ACIT 3855 – Lab 6A – Asynchronous Messaging Setup

Instructor	Mike Mulder (mmulder10@bcit.ca) – Sets A and C Tim Guicherd (tguicherd@bcit.ca) – Set B
Total Marks	10
Due Dates	Screenshots due: • Feb. 19th for Set C • Feb. 22nd for Sets A and B

Purpose

- Prepare for Lab 6B where we will refactor our Receiver and Storage Services to use asynchronous messages through Kafka and add a new Audit Service that will use the messages as a data source.

Part 1 – Setup a Cloud VM

Using Azure or AWS create a VM instance with at least 2GB of memory (to allow Kafka to successfully run, I used a B1ms on Azure) as follows:

- Image Type: Ubuntu 20.04 LTS
- Make sure you have SSH access
- Make sure you setup a DNS Name for your VM. We will use this in configurations (rather than an IP which may change).

SSH to your VM.

You need to install Docker on your VM. Follow the steps 1 and 2 at:
https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04
Note that Steps 3 to 7 are an optional tutorial that you may wish to go through.

Also, install Docker Compose on your VM: https://docs.docker.com/compose/install/
Make sure you use the instructions for installing on Linux.

Both the **docker** and **docker compose** commands should now work on your VM (without needing sudo).

We will talk about Docker and Docker Compose in a future class when we use it to package and deploy our services.

Part 2 – Setup MySQL and Kafka

Using a GitHub or GitLab account (use github.com or gitlab.com, not your VM from 4850), create a repository. Will be using this repository in subsequent labs to deploy to your services.

In your Git repo, create sub-folder called deployment. Add a docker-compose.yml file to the deployment sub-folder with the content below.

- Make sure to replace `<VM DNS Name>` with the DNS name of your VM.
- You can also update the MYSQL_DATABASE, MYSQL_USER, MYSQL_PASSWORD and MYSQL_ROOT_PASSWORD values to match those from your local MySQL install (and so you don't use default authentication values).

```
version: '3.3'
services:
  zookeeper:
    image: wurstmeister/zookeeper
    ports:
      - "2181"
    hostname: zookeeper
  kafka:
    image: wurstmeister/kafka
    command: [start-kafka.sh]
    ports:
      - "9092:9092"
    hostname: kafka
    environment:
      KAFKA_CREATE_TOPICS: "events:1:1" # topic:partition:replicas
      KAFKA_ADVERTISED_HOST_NAME: <VM DNS Name>  # docker-machine ip
      KAFKA_LISTENERS: INSIDE://:29092,OUTSIDE://:9092
      KAFKA_INTER_BROKER_LISTENER_NAME: INSIDE
      KAFKA_ADVERTISED_LISTENERS: INSIDE://kafka:29092,OUTSIDE://<VM DNS Name>:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: INSIDE:PLAINTEXT,OUTSIDE:PLAINTEXT
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    depends_on:
      - "zookeeper"
  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: 'events'
      # So you don't have to use root, but you can if you like
      MYSQL_USER: 'user'
      # You can use whatever password you like
      MYSQL_PASSWORD: 'password'
      # Password for root access
      MYSQL_ROOT_PASSWORD: 'password'
    ports:
      # <Port exposed> : < MySQL Port running inside container>
      - '3306:3306'
    expose:
      # Opens port 3306 on the container
      - '3306'
      # Where our data will be persisted
    volumes:
      - my-db:/var/lib/mysql
# Names our volume
volumes:
  my-db:
```

Clone the repo with your docker-compose.yml file on your VM. Navigate to the directory on your VM with the docker-compose.yml file and run the following command to build and run Zookeeper, Kafka and MySQL in the background:

docker compose up -d

Notes:

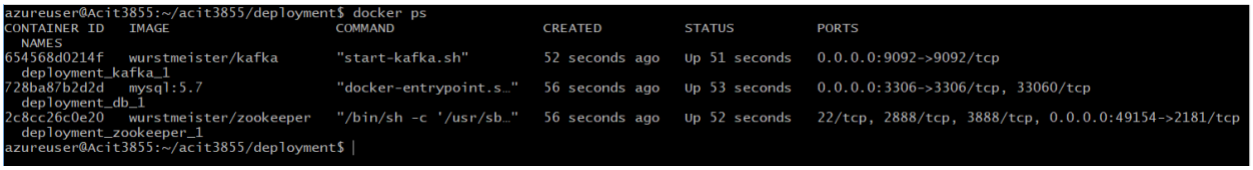
- *The above command may take a while to complete.*

- *The -d option runs Zookeeper, Kafka and MySQL in the background. If you run into problems, leave this option out to see more output (including error messages).*
- *You can use "docker compose down" to bring the services down again.*

Check that all three services are running with the following command:

docker ps

You should see something like this (you should not need sudo to run the command):



You will need to open the following ports on your VM network/security rules:

- Port 3306 for MySQL
- Port 9092 for Kafka

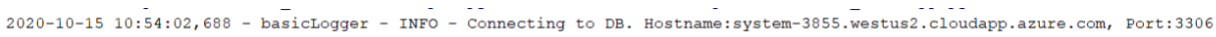
References:

- https://github.com/wurstmeister/kafka-docker
- https://hub.docker.com/_/mysql

Part 3 – Configure and Run the Storage Service with the MySQL on your VM

Locally on your laptop:

- You will need to update and run your DB scripts to create your events database (create_tables_mysql.py and drop_tables_mysql.py). Make sure to use the DNS Name of your VM for the host, the MYSQL username/password from the docker-compose.yml file for the username and password and the MYSQL database name from the docker-compose.yml file for the database.
- Update the configuration file of Storage Service (app_conf.yml). The datastore settings should correspond to those of the MySQL instance on your VM.
- Add an INFO log message to your Storage Service that displays the hostname and port of your MySQL database. It should look something like this (it should NOT be hard):



Run your services and use Postman or jMeter to make sure they still work. Troubleshoot as necessary.

Grading and Submission

Submit the following **four** screenshots to the Lab 6A Dropbox on D2L (they are worth 2.5 marks each):

- Your Git repository showing the contents of your docker-compose.yml file (you can hide the password information).
- The DNS Name of your VM from the Azure or AWS Portal.
- The output from docker ps on your VM showing MySQL, Kafka and Zookeeper running (it must include the command, and the command must not show sudo otherwise you will not receive marks for this one).
- Your log message from the console of the Storage Service showing the hostname (DNS Name) and port of the MySQL database (on your VM).

I should be able to see the DNS Name of your VM in the above screenshots.

Make sure to stop your VM when you are done. But don't delete it as we will be using it again.