Anwendungs-Konfigurationen

Cockpit (lokale Anwendung) hinter Traefik erreichbar machen

Um die Anwendung "Cockpit" über Traefik (läuft als Docker-Container) erreichbar zu machen sind folgende Schritte notwendig:

Zunächst muss eine Sub-Domain eingerichtet werden und diese über einen File-Provider konfiguriert werden. Diese Datei wird über Docker-Yaml-Datei in die Traefik-Konfiguration mit eingebunden:

```
services:
  traefik:
    image: traefik:latest
    container_name: traefik
    restart: always
    command:
      # Experimentell, um das Dashboard ohne Zugriffsschutz aufzurufen
      - "--api.insecure=true"
      - "--api.dashboard=true"
      - "--log.level=warn"
      - "--providers.docker"
      # File-Provider einbinden:
      - "--providers.file.directory=/etc/traefik/"
      - "--providers.file.watch=true"
      ##########################
      - "--providers.docker.exposedByDefault=false"
      - "--providers.docker.network=traefik_web"
      - "--entrypoints.http.address=:80"
      - "--entrypoints.https.address=:443"
      - "--entrypoints.http.http.redirections.entrypoint.to=https"
      - "--entrypoints.http.http.redirections.entrypoint.scheme=https"
      - "--entrypoints.https.address=:443"
```

```
      # Vermeidet, dass wir den resolver in jedem container mit
"traefik.http.routers.https.tls.certresolver=le" angeben muessen
      - "--entrypoints.https.http.tls.certResolver=le"
      - "--certificatesresolvers.le.acme.tlschallenge=true"
      # Staging-Server von Let's encrypt nutzen (für Testzwecke, größere Limits)
      #- "--certificatesresolvers.le.acme.caserver=https://acme-staging-
v02.api.letsencrypt.org/directory"
      - "--certificatesresolvers.le.acme.email=thomas@atj-krueger.de"
      - "--certificatesresolvers.le.acme.storage=/letsencrypt/acme.json"
      - "--certificatesresolvers.le.acme.tlschallenge=true"

    ports:
      - "80:80"
      - "443:443"
      - "8080:8080"

    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - /etc/docker/container/traefik:/letsencrypt
      # Konfigurationsdatei für File-Provider einbinden:
      # Hinweis: In diesem Fall muss die Datei auf dem Host links vom Doppelpunkt stehen.
      # Die Zieldatei im Traefik-Container steht entsprechende rechts von Doppelpunkt.
      -
/etc/docker/container/traefik/dynamic_conf/dynamic_conf.yml:/etc/traefik/dynamic_conf.yml

    networks:
      - web
    extra_hosts:
      - "host.docker.internal:host-gateway"



networks:
  web:
  external: true
    name: traefik_web
```

Inhalt der Datei dynamic_conf.yml (Korrekte Sub-Domain und lokale IP-dresse des Servers müssen noch angepasst werden):

```
# As YAML Configuration File
http:
  serversTransports:
    mytransport:
      insecureSkipVerify: true

  routers:
    cockpit:
      service: cockpit
      rule: "Host(`cockpit.domain.de`)"

  services:
    cockpit:
      loadBalancer:
        servers:
          - url: "http://ip-des-Servers:9090"
```

Darüber hinaus muss Cockpit selber auch noch etwas umkonfiguriert werden. Dazu legt man eine Datei mit Namen "cockpit.conf" im Pfad /etc/cockpit an. Deren Inhalt sieht bei mir so aus:

```
[WebService]
Origins = https://cockpit.domain.de wss://cockpit.domain.de
ProtocolHeader = X-Forwarded-Proto
AllowUnencrypted=true
```

Anschließend müssen der Traefik-Container neu erstellt und Cockpit neu gestartet werden:

```
sudo docker stop traefi
sudo docker rm traefik
sudo docker-compose -f traefik.yml up --force-recreate --build -d

sudo systemctl stop cockpit
```

Cockpit startet sich automatisch, wenn ein Aufruf erfolgt.

Mit dieser Konfiguration ist Cockpit nur noch über Traefik lauffähig. Ruft man Cockpit direkt über IP-Adresse und Port auf, kann man sich zwar anmelden, aber im weiteren Verlauf erscheint eine Fehlermeldung.

Graylog aufsetzen

Graylog (open) ist ein Logserver auf dem man Logs speichern und auswerten kann.

Damit Graylog funktionieren kann werden noch MongDB (Community Edition) und Opensearch benötigt.

