Traefik

Simplify Your Homelab with Traefik: SSL Certs and FQDN for Local Subdomains

If you’re running a homelab, you’ve probably struggled with organizing your services using IP addresses or clunky port numbers. This is where **Traefik**, a powerful reverse proxy, can change the game. Not only does it streamline the management of local services by providing human-readable fully qualified domain names (FQDNs), but it also makes securing your subdomains with **SSL certificates** incredibly easy. In this post, we'll dive into how you can use Traefik to enhance your homelab experience, explore its features, and walk through setting it up using **Docker Compose**.

Why Use Traefik in a Homelab?

Traefik stands out as a reverse proxy due to its simplicity, flexibility, and native integration with Docker. Here are some compelling reasons to choose Traefik for your homelab:

Key Features:

1. **Automated SSL Certificate Management**
   Traefik integrates seamlessly with Let's Encrypt, allowing you to automatically generate and renew SSL certificates for all your services without manual intervention.
2. **Dynamic Reverse Proxy Configuration**
   Traefik can automatically detect and configure new containers in your homelab, thanks to its auto-discovery feature. It handles load balancing and routes traffic based on your setup.
3. **Easy-to-Use Dashboard**
   The Traefik dashboard provides a visual interface to monitor your services, routes, certificates, and more.

4. **Human-Readable FQDNs**
   Instead of accessing services via IP and port, Traefik lets you set up FQDNs for local subdomains (e.g., `service.mylab.local`). This simplifies navigation within your homelab and gives a professional feel to your internal network.
5. **Seamless Integration with Docker**
   Traefik can act as a gateway to multiple containers, orchestrating how traffic is handled within Dockerized environments. No more manually configuring each container!

Common Use Cases

- **Internal Web Services:** If you're running multiple web-based services such as media servers, home automation, or development environments, Traefik helps route requests efficiently and provides SSL-secured connections.
- **Internal DNS Management:** By giving each service a human-readable subdomain, navigating your homelab becomes much easier. Access services by typing `service.mylab.local` instead of remembering ports.
- **Enhanced Security:** Traefik automatically manages SSL certificates, ensuring all traffic between your browser and services is encrypted.

Setting Up Traefik with Docker Compose

To make Traefik work for your homelab, we'll use Docker Compose to deploy it. Below is an example Docker Compose file to get you started.

Step 1: Basic Traefik Docker-Compose Configuration

```
services:
  traefik:
    image: traefik:v2.9
    container_name: traefik
    restart: unless-stopped
    command:
      - "--api.insecure=true" # Enable the dashboard
```

```
      - "--providers.docker=true" # Enable Docker provider
      - "--entrypoints.web.address=:80"
      - "--entrypoints.websecure.address=:443"
      - "--certificatesresolvers.mytlschallenge.acme.tlschallenge=true"
      - "--certificatesresolvers.mytlschallenge.acme.email=your-email@example.com"
      - "--certificatesresolvers.mytlschallenge.acme.storage=/letsencrypt/acme.json"
    ports:
      - "80:80"   # HTTP
      - "443:443" # HTTPS
      - "8080:8080" # Traefik dashboard
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock:ro"
      - "./letsencrypt:/letsencrypt"
    networks:
      - traefik-public

networks:
  traefik-public:
    external: true
```

Step 2: Configure Docker Labels for a Service

Once Traefik is running, you need to configure Docker labels for the services you want to route. Here’s an example of a service configuration:

```
services:
  webapp:
    image: nginx
    container_name: webapp
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.webapp.rule=Host(`webapp.mylab.local`)"
      - "traefik.http.routers.webapp.entrypoints=websecure"
      - "traefik.http.routers.webapp.tls.certresolver=mytlschallenge"
    networks:
      - traefik-public
```

```
networks:
  traefik-public:
    external: true
```

Step 3: Access Your Service

Once the service is up, visit `https://webapp.mylab.local` in your browser, and Traefik will handle the routing, SSL certificate, and more!

Additional Configuration for Local Subdomains

To resolve local subdomains (like `webapp.mylab.local`), you'll need to configure your local DNS settings or use a DNS resolver like **dnsmasq** to route requests from your machine to the right services. For a homelab, adding records to your `/etc/hosts` file is an easy approach for smaller setups.

Example `/etc/hosts` entry:

```
127.0.0.1   webapp.mylab.local
```

Enabling HTTPS Locally

If you're setting up Let's Encrypt certificates locally for services that are not exposed to the public internet, make sure your local network can validate Let's Encrypt's ACME challenges. Alternatively, you can use self-signed certificates or a local CA authority like **mkcert**.

Wrapping Up

With Traefik, managing and securing services in your homelab becomes a streamlined, efficient process. Not only does it simplify traffic routing with FQDNs, but its built-in SSL certificate management makes securing services a breeze. Whether you're running just a few containers or managing a more complex setup, Traefik provides the tools needed for a professional, scalable infrastructure. Try it out and take your homelab to the next level!

For more details, check out Traefik's official documentation.

Revision #1
Created 2024-09-17 18:25:20 UTC by thesabear
Updated 2024-09-17 18:30:41 UTC by thesabear