Docker ohne Traefik (ubusrv)

Einleitung

Auf dem lokalen Server ubusrv läuft ein Ubuntu 22.04 LTS mit Docker.

Als Docker Services sind installiert:

- **nginx** - ein Website für Tests http://ubusrv:80/
- **Maria DB** - eine Datenbank, die hauptsächlich für Bookstack genutzt wird
- **Mongo DB** - eine Non SQL Datenbank für Tests
- **Oracle XE DB** - eine Oracle 21g Datenbank für Tests
- **Bookstack** - ein Dokumentationswerkzeug http://ubusrv:9000/
- **ConfigServer** - ein beispielhafter Spring Boot Config Server
- **Adminer** - ein minimaler phpMyadmin clone http://ubusrv:8080/
- **Mongo Express** - eine Bedienoberfläche für Mongo http://ubusrv:8081/

Die Konfiguration erfolgt in der Docker Compose Standarddatei docker-compose.yml im Verzeichnis ~/workspace/docker.

Services: docker_compose.yml

ubusrv stellt diverse Services wie Bokkstack, Oracle-XE Datenbank, Maria-DB etc. bereit. Alle Services werden über Docker Images realisiert. Die Images und deren Parameterisierung werden in einer einzigen docker-compose.yml Datei verwaltet.

```
#
# Start with sudo docker-compose up -d
# Stop  with sudo docker-compose down
#
version: '3.3'

#
# declare Services
#
services:
  #
  # nginx
  #
  nginx:
    image: nginx
    ports:
      - 80:80
      - 443:443
    volumes:
      - /home/ralf/workspace/docker/data/nginx:/usr/share/nginx/html
    restart: always

  #
  # Maria DB Database
  #
  mariadb:
    image: mariadb
    volumes:
```

```
    - /home/ralf/docker/data/mariadb:/var/lib/mysql
  restart: always
  ports:
    - 3306:3306
  environment:
    MYSQL_ROOT_PASSWORD: bistef03

#
# Mongo DB
#
mongo:
  image: mongo
  volumes:
    - /home/ralf/docker/data/mongo:/data/db
  restart: always
  ports:
    - 27017:27017
  environment:
      MONGO_INITDB_ROOT_USERNAME: ralf
      MONGO_INITDB_ROOT_PASSWORD: bistef03

#
# Oracle XE DB
#
oracle:
  image: gvenzl/oracle-xe:full
  volumes:
    - /home/ralf/docker/data/oracle:/data/oradb
  restart: always
  ports:
    - 1521:1521
  environment:
    ORACLE_PASSWORD: bistef03
    APP_USER: anlei
    APP_USER_PASSWORD: anlei

#
# Bookstack
#
bookstack:
```

```
    image: lscr.io/linuxserver/bookstack
    container_name: bookstack
    environment:
      - PUID=1000
      - PGID=1000
      - APP_URL=http://ubusrv:9000
      - DB_HOST=db
      - DB_USER=root
      - DB_PASS=bistef03
      - DB_DATABASE=bookstack
    volumes:
      - /home/ralf/docker/data/bookstack:/config
    ports:
      - 9000:80
    restart: unless-stopped
    depends_on:
      - mariadb

  #
  # Spring Boot Config Server
  #
  config:
    depends_on:
      - mariadb
    image: hyness/spring-cloud-config-server
    restart: always
    ports:
      - 8888:8888
    environment:
      SPRING_PROFILES_ACTIVE: jdbc
      SPRING_DATASOURCE_URL: jdbc:mariadb://db:3306/cloud_config
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: bistef03
      SPRING_CLOUD_CONFIG_SERVER_JDBC_SQL: 'SELECT `key`, value FROM PROPERTIES WHERE
application=? and profile=? and label=?'

  #
  # Adminer: Small implementation of phpMyAdmin
  #
  adminer:
```

```
  depends_on:
    - mariadb
  image: adminer
  restart: always
  ports:
    - 8080:8080

#
# Mongo-Express
#
mongo-express:
  depends_on:
    - mongo
  image: mongo-express
  restart: always
  ports:
    - 8081:8081
  environment:
    ME_CONFIG_MONGODB_ADMINUSERNAME: ralf
    ME_CONFIG_MONGODB_ADMINPASSWORD: bistef03
    ME_CONFIG_BASICAUTH_USERNAME: ralf
    ME_CONFIG_BASICAUTH_PASSWORD: bistef03
```

Oracle XE DB Installation

> Die Oracle XE Datenbank läuft nur auf Intel-Prozessoren. Auf ARM basierten Servern wie z. B. dem Raspberry PI lässt sich die Datenbank nicht installieren.

