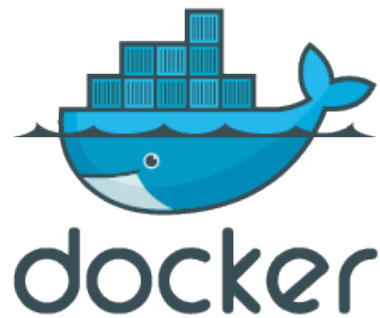


Step-by-Step Install Guide for Pi-hole with Traefik on Docker Swarm



Pi-hole®

This guide will walk you through setting up **Pi-hole**, a powerful network-wide ad blocker, on a Docker Swarm with **Traefik** as the reverse proxy. We will use Docker to deploy Pi-hole and Traefik for secure and managed access.

Prerequisites

- A Docker Swarm environment with at least one Linux node. -> [Check this Article](#)
- Traefik set up as a reverse proxy (Assuming a Traefik stack is already configured). -> [Check this Article](#)
- A mounted shared volume with **GlusterFS** for data persistence. -> [Check this Article](#)
- A domain or local DNS entry pointing to your Pi-hole service (e.g., `pihole.local`).
- Access to Docker CLI and necessary credentials.

Step 1: Prepare the Directory Structure

To ensure persistent storage for Pi-hole, we will create directories on the shared GlusterFS mount:

```
mkdir /mnt/glustermount/data/pihole_data
mkdir /mnt/glustermount/data/pihole_data/dns
mkdir /mnt/glustermount/data/pihole_data/etc
```

These directories will store Pi-hole's configuration files and DNS settings.

Step 2: Create the `docker-compose.yml` File

In your working directory, create a `docker-compose.yml` file with the following content:

```
version: '3'

services:
  pihole:
    networks:
      - management_net # For management via Traefik
    image: pihole/pihole:latest
    ports:
```

- "53:53/tcp"
- "53:53/udp"
- "888:80"

environment:

TZ: 'Europe/Zurich'

WEBPASSWORD: '\${PIHOLE_PASSWORD}'

volumes:

- '/mnt/gluster mount/data/pihole_data/etc:/etc/pihole'
- '/mnt/gluster mount/data/pihole_data/dns:/etc/dnsmasq.d'

restart: unless-stopped

deploy:

mode: replicated

replicas: 1

placement:

constraints: [node.platform.os == linux]

labels:

- 'traefik.enable=true'
- "traefik.http.services.pihole.loadbalancer.server.port=888"

networks:

management_net:

external: true

- **Volumes:** The `/mnt/gluster mount/data/pihole_data/etc` and `/mnt/gluster mount/data/pihole_data/dns` directories are used to persist Pi-hole data.
- **Ports:** Port `53` for DNS queries is exposed on both TCP and UDP.
- **Environment Variables:** Set the timezone (`TZ`) and the Pi-hole admin password (`WEBPASSWORD`).
- **Traefik Labels:** These labels enable Pi-hole to be accessible through Traefik via the domain `pihole.local` using HTTPS.

Step 3: Deploy the Stack

Deploy the Stack: Use the following command to deploy Pi-hole with Traefik in Docker Swarm.

```
docker stack deploy -c docker-compose.yml pihole
```

Step 4: Access Pi-hole Web Interface

After the deployment completes, you can access Pi-hole's admin interface by navigating to `https://pihole.local/admin` in your browser. Log in with the password you specified in the environment variable (`PIHOLE_PASSWORD`).

Step 5: Update DNS Settings

To start using Pi-hole, configure your router or devices to use your Pi-hole instance as the DNS server. The IP address of your Pi-hole service is the one assigned by Docker, which you can retrieve using:

```
docker service ps pihole_pihole
```

OR if you use Keepalived you can use your VIP.

Conclusion

Setting up Pi-hole in a Docker Swarm environment with Traefik as a reverse proxy provides network-wide ad blocking, improving privacy and performance for all devices connected to your network. By leveraging Docker Swarm and Traefik, you achieve high availability, flexibility, and ease of management for your Pi-hole deployment.

Revision #10

Created 8 September 2024 17:55:02 by aeoneros

Updated 10 October 2024 23:16:00 by aeoneros