Self host password manager with Vaultwarden and Traefik

https://blog.puvvadi.me/posts/selfhost-paswword-manager-vaultwarden-traefik/

https://tech.aufomm.com/deploy-bitwarden-with-docker-and-traefik/

Vaultwarden is light weight feature rich drop in replacement for Bitwarden server. It’s essentially debloated version of the Bitwarden.

To use Vaultwarden, SSL is required. Otherwise, singing in to the server is impossible. That’s where traefik comes in. Here I’m assuming you have a domain, cloudflare account & domain added to your account, and docker is already set up and ready to go.

Cloudflare

Please keep cloudflare’s `Email`, `API KEY` or `API TOKEN` ready.

Traefik setup

Create required files

directory structure

```
```

```
touch docker-compose.yml
touch config.yml
mkdir data && cd data
touch acme.json
touch traefik.yml
```

Directory structure should be like this

```
```

```
├── docker-compose.yml
├── config.yml
└── data
    ├── acme.json
    └── traefik.yml
```

Docker network

Create a network with following

```
```

```
docker network create -d bridge proxy
```

Docker compose

First open `docker-compose.yml` and add following

```
```

```
version: '3'

services:
  traefik:
    image: traefik:latest
    container_name: traefik
    restart: unless-stopped
    security_opt:
      - no-new-privileges:true
    networks:
      - proxy
    ports:
      - 80:80
      - 443:443
    dns:
      - 1.1.1.1
      - 8.8.8.8
    environment:
      - CF_API_EMAIL=email@example.com
    # - CF_API_KEY= # use either api key or api token based on you usecase
    #   - CF_API_TOKEN=
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - /home/user/traefik/data/traefik.yml:/traefik.yml:ro
      - /home/user/traefik/data/acme.json:/acme.json
      - /home/user/traefik/config.yml:/config.yml:ro
    labels:
      - "traefik.enable=true"
      # http entrypoint
      - "traefik.http.routers.traefik.entrypoints=http"
      # Dashboard
      - "traefik.http.routers.traefik.rule=Host(`traefik.internal.example.net`)"
      # To create a user:password pair, the following command can be used:
      # echo $(htpasswd -nb user password) | sed -e s/\\$/\\$\\$/g
      - "traefik.http.middlewares.traefik-auth.basicauth.users=<user & password>"
      # redirect middleware
      - "traefik.http.middlewares.traefik-https-redirect.redirectscheme.scheme=https"
      - "traefik.http.middlewares.sslheader.headers.customrequestheaders.X-Forwarded-Proto=https"
      - "traefik.http.routers.traefik.middlewares=traefik-https-redirect"
      # https entrypoint
      - "traefik.http.routers.traefik-secure.entrypoints=https"
      - "traefik.http.routers.traefik-secure.rule=Host(`traefik.internal.example.net`)"
      - "traefik.http.routers.traefik-secure.middlewares=traefik-auth"
      - "traefik.http.routers.traefik-secure.tls=true"
      - "traefik.http.routers.traefik-secure.tls.certresolver=cloudflare"
      - "traefik.http.routers.traefik-secure.tls.domains[0].main=internal.example.net"
      - "traefik.http.routers.traefik-secure.tls.domains[0].sans=*.internal.example.net"
      - "traefik.http.routers.traefik-secure.service=api@internal"

networks:
  proxy:
    external: true
```

> dns records should already pointed to you docker host. e.g. if docker host ip is `10.20.20.5` A record for `traeif.internal` should point to `10.20.20.5`.

traefik config

```
api:
  dashboard: true
  debug: true
entryPoints:
  http:
    address: ":80"
    http:
      redirections:
        entryPoint:
          to: https
          scheme: https
  https:
    address: ":443"
serversTransport:
  insecureSkipVerify: true
providers:
  docker:
    endpoint: "unix:///var/run/docker.sock"
    exposedByDefault: false
  file:
    filename: /config.yml
certificatesResolvers:
  cloudflare:
    acme:
      email: email@example.net
      storage: acme.json
      dnsChallenge:
        provider: cloudflare
        disablePropagationCheck: true
        resolvers:
          - "1.1.1.1:53"
          - "1.0.0.1:53"
```

To spin up the traefik docker container, run

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

Once docker container created, traefik will generate ssl certs for `internel.example.net` & wildcard cert for `*.internel.example.net`. traefik dashboard will be available at `traefik.internal.example.net`.

