SSL

Let's Encrypt SSL Certificates

Let’s Encrypt uses the ACME protocol to verify that you control a given domain name and to issue you a certificate. To get a Let’s Encrypt certificate, you’ll need to choose a piece of ACME client software to use.

The ACME clients are offered by third parties. Let’s Encrypt does not control or review third party clients and cannot make any guarantees about their safety or reliability.

The container `acme-openemail` container is based on ACME Tiny. It will try to obtain a valid LE certificate automatically for your domains configured in Openemail.

> **Warning**
>
> Openemail **must** be available on port 80 for the acme-client to work.

By default, which means **0 domains** are added to Openemail, it will try to obtain a certificate for `${OPENEMAIL_HOSTNAME}`

For each domain you add, it will try to resolve `autodiscover.ADDED_MAIL_DOMAIN` and `autoconfig.ADDED_MAIL_DOMAIN` to your servers IPv4 address. If it succeeds, these names will be added as SANs to the certificate request.

You can skip the IP verification by adding `SKIP_IP_CHECK=y` to `openemail.conf`. Be warned that a misconfiguration will get you ratelimited by Let's Encrypt! This is primarily useful for multi-IP setups where the IP check would return the incorrect source IP. Due to using dynamic IPs for `acme-openemail`, source NAT is not consistent over restarts.

If you have added an A record for `autodiscover` but omitted `autoconfig`, the client will only validate `autodiscover` and skip `autoconfig` then.

For every domain you remove, the certificate will be moved and a new certificate will be requested. It is not possible to keep domains in a certificate, when we are not able validate the challenge for those.

If you want to re-run the ACME client, use

```
docker-compose restart acme-openemail
```

Additional Domain Names

Edit `openemail.conf` and add a parameter `ADDITIONAL_SAN` like this:

```
ADDITIONAL_SAN=cert1.example.org,cert1.example.com,cert2.example.org,cert3.example.
```

Each name will be validated against its IPv4 address.

Run `docker-compose up -d` to recreate changed containers.

```
docker-compose up -d
```

Skip Let's Encrypt Function

Change `SKIP_LETS_ENCRYPT=y` in `openemail.conf` and restart the stack by running

```
docker-compose down && docker-compose up -d
```

Using Your Own Certificates

To use your own certificates, just save the combined certificate (containing the certificate and intermediate CA/CA if any) to `data/assets/ssl/cert.pem` and the corresponding key to `data/assets/ssl/key.pem`.

Restart changed containers by running

```
docker-compose up -d
```

Check Your Configuration

To find out why a validation fails, run:

```
docker-compose logs acme-openemail
```

To check if `nginx` serves the correct certificate, simply use a browser of your choice and check the displayed certificate.

To check the certificate served by dovecot or postfix we will use `openssl`:

1. Connect via SMTP (25)

```
openssl s_client -starttls smtp -crlf -connect mail.openemail.io:25
```

2. Connect via SMTPS (465)

```
openssl s_client -showcerts -connect mail.openemail.io:465
```

3. Connect via SUBMISSION (587)

```
openssl s_client -starttls smtp -crlf -connect mail.openemail.io:587
```