Um mit der freien Oracle XE Datenbank arbeiten zu können, wird diese auf dem lokalen Server ubusrv als Docker Container installiert. Hierzu ist die Datei ~/workspace/docker/docker-compose.yml um folgenden Eintrag zu ergänzen:

```
    #
    # Oracle XE DB
    #
    oracle:
        image: gvenzl/oracle-xe:full
        volumes:
            - oracle_data:/data/oradb
        restart: always
        ports:
            - 1521:1521
        environment:
            ORACLE_PASSWORD: bistef03
            APP_USER: anlei
            APP_USER_PASSWORD: anlei


...
#
# Declare Volumes
#
volumes:
    db_data: {}
    mongo_data: {}
    oracle_data: {}
```

Nach einem `sudo docker-compose down` gefolgt von einem sudo `Docker-compose up &` wird das Image vom Docker-Hub heruntergeladen, installiert und gestartet.

Die Datenbank ist nun über das Port 1521 erreichbar. Neben dem Datenbank Basis-Container XE-DB wurde im Datenbank-Container XEPDB1 automatisch das Schema ANLEI angelegt. Auslöser sind

die Parameter APP_USER und APP_USER_PASSWORD in obiger .yml Datei. Somit können zwei DBA-Verbindungen und eine USER-Verbindung zur Datenbank aufgebaut werden. Die Verbindungsparameter für diese lauten für den SQL-Developer wie folgt:

```
SYS@XE-DB
=========
□Datenbanktyp: □□□Oracle
□Authentifizierungstyp: □Standard
□Benutzername:□□□sys
□Rolle:□□□□□SYSDBA
□Kennwort:□□□□bistef03
□Verbindungstyp:□□□Einfach
    Hostname:□□□□ubusrv
    Port:□□□□□1521
    SID:□□□□□XE
    Servica-Name:□□□(kein Eintrag)

SYS@XEPDB1
==========
□Datenbanktyp: □□□Oracle
□Authentifizierungstyp: □Standard
□Benutzername:□□□sys
□Rolle:□□□□□SYSDBA
□Kennwort:□□□□bistef03
□Verbindungstyp:□□□Einfach
    Hostname:□□□□ubusrv
    Port:□□□□□1521
    SID:□□□□□(kein Eintrag)
    Servica-Name:□□□XEPDB1

ANLEI@XEPDB1
==========
□Datenbanktyp: □□□Oracle
□Authentifizierungstyp: □Standard
□Benutzername:□□□anlei
□Rolle:□□□□□Standard
□Kennwort:□□□□anlei
□Verbindungstyp:□□□Einfach
    Hostname:□□□□ubusrv
    Port:□□□□□1521
```

```
    SID:□□□□□(kein Eintrag)
    Servica-Name:□□□XEPDB1
```

Um ein weitere Schemas im Arbeitscontainer XEPDB1 anzulegen, sind nachfolgende SQL-Befehle als SYS-User auszuführen:

```
//
// Es wird davon ausgegangen, dass SYS sich auf dem Basis-Container
// angemeldet hat. Sollte sich SYS auf dem dem Arbeits-Container
// XEPDB1 angemeldet haben, können die beiden ALTER SESSION Befehle
// zu Beginn und zum Ende des Skripts auskommentiert werden.
//
// Der Term <schemaname> ist durch den gewünschten Schema-Namen zu
// ersetzen.
//
ALTER SESSION SET CONTAINER = XEPDB1;

CREATE BIGFILE TABLESPACE
    <schemaname>_bfts
    DATAFILE '<schema_name>_bfts1.dbf'
    SIZE 20M
    AUTOEXTEND ON
;

CREATE USER
    <schemaname>
    IDENTIFIED BY <password>
    DEFAULT TABLESPACE <schemaname>_bfts
    QUOTA UNLIMITED ON <schema_name>_bfts
;

GRANT ALL PRIVILEGES TO <schema_name> WITH ADMIN OPTION;

ALTER SESSION SET CONTAINER = CDB$ROOT;
```

Oracle XE DB: DBMS_CRYPTO

Um Passwörter sicher in eigenen Tabellen zu speichern, sollte stets eine sichere HASH-Funktion verwendet werden. Oracle bietet eine solche im Package SYS.DBMS_CRYPTO an.

