OnlyOffice Docker

Mettre en place OnlyOffice Docker avec Nextcloud

Prérequis :

- Avoir un nom de domaine
  - Dans notre cas nous utiliserons le nom de domaine : `miraceti.net`
- Avoir une configuration de **Nextcloud Docker** fonctionnel
  - Sinon suivre le tuto suivant : Déployer Nextcloud Docker

Configuration nom de domaine

Grâce à notre nom de domaine, nous allons pouvoir créer des sous domaine (en CNAME `A`).

Sur le site de gestion de notre nom de domaine respectif, dans la section DNS, ajouter l'entrée suivante :

- sub.domaine.ext A ip.adress.server
  - Soit
- office.miraceti.net A 12.34.56.78

Préparation des fichiers de configuration

> Il est conseillé de rajouter les configurations suivantes (`docker-compose.yml` et `.env`) directement dans les fichiers

> correspondants à la configuration Nextcloud.
> Mais il est aussi possible de séparer la configuration Nextcloud et la configuration Onlyoffice. Cependant il faudra mettre le conteneur Nextcloud dans un même `network` que le conteneur Onlyoffice.

Se placer dans le dossier contenant les fichiers `docker-compose.yml` et `.env` de la configuration Nextcloud Docker.

Edition du fichier `docker-compose.yml` :

```
...(services nextcloud)...

 postgres-onlyoffice:
   image: ${ONLYOFFICE_DOCUMENT_POSTGRES_IMAGE_TAG}
   container_name: postgres-onlyoffice
   volumes:
     - onlyoffice-document-postgres:/var/lib/postgresql/data
   environment:
     POSTGRES_DB: ${ONLYOFFICE_DOCUMENT_DB_NAME}
     POSTGRES_USER: ${ONLYOFFICE_DOCUMENT_DB_USER}
     POSTGRES_HOST_AUTH_METHOD: trust
   networks:
     - office-network
   healthcheck:
     test: ["CMD-SHELL", "pg_isready -U $$POSTGRES_USER -d $$POSTGRES_DB"]
     interval: 10s
     timeout: 5s
     retries: 3
     start_period: 60s
   restart: unless-stopped

 redis-onlyoffice:
   image: ${ONLYOFFICE_DOCUMENT_REDIS_IMAGE_TAG}
   container_name: redis-onlyoffice
   volumes:
     - redis-onlyoffice-document-data:/data
   networks:
     - office-network
   healthcheck:
     test: ["CMD", "redis-cli", "ping"]
     interval: 10s
     timeout: 5s
     retries: 3
     start_period: 60s
   restart: unless-stopped

 rabbitmq-onlyoffice:
   image: ${ONLYOFFICE_DOCUMENT_RABBITMQ_IMAGE_TAG}
```

```
    container_name: rabbitmq-onlyoffice
    volumes:
      - rabbitmq-onlyoffice-document-data:/bitnami/rabbitmq/mnesia
    environment:
      RABBITMQ_USERNAME: ${ONLYOFFICE_DOCUMENT_RABBITMQ_USER}
      RABBITMQ_PASSWORD: ${ONLYOFFICE_DOCUMENT_RABBITMQ_PASSWORD}
      POSTGRES_PASSWORD: ${ONLYOFFICE_DOCUMENT_DB_PASSWORD}
      RABBITMQ_MANAGEMENT_ALLOW_WEB_ACCESS: true
    networks:
      - office-network
    healthcheck:
      test: rabbitmq-diagnostics -q ping
      interval: 10s
      timeout: 5s
      retries: 3
      start_period: 90s
    restart: unless-stopped

  onlyoffice-document:
    image: ${ONLYOFFICE_DOCUMENT_IMAGE_TAG}
    container_name: onlyoffice
    volumes:
      - onlyoffice-document-data:/var/www/onlyoffice/Data
      - onlyoffice-document-log:/var/log/onlyoffice
      - onlyoffice-document-cache-
files:/var/lib/onlyoffice/documentserver/App_Data/cache/files
      - onlyoffice-document-public-files:/var/www/onlyoffice/documentserver-
example/public/files
      - onlyoffice-document-fonts:/usr/share/fonts
    environment:
      DB_TYPE: postgres
      DB_HOST: postgres-onlyoffice
      DB_PORT: 5432
      DB_NAME: ${ONLYOFFICE_DOCUMENT_DB_NAME}
      DB_USER: ${ONLYOFFICE_DOCUMENT_DB_USER}
      DB_PWD: ${ONLYOFFICE_DOCUMENT_DB_PASSWORD}
      AMQP_URI:
amqp://${ONLYOFFICE_DOCUMENT_RABBITMQ_USER}:${ONLYOFFICE_DOCUMENT_RABBITMQ_PASSWORD}@rabbitmq-
onlyoffice
      REDIS_SERVER_HOST: redis-onlyoffice
```

```
      REDIS_SERVER_PORT: 6379
      JWT_ENABLED: true
      JWT_SECRET: ${ONLYOFFICE_DOCUMENT_JWT_SECRET}
      JWT_HEADER: Authorization
      JWT_IN_BODY: 'true'
    networks:
      - office-network
      - bddnetwork
    ports:
      - 8081:80
    restart: unless-stopped
    depends_on:
      postgres-onlyoffice:
        condition: service_healthy
      rabbitmq-onlyoffice:
        condition: service_healthy
#      nextcloud:
#        condition: service_healthy
```

