

Docker-compose with Let's Encrypt: DNS Challenge & Cloudflare (Recommended)



Introduction

This guide aims to demonstrate how to create a certificate with the Let's Encrypt DNS-01 Challenge to use HTTPS on a simple service exposed with Traefik.

Why Use DNS-01 Challenge Instead of TLS?

Using the DNS-01 Challenge instead of TLS-ALPN-01 provides several advantages:

1. **Wildcard Certificates:** DNS-01 is the only challenge type that supports wildcard domains (e.g., `*.example.com`), simplifying the management of multiple subdomains.
2. **No Open Ports Required:** Unlike TLS-ALPN, DNS-01 doesn't rely on ports 443 or 80 being open, offering better security and flexibility for non-web services.
3. **Multi-Server and Complex Setups:** DNS-01 works in distributed or multi-server environments without requiring direct communication from Let's Encrypt to each server.
4. **Behind Proxies or Firewalls:** Ideal for servers behind NAT, private networks, or reverse proxies where direct access to the server is limited.

Overview of DNS-01 Challenge

The DNS-01 Challenge is a method used by Let's Encrypt to verify domain ownership by creating a specific DNS TXT record for the domain. This challenge is particularly useful for issuing wildcard certificates (`*.example.com`), securing services that are not publicly accessible, and environments where direct access to ports 80 or 443 is not possible. It provides flexibility and higher security, especially when combined with automated DNS providers like Cloudflare.

Prerequisite

- A publicly accessible host allowing connections on port `443` with Docker and Docker Compose installed.
- A working DNS provider (e.g., Cloudflare) with credentials to create and remove DNS records.
- If you use a 3rd-party hosting provider, make sure your domain uses Cloudflare's nameservers. Learn more about changing nameservers in [this guide](#).

Step 0: Add Domain to Cloudflare

To add your domain to the Cloudflare dashboard, follow Cloudflare's full setup guide: [Cloudflare Guide](#).

Step 1: Change Nameservers to Cloudflare

Log in to your hosting provider's control panel and change your domain's nameservers to Cloudflare's. This process may take up to 24 hours to propagate. For more information or an Example, check [this guide](#).

When changing the NS, most Hosting Providers take up to 24h to update them.
Make Sure to ask your Hoster for more Informations

Step 2: Create a Custom Cloudflare API Token

Log in to Cloudflare and navigate to [API Tokens](#). Click "Create Token".

API Tokens

Manage access and permissions for your accounts, sites, and products

Create Token

Steps to create the token:

1. Give your token a descriptive name (e.g., "Homelab Traefik API-Token").
2. Set the permissions as shown in the example image (Zone → Zone [Read], Zone → Zone Settings [Read], Zone → DNS [Edit]).
3. Select your domain.
4. Click "Continue to Summary" and create the token.

User API Tokens

[← Back to view all tokens](#)

Create Custom Token

Token name

Give your API token a descriptive name.

Permissions

Select edit or read permissions to apply to your accounts or websites for this token.

Zone	Zone	Read	×
Zone	Zone Settings	Read	×
Zone	DNS	Edit	×

[+ Add more](#)

Zone Resources

Select zones to include or exclude.

Include	Specific zone	aeoneros.com
---------	---------------	--------------

[+ Add more](#)

Client IP Address Filtering

Select IP addresses or ranges of IP addresses to filter. This filter limits the client IP addresses that can use the API token with Cloudflare. By default, this token will apply to all addresses.

Operator	Value
Select item...	e.g. 192.168.1.88

[+ Add more](#)

TTL

Define how long this token will stay active.

Start Date → End Date

Cancel

Continue to summary

What Does These Permissions do?

Zone → Zone (Read):

Allows read-only access to basic information about the specified zone (e.g., DNS records, configuration, status, etc.).

Zone → Zone Settings (Read):

Grants read-only access to view the zone's settings (e.g., SSL/TLS settings, security settings, and performance configurations).

Zone → DNS (Edit):

Allows full control over DNS records in the specified zone. This means you can create, edit, and delete DNS records.

