Deploying a distributed application with *OpenStack* and *AWS Amazon*

September 2019

This exercise is to learn usage of OpenStack and AWS Amazon through:

- the *horizon* graphical interface of OpenStack and the AWS portal,
- the *OpenStack SDK* and boto (AWS Amazon SDK)

The following resources and tools are required for this exercise:

- An Ubuntu 16.04 machine or higher, Mac OS or Windows machine (with putty), with python 3 (or higher)
- An account on SwitchEngine (OpenStack)
- An account on AWS Amazon

Students must work in pairs. One will do the work on SwitchEngine, the other will do the work on AWS.

The goal of this project is to deploy the distributed Festival Search Engine (FSE) application[1] on two different IaaS: SwitchEngine and AWS Amazon. The deployment will be done in two ways:

1. Manual deployment : by using the GUI of the two Cloud infrastructures
2. Automatic deployment: by using the proprietary API of the two Cloud infrastructures.

FSE enables users to:

1. locate concerts/festivals locations on an interactive map,
2. Select and display useful information about a given concert/festival (location, online ticket shop, artists, etc.).
3. listen to song's excerpts of artists who will attend the concert/festival,
4. display useful information about the artists (type of music, the most popular songs, hometown, etc.) who will attend the selected concert/festival

The applications is composed of three modules:

A front end module : allows the user to visualise (on a google map) the location of the concerts/festivals. When the user clicks on an event on the map, FSE:

- displays a full page containing detailed information of the event and the artists,

[1] **This application was developed by Maxime Lovino and Thomas Ibanez.**

- plays a song from an artist present at the event,
- Provides information about the song currently played, i.e., the artist, the song title, the album cover....

The front end module uses Google Map API. For this purpose, you need a Google Map Key.

How to create your Google Map Key?

1. Go to the Google Cloud Platform: https://console.cloud.google.com/google/maps-apis/new
2. Create or select a project.
3. Click the left menu button and select **APIs & Services > Credentials**.
4. On the **Credentials** page, click **Create credentials > API key**.
   The **API key created** dialog displays your newly created API key.
5. Click on Google Maps in the left menu
6. Activate Maps JavaScript API and Places API

A back end module: A REST Server which wraps two public APIs used to collect the data related to concerts and festivals: Spotify and Eventful APIs.

Spotify supports functions such as playing song previews and providing artists pictures, while *Eventful* provides information about events such as the line-up, the location, and the date.

In addition, the backend module acts as a cache memory. It stores the previous requests in a database to limit the number of requests sent to Spotify and Eventful.

An authentication procedure is necessary to use *Spotify* and *Eventful*:

Spotify requests use two parameters: *Spotify Client ID* and *Spotify Client secret*. This URL https://developer.spotify.com/dashboard/ explains how to create them.

EventFul requests use one parameter: *EventFul Application key*. This URL: https://api.eventful.com/keys explains how to create this key.

A database module: This module supports a mongoDB database used to store all requests generated by the user as well as their results. The idea is to consult the requests/results history before sending the current request to Spotify and EventFul. The goal is to minimise the number of requests sent to the two APIs.

Each of the three modules will be deployed on a separate instance.

Exercise 1: Using *horizon* and the *AWS Amazon GUI* to deploy FSE (manual deployment)

Please use these images:

- AWS: Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-0cfee17793b08a293
- SwitchEngine: Ubuntu Xenial 16.04 (SWITCHengines)

Creation of a MongoDB instance

Access your instance by ssh and install MongoDB:

sudo apt-get update

sudo apt-get install mongodb

Configuration: Change the /etc/mongodb.conf configuration file in order to allow the database to listen on the network: change « *bind_ip = 127.0.0.1* » by « *bind_ip = 0.0.0.0* »

Reboot the machine: *sudo reboot*

To check whether the database listens on the network: *ge*

You will get this output (Figure 1):

```
Last login: Tue Sep 18 12:57:58 2018 from 195.176.12.2
ubuntu@mongodb:~$ ls -l
total 0
ubuntu@mongodb:~$ sudo netstat --listen -a -p | grep mongo
tcp        0      0 *:28017                 *:*                     LISTEN      1043/mongod
tcp        0      0 *:27017                 *:*                     LISTEN      1043/mongod
tcp        0      0 10.0.1.167:27017        fl-4-142.unil.clo:35984 ESTABLISHED 1043/mongod
tcp        0      0 10.0.1.167:27017        fl-4-142.unil.clo:35978 ESTABLISHED 1043/mongod
tcp        0      0 10.0.1.167:27017        fl-4-142.unil.clo:35814 ESTABLISHED 1043/mongod
tcp        0      0 10.0.1.167:27017        fl-4-142.unil.clo:35980 ESTABLISHED 1043/mongod
tcp        0      0 10.0.1.167:27017        fl-4-142.unil.clo:35982 ESTABLISHED 1043/mongod
unix  2      [ ACC ]     STREAM     LISTENING     10213    1043/mongod         /tmp/mongodb-27017.sock
ubuntu@mongodb:~$
```

Figure 1 : Output of the command *sudo netstat --listen -a -p | grep mongo*

Create a snapshot of your instance so as you can use it later without any reinstallation and/or configuration.

Creation of a Backend instance

1. Access your instance by ssh
2. Upload the zipped file *FSE.zip* and unzip it (if unzip if not installed: sudo apt-get install unzip)
3. cd FSEArchive
4. tar -xvf node-v8.11.4-linux-x64.tar.xz
5. export PATH=$PATH:"/home/ubuntu/FSEArchive/node-v8.11.4-linux-x64/bin"
6. You can check that npm is correctly installed by typing: npm --version
7. cd server

8. npm install
9. npm install -g forever
10. Create a file with the name keys.env. Fill it with this information:
    a. SPOTIFY_ID=*SPOTIFY_ID*
    b. SPOTIFY_SECRET=*SPOTIFY_SECRET*
    c. EVENTFUL=*EVENTFUL_KEY*
    d. DATABASE=mongodb://<MONGODB_IP:MONGODB_PORT>/festivaldb. By default the port used by Mongo is 27017.

