OpenCTI Docker deployment

This is a fork from OpenCTI

OpenCTI could be deployed using the *docker-compose* command.

> For production deployment, we advise you to deploy `Grakn` and `ElasticSearch` manually in a dedicated environment and then to start the other components using `Docker`.

Clone the repository

```
$ mkdir /path/to/your/app && cd /path/to/your/app
$ git clone https://github.com/OpenCTI-Platform/docker.git
$ cd docker
```

Configure the environment

Before running the `docker-compose` command, don't forget to change the admin token (this token must be a valid UUID) and the password in the file `.env`. There is a file `.env.example` with a preset of variables for a demonstration purpose only.

If you cannot or don't want to use the `.env`, please edit the file `docker-compose.yml` with:

```
- APP__ADMIN__PASSWORD=ChangeMe
- APP__ADMIN__TOKEN=ChangeMe
```

And change the variable `OPENCTI_TOKEN` (for the `worker` and all connectors) according to the value of `APP__ADMIN__TOKEN`

```
- OPENCTI_TOKEN=ChangeMe
```

As OpenCTI has a dependency to ElasticSearch and Grakn, you have to set the `vm.max_map_count` before running the containers, as mentioned in the ElasticSearch documentation.

```
$ sysctl -w vm.max_map_count=1048575
```

To make this parameter persistent, please update your file `/etc/sysctl.conf` and add the line:

```
$ vm.max_map_count=1048575
```

Run

In order to have the best experience with Docker, we recommend to use the Docker stack feature. In this mode we will have the capacity to easily scale your deployment.

```
$ env $(cat .env | grep ^[A-Z] | xargs) docker stack deploy --compose-file
docker-compose.yml opencti
```

> In some configuration, Grakn could fail to start with the following error: `Starting Storage.....FAILED!` You can restart it by using the command `$ docker service update --force opencti_grakn`.

You can also deploy with the standard Docker command:

```
$ docker-compose --compatibility up
```

You can now go to http://localhost:8080 and log in with the credentials configured in your environment variables.

Update the stack or delete the stack

```
$ docker service update --force service_name
$ docker stack rm opencti
```

Behind a reverse proxy

If you want to use OpenCTI behind a reverse proxy with a context path, like `https://myproxy.com/opencti`, please change the base_path configuration.

```
- APP__BASE_PATH=/opencti
```

By default OpenCTI use Websockets so dont forget to configure your proxy for this usage.

Data persistence

If you wish your OpenCTI data to be persistent in production, you should be aware of the `volumes` section for `Grakn`, `ElasticSearch` and `MinIO` services in the `docker-compose.yml`.

Here is an example of volumes configuration:

```
volumes:
  grakndata:
    driver: local
```

```
    driver_opts:
      o: bind
      type: none
  esdata:
    driver: local
    driver_opts:
      o: bind
      type: none
  s3data:
    driver: local
    driver_opts:
      o: bind
      type: none
```

Memory configuration

OpenCTI default `docker-compose.yml` file does not provide any specific memory configuration. But if you want to adapt some dependencies configuration, you can find some links below.

OpenCTI - Platform

OpenCTI platform is based on a NodeJS runtime, with a memory limit of **512MB by default**. We do not provide any option to change this limit today. If you encounter any `OutOfMemory` exception, please open a Github issue.

OpenCTI - Workers and connectors

OpenCTI workers and connectors are Python processes. If you want to limit the memory of the process we recommend to directly use Docker to do that. You can find more information in the official Docker documentation.

> If you do not use Docker stack, think about `--compatibility` option.

Grakn

Grakn is a JAVA process that rely on Cassandra (also a JAVA process). In order to setup the JAVA memory allocation, you can use the environment variable `SERVER_JAVAOPTS` and `STORAGE_JAVAOPTS`.

> The current recommendation is `-Xms4G` for both options.

You can find more information in the official Grakn documentation.

ElasticSearch

ElasticSearch is also a JAVA process. In order to setup the JAVA memory allocation, you can use the environment variable `ES_JAVA_OPTS`.

> The minimal recommended option today is `-Xms512M -Xmx512M`.

You can find more information in the official ElasticSearch documentation.

Redis

Redis has a very small footprint and only provides an option to limit the maximum amount of memory that can be used by the process. You can use the option `--maxmemory` to limit the usage.

You can find more information in the Redis docker hub.

MinIO

MinIO is a small process and does not require a high amount of memory. More information are available for Linux here on the Kernel tuning guide.

RabbitMQ

The RabbitMQ memory configuration can be find in the RabbitMQ official documentation. Basically RabbitMQ will consumed memory until a specific threshold. So it should be configure along with the Docker memory limitation.