Step 3: Create a Local API Token File

In This Step you could also Work with Docker Secrets. I did not get it Working with the Docker Secret, that why i put it in a File. If you want to use a Docker Secret, please visit Official Docker Docs: <https://docs.docker.com/compose/how-tos/use-secrets/>

Go to your Traefik folder and create a file to store your Cloudflare API token securely:

```
nano ./traefik_data/cloudflare_api_token
```

Paste your token in the file, then save and close it (CTRL+O, CTRL+X).

Step 4: Create the ACME.json File

The `acme.json` file stores Let's Encrypt certificates, keys, and account information in JSON format for Traefik's ACME integration.

Ensure the file is created and secured with the Rights 600:

```
mkdir ./letsencrypt  
touch ./letsencrypt/acme.json  
chmod 600 ./letsencrypt/acme.json
```

Step 5: Adjust Your Traefik Docker-Compose File

In This Step you could also Work with Docker Secrets. I did not get it Working with the Docker Secret, that why i put it in a File. If you want to use a Docker Secret, please visit Official Docker Docs: <https://docs.docker.com/compose/how-tos/use-secrets/>

Mount your Cloudflare API token file and set the appropriate environment variables:

Example Docker Compose Configuration:

```
services:
  traefik:
    image: "traefik:v3.3"
    volumes:
      - "/traefik_data/cloudflare_api_token:/cloudflare_api_token:ro"
      - "/letsencrypt:/letsencrypt"
    environment:
      - TZ=Europe/Zurich
      - CF_API_EMAIL=yourmail@example.com
      - CF_DNS_API_TOKEN_FILE=/cloudflare_api_token
```

Step 6: Adjust your Traefik Config

You can also use a `static.yaml` or `static.toml` file for this configuration. Check out [this guide](#) for more information.

You can Change the Value "**leresolver**" to your own Resolvername. Make sure to adjust your Traefik Router Labels/Settings.
Make Sure to Adjust the Variable "**yourmail@example.com**" to your actual Email.

Example for Traefik CLI:

```
command:
  - --certificatesresolvers.leresovler.acme.email=yourmail@example.com
  - --certificatesresolvers.leresovler.acme.storage=/letsencrypt/acme.json
  - --certificatesresolvers.myresolver.acme.caserver=https://acme-v02.api.letsencrypt.org/directory
  - --certificatesresolvers.myresolver.acme.dnschallenge.provider=cloudflare
  - --certificatesresolvers.leresovler.acme.dnschallenge.delaybeforecheck=0
  - --certificatesresolvers.leresovler.acme.dnschallenge.resolvers=1.1.1.1:53,1.0.0.1:53,8.8.8.8:53
```

Example for a static.yaml:

```
certificatesResolvers:
  leresovler:
    acme:
      email: 'yourmail@example.com'
      storage: '/letsencrypt/acme.json'
      caServer: 'https://acme-v02.api.letsencrypt.org/directory'
      dnsChallenge:
        provider: 'cloudflare'
        delayBeforeCheck: '0'
      resolvers:
        - '1.1.1.1:53'
        - '1.0.0.1:53'
        - '8.8.8.8:53'
```

Example for static.toml:

```
[certificatesResolvers.leresolver.acme]
email = "yourmail@example.com"
storage = "/letsencrypt/acme.json"
caServer = "https://acme-v02.api.letsencrypt.org/directory"

[certificatesResolvers.leresolver.acme.dnsChallenge]
provider = "cloudflare"
delayBeforeCheck = 0
resolvers = ["1.1.1.1:53", "1.0.0.1:53", "8.8.8.8:53"]
```

Step 7: Restart Traefik

Restart Traefik and make sure your application routers are configured to use the `leresolver` for certificate generation.

Have fun Troubleshooting. You can Change your Traefik-Log-Level to DEBUG for further Troubleshooting.

Additional Tips

To improve security, you can add a plugin to allow only Cloudflare traffic to access your services. Check out [this guide](#).

Revision #4

Created 12 September 2024 14:14:04 by aeoneros

Updated 11 February 2025 12:49:21 by aeoneros