How to Secure a Connection between Exalate and a PostgreSQL Database in Docker?

Last Modified on 05/09/2025 9:49 am EDT

Introduction

Assume you are bringing up an Exalate on docker, and you have the need to secure the connection between the application and the database using SSL.

With the standard configuration you will encounter the following error:

```
Caused by: org.postgresql.util.PSQLException: FATAL: SSL connection is required. Please specify SSL options and retry.
```

Or alternatively, you want to be sure that PostgreSQL is accessed over SSL.

> **Note**: For more information we suggest checking out the following links:
>
> Configuration of the database server
> https://www.postgresql.org/docs/9.1/ssl-tcp.html
>
> Configuration of Docker:
> https://github.com/readthedocs/readthedocs.org/pull/5556
>
> Setting the permissions:
> https://stackoverflow.com/questions/55072221/deploying-postgresql-docker-with-ssl-certificate-and-key-with-volumes

Setting up PostgreSQL using SSL

Configure the docker-compose to bring up PostgreSQL

Adapt the docker-compose.yml such that PostgreSQL comes up in an SSL mode:

```
version: '2'

services:
  database:
    restart: unless-stopped
    volumes:
      - ./persist/db:/var/lib/postgresql/data
      - ./createdb.sh:/docker-entrypoint-initdb.d/init-user-db.sh
           #
           # provide the certificate and the key to the postgres server
          #

      - ./ca/server.crt:/var/lib/postgresql/server.crt
      - ./ca/server.key:/var/lib/postgresql/server.key
    image: postgres:alpine

    #
    # ensure postgres is coming up with ssl mode on
    #
    command: -c ssl=on -c ssl_cert_file=/var/lib/postgresql/server.crt -c ssl_key_file=/var/lib/postgresql/server.key
    environment:
      POSTGRES_DB: mydb
      POSTGRES_USER: user
      POSTGRES_PASSWORD: secret
    environment:
      - POSTGRES_PASSWORD=password
      - DB_NAME=snownode
      - DB_USER=idalko
      - DB_PASS=idalko
    networks:
      - database

networks:
  database:
    driver: bridge
  default:
    driver: bridge
```

Create the certificates

You can create self-signed certificates as follows

```
# Store the certificates in a specific folder on your host
mkdir ca
cd ca

# use openssl to generate the certificates

openssl req -new -text -out server.req
openssl rsa -in privkey.pem -out server.key
rm privkey.pem
openssl req -x509 -in server.req -text -key server.key -out server.crt

# change ownership and permissions.  It depend on the underlying operating system.  Userid 70 is postgres on the po
stgres:alpine image

sudo chown 70:70 server.key
sudo chmod 600 server.key

cd ..
docker-compose up -d database
```

Validate

We like to validate if it works before moving on

```
# assuming that the database 'snownode' has been setup. if there is another database - use that
docker exec -it <name of the container running the database> /bin/bash
psql -U idalko -h localhost snownode
```

It must confirm that the SSL is enabled

```
bash-5.0# psql -U idalko -h localhost snownode
psql (12.3)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
```

An additional check is to do a PLSQL command

```
bash-5.0# psql -U idalko -h localhost snownode
psql (12.3)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

snownode=# show ssl
snownode-# ;
ssl
-----
on
```

Configure the app to access the database using SSL

The only configuration to be added to the docker-compose is by specifying that the PGSSLMODE is required

The adapted docker-compose looks like

```
version: '2'

services:
  database:

# <snip>

  snownode:
    restart: unless-stopped
    ports:
      - 9000:9000
    image: idalko/snownode:5.0.19
    depends_on:
      - database #wait for postgres to be started, not for ready
    volumes:
      - ./persist/home:/opt/snownode/data
    environment:

#ensure that the connection to the database is using SSL
      - PGSSLMODE="require"
      - SNOWNODE_PORT=9000
      - SNOWNODE_PG_HOST=database
      - SNOWNODE_PG_DB=snownode
      - SNOWNODE_PG_USER=idalko
      - SNOWNODE_PG_PWD=idalko
    networks:
      - database
      - default


networks:
  database:
    driver: bridge
  default:
    driver: bridge
```