Vaultwarden

Volume

I’m using named volumes here for the sake. You can use any directory on the host and bind that.

```
```

```
docker volume create vaultwarden
```

Reason for creating volume outside the compose file, in case, container destroyed with `rm`, data would be still available from the volume.

Docker-compose

Create a new directory in your home `vaultwarden` and add new file `docker-compose.yml`.

```
```

```
version: '3'

services:
  vaultwarden:
    image: vaultwarden/server:latest
    container_name: vaultwarden
    restart: unless-stopped
    security_opt:
      - no-new-privileges:true
    networks:
      - proxy
    ports:
      - 8100:80
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - vaultwarden:/data
    environment:
      - DOMAIN=https://vaultwarden.internal.example.net
      - SMTP_HOST=smtp.example.com
      - SMTP_FROM=email@example.com
      - SMTP_FROM_NAME=Vaultwarden
      - SMTP_SECURITY=SECURITYMETHOD
      - SMTP_PORT=XXXX
      - SMTP_USERNAME=email@example.com
      - SMTP_PASSWORD=YourReallyStrongPasswordHere
      - SMTP_AUTH_MECHANISM="Mechanism"
    labels:
      - traefik.enable=true
      - traefik.docker.network=proxy
      - traefik.http.middlewares.redirect-https.redirectScheme.scheme=https
      - traefik.http.middlewares.redirect-https.redirectScheme.permanent=true
      - traefik.http.routers.vaultwarden-https.rule=Host(`vaultwarden.internal.example.net`)
      - traefik.http.routers.vaultwarden-https.entrypoints=https
      - traefik.http.routers.vaultwarden-https.tls=true
      - traefik.http.routers.vaultwarden-https.service=vaultwarden
      - traefik.http.routers.vaultwarden-http.rule=Host(`vaultwarden.internal.example.net`)
      - traefik.http.routers.vaultwarden-http.entrypoints=http
      - traefik.http.routers.vaultwarden-http.middlewares=redirect-https
      - traefik.http.routers.vaultwarden-http.service=vaultwarden
      - traefik.http.services.vaultwarden.loadbalancer.server.port=80
      - traefik.http.routers.vaultwarden-websocket-https.rule=Host(`vaultwarden.internal.example.net`) && Path(`
      - traefik.http.routers.vaultwarden-websocket-https.entrypoints=https
      - traefik.http.routers.vaultwarden-websocket-https.tls=true
      - traefik.http.routers.vaultwarden-websocket-https.service=vaultwarden-websocket
      - traefik.http.routers.vaultwarden-websocket-http.rule=Host(`vaultwarden.internal.example.net`) && Path(`/
      - traefik.http.routers.vaultwarden-websocket-http.entrypoints=http
      - traefik.http.routers.vaultwarden-websocket-http.middlewares=redirect-https
      - traefik.http.routers.vaultwarden-websocket-http.service=vaultwarden-websocket
      - traefik.http.services.vaultwarden-websocket.loadbalancer.server.port=3012

networks:
  proxy:
    external: true

volumes:
  vaultwarden:
    external: true
```

> dns records should already pointed to you docker host. e.g. if docker host ip is `10.20.20.5` A record for `vaultwarden.internal` should point to `10.20.20.5`.

To spin up the vaultwarden docker container, run

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

Conclusion

For more details and documentation, visit Official github repo. Any queries, feel free to drop a comment. `Au Revoir`.

Revision #3
Created 3 August 2024 02:25:46 by ColtM
Updated 7 August 2024 23:24:39 by ColtM