Yaml-Datei für den gesamten Docker-Stack (ohne Passwörter):

```
services:
  mongodb:
    image: mongodb/mongodb-community-server:7.0.14-ubi9
    container_name: log-mongodb
    command: mongod --auth --port 50001 #Authentifizierung erzwingen und Port auf 50001 ändern
    volumes:
      - mongodb_data:/data/db
    restart: unless-stopped
    environment:
      TZ: "Europe/Berlin"


    #Port öffnen, wenn von Ferne zugegriffen werden soll
    #ports:
    #  - "27017:27017"

    networks:
      traefik_web:


  opensearch:
    image: opensearchproject/opensearch:latest
    container_name: log-opensearch
    environment:
      - TZ=Europe/Berlin
      - OPENSEARCH_JAVA_OPTS=-Xms500m -Xmx500m #500 MB Speicher wird zur Verfügung gestellt
      - bootstrap.memory_lock=true
      - discovery.type=single-node
```

```
      - action.auto_create_index=false
      - plugins.security.ssl.http.enabled=false
      - plugins.security.disabled=true
      - OPENSEARCH_INITIAL_ADMIN_PASSWORD=<Initial_Admin_Passwort>
    ulimits:
      memlock:
        hard: -1
        soft: -1
    volumes:
      - os_data:/usr/share/opensearch/data
    restart: unless-stopped

    networks:
      - traefik_web

  graylog:
    hostname: server
    image: graylog/graylog:6.0
    container_name: log-graylog
    depends_on:
      opensearch:
        condition: service_started
      mongodb:
        condition: service_started
    entrypoint: /usr/bin/tini -- wait-for-it opensearch:9200 --  /docker-entrypoint.sh
    environment:
      TZ: "Europe/Berlin"
      GRAYLOG_NODE_ID_FILE: "/usr/share/graylog/data/config/node-id"
      GRAYLOG_PASSWORD_SECRET: "${GRAYLOG_PASSWORD_SECRET:?Please configure
GRAYLOG_PASSWORD_SECRET in the .env file}"
      GRAYLOG_ROOT_PASSWORD_SHA2: "${GRAYLOG_ROOT_PASSWORD_SHA2:?Please configure
GRAYLOG_ROOT_PASSWORD_SHA2 in the .env file}"
      GRAYLOG_HTTP_BIND_ADDRESS: "0.0.0.0:9000"
      GRAYLOG_HTTP_EXTERNAL_URI: "http://localhost:9000/"
      GRAYLOG_ELASTICSEARCH_HOSTS: "http://opensearch:9200"
      GRAYLOG_MONGODB_URI: "mongodb://<graylog-user>:<graylog-user-
passwort>@mongodb:50001/graylog"
      GRAYLOG_ROTATION_STRATEGY: "time"
      GRAYLOG_ELASTICSEARCH_MAX_TIME_PER_INDEX: "1d"
      GRAYLOG_ELASTICSEARCH_MAX_NUMBER_OF_INDICES: "7"
```

```
      GRAYLOG_RETENTION_STRATEGY: "delete"
      GRAYLOG_ROOT_TIMEZONE: "Europe/Berlin"
      GRAYLOG_ELASTICSEARCH_ANALYZER: "standard"

    ports:
      - "9000:9000/tcp"   # Server API

    volumes:
      - "graylog_data:/usr/share/graylog/data/data"
      - "graylog_journal:/usr/share/graylog/data/journal"
    restart: unless-stopped

    networks:
      - traefik_web

volumes:
  mongodb_data:
  os_data:
  graylog_data:
  graylog_journal:



networks:
  traefik_web:
    external: true
```

Nach dem ersten Start kann Graylog so noch nicht funktionieren, da die MongoDB noch manuell konfiguriert werden muss.

MongoDB konfigurieren

Damit Graylog mit der MomDB zusammenarbeiten kann, müssen wir einen User (mit Adminrechten) und eine Datenbank "graylog" anlegen.

User anlegen

Über Portainer kann man eine eine Konsole auf dem Container starten. Hier startet man die Anwendung "mongosh". Wenn die MongoDB nicht über den Standardport erreichbar ist, muss der Aufruf wie folgt erforlgen:

```
mongosh mongodb://localhost:50001
```

Danach legen wir wie folgt den Benutzer "graylog-user" an:

```
db.createUser(
{
user: "graylog-user",
pwd: "passwort",
roles: [ "readWrite", "dbAdmin" ]
}
)
```

Für "passwort" natürlich ein geeignetes Passwort vergeben. Der Benutzer "graylog-user" kann lesen und schreiben und ist Datenbankadmin.

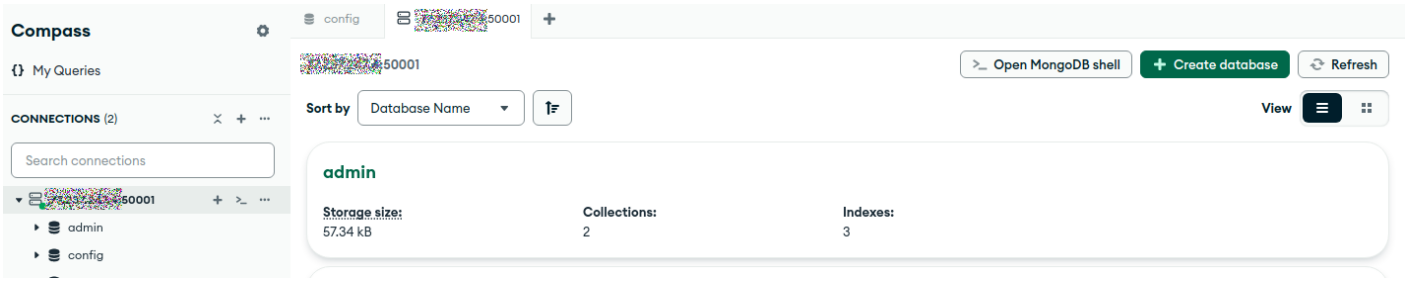
Datenbank "graylog" anlegen

Nach meiner Erfahrung nach ist es am einfachsten hierzu das Tool "Compass" zu nutzen.

Das Toll wird auf einem PC installiert.

Nach dem Start muss man zunächst eine Verbindung zu der MongoDB aufbauen. Hierzu muss die MongoDB von außen erreichbar sein (der Port muss nach außen weiter gereicht werden und eine möglich Firewall muss ebenfalls den Port durchlassen).

Für die Anmeldung nutzt man den zuvor angelegten Benutzer "graylog-user".



Hier wählt man "Create database". In dem dann erscheinenden Dialog trägt man in beiden Feldern "graylog" ein und bestätigt mit "Create database".

Damit ist die MongoDB für den Einsatz mit Graylog vorbereitet.