Das Package einem User nutzbar machen

Der Zugriff auf das Package für "normale" User kann erst nach einem GRANT EXECUTE erfolgen. Hierzu ist wie folgt vorzugehen:

- als User SYS auf dem Arbeitsmarkt-Container XEPDB1 anmelden
- den Befehl `grant EXECUTE on sys.dbms_crypto to <user>;` ausführen

Password-Hash

Kennwörter sollten immer als nicht rückrechenbare Hash-Werte gespeichert werden. Sie müssen nicht entschlüsselt werden, um sie auf Gleichheit zu überprüfen. Stattdessen überprüft man den Hash-Wert auf Gleichheit. Hier bietet das Package eine sinnvolle Unterstützung, da es Strings in einen nach derzeitigem Stand sicheren Hash-Wert umwandelt. Allerdings ist darauf zu achten, dass nur sichere Algorithmen verwendet werden, denn das Package bietet aus Gründen der Kompatibilität auch unsichere Algorithmen wie MD5 an. Um dieses Problem zu umgehen und als eine für eine Datenbankunabhängige Nutzung sinnvolle Erleichterung sollte eine Funktion installiert werden, die den Aufruf des Package abstrahiert. Für Oracle sieht diese Funktion wie folgt aus:

```
//
// Funktion zum Erzeugen eines sicheren HASH aus dem übergenen String
// Es wird der SHA-512 Algorythmus verwendet (gekennzeichnet durch den
// 2. Parameter im dbms-Aufruf). Er ist der derzeit sicherste angebotene
// Algorythmus.
//
CREATE OR REPLACE
FUNCTION CRYPTO_HASH (v_input VARCHAR2)
RETURN RAW DETERMINISTIC
AS
   PRAGMA UDF;
BEGIN
   RETURN dbms_crypto.hash(utl_raw.cast_to_raw(v_input), 6);
```

```
END CRYPTO_HASH ;
/
```

Durch den Aufruf `select crypto_hash('Das wäre mal ein Kennwörtlein') as password from dual;` erhält man als HASH-Wert etwa folgenden 128 Zeichen langen String: `343D52872991F63A64191971501B5515703A34357E2A54956D3AFF79C225AC45BF9DF4CC7EC497211CE75FCE51A2F0A1E621DEA3E656480C5E55ABFF61D8BA6F`.

Decrypt / Encrypt

Die Funktionen zum Verkauf- und Entschlüsseln eines Textes sollten nur mit äußerster Vorsicht verwendet werden. Wenn sich ein Angreifer Zugriff auf die Datenbank verschaffen konnte, kann er auch mit Hilfe der Standardfunktionen derart verschlüsselte Texte entschlüsseln. Sie eignen sich daher nicht für Kennwörter, die wieder entschlüsselt werden müssen (z. B. für Geschütze API-Zugriffe). Hier bietet Vault eine deutlich sicherere Lösung.

Oracle XE DB: Mac M1 Docker Image

Um die Oracle Datenbank Oracle-XE auf einem Apple-Rechner mit M1 Chip zu installieren, wird die Hilfsoftware COLIMA benötigt. Sie macht Docker-Images, die für die X86/AMD-Architektur erstellt wurden unter der ARM-Architektur des Apple Silicon lauffähig. Installiert wird sie mit Homebew:

```
brew install colima
```

Anschließend kann, wenn noch nicht geschehen, der Docker Client ebenfalls mit Hilfe von Homebrew installiert werden:

```
brew install docker
```

Sind beide Pakete erfolgreich installiert, muss zuerst Colima gestartet werden. Dazu wird über Kommandozeilenparameter die erforderliche Architektur eingestellt. Mit weiteren Parametern kann Einfluss auf die Zahl der CPUs, die Größe des RAMs und die Größe des Dateisystems Einfluss genommen werden. Genauere Informationen zu den Parametern liefert die Hilfe. Der einfache Start für die Oracle-XE sieht wie folgt aus:

```
#
# For help use: colina start --help
#
colina start --arch x86_64 --memory 4
```

Die VM für Docker Container wird nun gestartet und es dauert eine Weile, bis das Terminal wieder Befehle entgegennimmt. Ist der Eingabeprompt wieder vorhanden, kann die Datenbank gestartet werden:

```
docker run -d -p 1521:1521 -e ORACLE_PASSWORD=geheimesPassword gvenzl/oracle-xe
```

Zum Stoppen der Datenbank erst das Image wie gewohnt anhalten, dann Colima stoppen: `colima stop`.