Edition du fichier `.env` :

```
...(Nextcloud Variables)..

# OnlyOffice Variables
ONLYOFFICE_DOCUMENT_POSTGRES_IMAGE_TAG=postgres:15
ONLYOFFICE_DOCUMENT_REDIS_IMAGE_TAG=redis:7.2
ONLYOFFICE_DOCUMENT_RABBITMQ_IMAGE_TAG=bitnami/rabbitmq:3.13.4
ONLYOFFICE_DOCUMENT_IMAGE_TAG=onlyoffice/documentserver:8.1
ONLYOFFICE_DOCUMENT_HOSTNAME=office.miraceti.net
ONLYOFFICE_DOCUMENT_DB_NAME=onlyoffice
ONLYOFFICE_DOCUMENT_DB_USER=onlyofficedbuser
ONLYOFFICE_DOCUMENT_DB_PASSWORD=yVNyHP2keUeD5wD3vkrb
ONLYOFFICE_DOCUMENT_JWT_SECRET=7bukHBGUEzvCjHrYKuT8
ONLYOFFICE_DOCUMENT_RABBITMQ_USER=rabbitmqdb
ONLYOFFICE_DOCUMENT_RABBITMQ_PASSWORD=FX7zuRA7y2wVUNkYwtwH
```

Création du network pour OnlyOffice

Nous allons créer un network "`office-network`" pour lier les conteneurs/services suivants entre eux :

- postgres-onlyoffice
- redis-onlyoffice
- rabbitmq-onlyoffice
- onlyoffice

```
sudo docker network create office-network
```

Vérification que le nouveau network existe :

```
sudo docker network ls
```

Configuration Apache2

Création d'un nouveau host dans `/etc/apache2/sites-available` avec le nom suivant "`office.miraceti.net.conf`":

```
<VirtualHost *:80>
        ServerName office.miraceti.net

□ErrorLog ${APACHE_LOG_DIR}/office.miraceti.net.log
        CustomLog ${APACHE_LOG_DIR}/office.miraceti.net.log combined

□ProxyPreserveHost On
□ProxyRequests Off
        ProxyPass / http://localhost:8083/□□□# Port Used by container OnlyOffice
        ProxyPassReverse / http://localhost:8083/□# Port Used by container OnlyOffice

        <IfModule mod_dav.c>
           Dav off
        </IfModule>

□  <Proxy *>
          Order deny,allow
          Allow from all
        </Proxy>
</VirtualHost>
```

Tester la configuration :

```
sudo apachectl configtest
```

Activation du nouveau host :

```
a2ensite office.miraceti.net.conf
```

Vérifier que l'accès via L'URL `office.miraceti.net` est bien accessible en HTTP.

Une fois accessible, il faut rendre sécurisé (HTTPS) l'accès à cette URL :

```
sudo certbot --apache -d office.miraceti.net
```

Ce qui générera le fichier suivant `office.miraceti.net-le-ssl.conf`. Ajouter la configuration suivante à la fin du fichier avant la balise de fin "`<VirtualHost>`" :

```
SetEnvIf Host "^(.*)$" THE_HOST=$1
RequestHeader setifempty X-Forwarded-Proto https
RequestHeader setifempty X-Forwarded-Host %{THE_HOST}e
ProxyAddHeaders Off
ProxyPass /.well-known !
ProxyPassMatch (.*)(\/websocket)$ "ws://localhost:8081/$1$2"
ProxyPass / http://localhost:8081/
ProxyPassReverse / http://localhost:8081/
```

Ce qui donne le fichier final :

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
□□ServerName office.miraceti.net

□□ErrorLog ${APACHE_LOG_DIR}/office.miraceti.net.log
        CustomLog ${APACHE_LOG_DIR}/office.miraceti.net.log combined

        SetEnvIf Host "^(.*)$" THE_HOST=$1
        RequestHeader setifempty X-Forwarded-Proto https
        RequestHeader setifempty X-Forwarded-Host %{THE_HOST}e
        ProxyAddHeaders Off
        ProxyPass /.well-known !
        ProxyPassMatch (.*)(\/websocket)$ "ws://localhost:8081/$1$2"
        ProxyPreserveHost On
        ProxyRequests Off
        ProxyPass / http://localhost:8081/
        ProxyPassReverse / http://localhost:8081/
```

```
        <IfModule mod_dav.c>
           Dav off
        </IfModule>

□□<Proxy *>
          Order deny,allow
          Allow from all
        </Proxy>

        SSLCertificateFile /etc/letsencrypt/live/office.miraceti.net/fullchain.pem
        SSLCertificateKeyFile /etc/letsencrypt/live/office.miraceti.net/privkey.pem
        Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>
```