Auto configure HTTPS with Let's Encrypt on Docker

If you are migrating your passbolt instance from HTTP to HTTPS, you must ensure **all of your users have a backup of their private key**.
As for passbolt browser extension, domain will change from HTTP to HTTPS, it will trigger **an account recovery**.

Important requirement: This tutorial assumes your machine has a valid domain name assigned in order to work with let's encrypt. If you want to use user provided certificates or self-signed certificates skip to the next section

Note: the configuration does not support serving passbolt on a subdirectory fashion. For example, scenarios like https://mydomain.com/passbolt are not supported by default

Requirements

- docker setup with docker-compose
- Firewall rules implemented
- A domain name reachable over the internet

Add traefik service to handle https

As explained in our installation documentation, you should have defined **db** and **passbolt** services for your passbolt stack.

To handle HTTPS setup with Let's Encrypt, add a traefik service as follow:

```
version: '3.7'
services:
  db:
    ...
  passbolt:
```

```
  traefik:
    image: traefik:v2.5.5
    restart: always
    ports:
      - 80:80
      - 443:443
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - ./traefik.yaml:/traefik.yaml:ro
      - ./conf/:/etc/traefik/conf
      - ./acme.json:/acme.json
```

Traefik will:

- act as a proxy in front of passbolt service, that's why we defined ports 80 and 443.
- handle Let's Encrypt certificates renew.

volumes

Create a **traefik.yaml** configuration file with this content (replace yourname@domain.tld with your email for Let's Encrypt):

```
global:
  sendAnonymousUsage: false
log:
  level: INFO
  format: common
providers:
  docker:
    endpoint: 'unix:///var/run/docker.sock'
    watch: true
    exposedByDefault: true
    swarmMode: false
  file:
    directory: /etc/traefik/conf/
    watch: true
api:
  dashboard: false
  debug: false
  insecure: false
entryPoints:
  web:
```

```
      http:
        redirections:
          entryPoint:
            to: websecure
            scheme: https
            permanent: true
    websecure:
      address: ':443'
    web-non-root:
      address: ':8080'
      http:
        redirections:
          entryPoint:
            to: websecure
            scheme: https
            permanent: true
    websecure-non-root:
      address: ':4443'
  certificatesResolvers:
    letsencrypt:
      acme:
        email: yourname@domain.tld
        storage: acme.json
        caServer: 'https://acme-v02.api.letsencrypt.org/directory'
        keyType: EC256
        httpChallenge:
          entryPoint: web
        tlsChallenge: {}
```

Create a **conf** folder:

```
mkdir conf
```

In the **conf** folder, create 2 files:

conf/headers.yaml:

```
http:
  middlewares:
    SslHeader:
      headers:
        FrameDeny: true
```

```
    AccessControlAllowOriginList:
      - origin-list-or-null
    AccessControlMaxAge: 100
    AddVaryHeader: true
    BrowserXssFilter: true
    ContentTypeNosniff: true
    ForceSTSHeader: true
    STSIncludeSubdomains: true
    STSPreload: true
    ContentSecurityPolicy: default-src 'self' 'unsafe-inline'
    CustomFrameOptionsValue: SAMEORIGIN
    ReferrerPolicy: same-origin
    PermissionsPolicy: vibrate 'self'
    STSSeconds: 315360000
```

conf/tls.conf:

```
tls:
  options:
    default:
      minVersion: VersionTLS12
      sniStrict: true
      curvePreferences:
        - CurveP521
        - CurveP384
      cipherSuites:
        - TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
        - TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
        - TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
```

traefik.yaml, **conf/headers.yaml** and **conf/tls.yaml** will be mounted inside traefik container. There is no need to create an **acme.json** file, it will be filled automatically.

Handle passbolt with Traefik

To make Traefik redirect incoming requests to passbolt, edit the **passbolt** service as follow:

Step 1. As traefik will handle HTTPS connexion, remove the ports definition

Step 2. Add docker labels to make Traefik aware of passbolt service

```
services:
  db:
    ...
  passbolt:
    ...
    labels:
      traefik.enable: "true"
      traefik.http.routers.passbolt-http.entrypoints: "web"
      traefik.http.routers.passbolt-http.rule:
"Host(`passbolt.domain.tld`)"
      traefik.http.routers.passbolt-http.middlewares:
"SslHeader@file"
      traefik.http.routers.passbolt-https.middlewares:
"SslHeader@file"
      traefik.http.routers.passbolt-https.entrypoints: "websecure"
      traefik.http.routers.passbolt-https.rule:
"Host(`passbolt.domain.tld`)"
      traefik.http.routers.passbolt-https.tls: "true"
      traefik.http.routers.passbolt-https.tls.certresolver:
"letsencrypt"
  traefik:
    ...
```

Caveats:

Ensure you have correctly set your domain name (replace passbolt.domain.tld twice in the example above).

If you are using non-root images, replace **web** and **websecure** entrypoints with **web-non-root** and **websecure-non-root**.

And that's it you should be able to reach your server on the domain you specified.

Last updated

This article was last updated on December 16th, 2021.

Are you experiencing issues when installing passbolt?

Ask the community!