To start the backend server, type: "forever start start.js". The server runs on port 3000.

Create a snapshot of your instance so as you can use it later without any reinstallation and/or configuration.

Creation of a Frontend instance

1. Access your instance by ssh
2. Upload the zipped file: FSE.zip and unzip it.
3. cd FSEArchive
4. tar -xvf node-v8.11.4-linux-x64.tar.xz
5. export PATH=$PATH:"/home/ubuntu/FSEArchive/node-v8.11.4-linux-x64/bin"
6. You can check that npm is correctly installed by typing: npm --version
7. cd client
8. npm install
9. npm install -g forever
10. create and edit keys.env, then write 'GMAP=<YOUR_GOOGLE_MAP_KEY>'
11. To start: forever start start.js
    --serverPublic=http://<SERVER_PUBLIC_IP>:<SERVER_PORT>

The client runs on port 8080. Create a snapshot of your instance so that you can use it later without any reinstallation and/or configuration.

To access the FSE application, type: http://client_IP:8080, where *client_IP* is the IP of the Frontend instance.The interface of the application is shown in Figure 2.

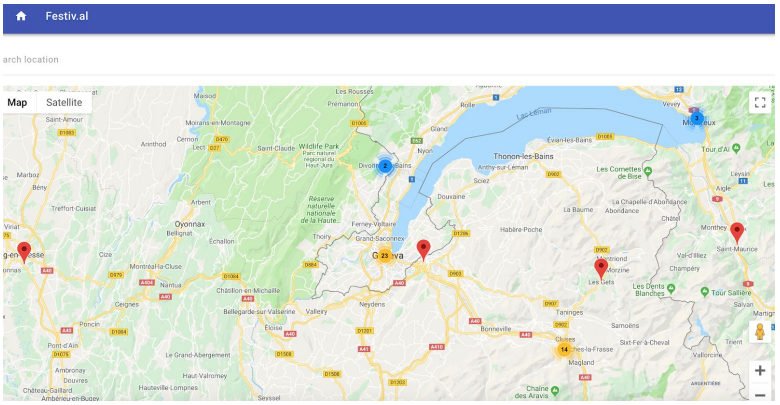

Figure 2: FSE interface

Exercise 2: Using *boto* to deploy *FSE* on AWS Amazon (automatic deployment)

The aim of this exercise is to use *boto* to automatically deploy FSE. *boto* is the Amazon Web Services (AWS) SDK for Python. It allows Python developers to write software that makes use of Amazon services. Boto provides an easy to use, object-oriented API as well as low-level direct service access.

For this exercise, you must use boto 3:
https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html

In order to use boto3 you need to install the *AWS CLI*:

https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html

Installing AWS CLI and boto3

1. Install AWS CLI*: pip3 install awscli --upgrade --user*

   Troubleshootings: If you get a *pip3* error during *awscli* installation on your local machine, you can use this alternative: Deploy an ubuntu 18.04 instance on AWS (ami-07d0cf3af28718ef8, in N. Virginia region) and run your code from this instance. To install awscli on this instance run the following commands:

   - *sudo apt update*
   - *sudo apt install python3-pip*
   - *pip3 install awscli --upgrade --user*
   - *source* ~/.profile

2. Execute this command: *aws configure.* Paste your access key, your secret key and choose in which region you want to work. boto3 will use this configuration. You can retrieve your access key and secret keys from your AWS Amazon account (See “My Security Credentials” menu).

3. Install boto3: *pip3 install boto3*

An incomplete code of the main python program that deploys FSE on AWS Amazon is available on CyberLearn.

Complete these routines:

- *create_connection*: create a connection to the AWS service
- *create_server*: launches one instance on AWS. create_server must wait until the instance can be accessed with ssh.
- *delete_server*: terminate the instance

The arguments of these routines are detailed in the code.

For the three instances to create, you can use your own images (Exercise 1) or these public AMIs:

- MONGO_IMG = 'ami-0ac04acec997e5fa5'
- BACKEND_IMG = 'ami-0b6176badcfb1f13d'
- FRONTEND_IMG = 'ami-0d531b0206af6f44b'

Exercise 3: Using *OpenStack SDK* to deploy *FSE* on SwitchEngine (automatic deployment)

The aim of this exercise is to use the OpenStack SDK to automatically deploy FSE. The OpenStack Python Software Development Kit (SDK) is used to write Python automation scripts that create and manage resources in your OpenStack cloud

To install OpenStack SDK: pip3 install openstacksdk

Troubleshootings: If you get a pip3 error during *openstacksdk* installation on your local machine, you can use this alternative: Deploy an ubuntu instance on Switch or AWS Amazon and run your code from this instance. To install *openstacksdk* on this instance run the following commands:

- *sudo apt update*
- *sudo apt install python3-pip*
- pip3 install openstacksdk

An incomplete code of the main python program that deploys FSE on AWS Amazon is available on CyberLearn.

Complete these routines:

- *create_connection(auth_url, project_name, username, password)*
- *delete_server(conn, srv)*
- *create_server(conn, name, img, flv, net, key, grp, userdata = "")*
- *get_unused_floating_ip(conn, public_network='public')*
- *attach_floating_ip_to_instance(conn, instance, floating_ip)*

The arguments of these routines are detailed in the code.