Subhakanta Mishra
Subhamishra.in@gmail.com

Containerization with Docker

Course-End Project: **Swarm Microservice Deployment**

Objective

To deploy a scalable, multi-service voting application on a manager node, ensuring efficient orchestration, fault tolerance, and seamless monitoring through Docker visualizer.

Real-time scenario:

John, a DevOps engineer, is tasked with deploying a voting application through multiple microservices. By creating a Docker compose file and deploying it on a manager node in a distributed system, they ensure that each service is efficiently orchestrated and fault-tolerant. To monitor the deployment, John integrates Docker visualizer as a microservice, providing real-time insights. This setup simplifies the deployment process, enhances scalability, and ensures the application runs smoothly in a production environment.

Tasks

The following tasks outline the process of deploying swarm microservice:

1. Set up the network and storage infrastructure.
2. Define and configure microservices.
3. Deploy microservices across Docker swarm.

Subhakanta Mishra
Subhamishra.in@gmail.com

Solution

Step 1: Configure docker swarm

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker swarm init
Swarm initialized: current node (5f65lxziswjszlqj410kxqnbr) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-2y18ohgiwuovsw38k9jbmf4ufx27i0mknc4gpia6ijhw6zlptg-7l0q8pe4lonk2phhcgqmrcrrk 172.31.19.12:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step-2: Create necessary files as below.

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ ls
Dockerfile  app.py  compose.yaml  requirements.txt
```

Step-3: Define multi-container app structure. (compose.yml)

```yaml
version: "3"
services:
  web:
    build: .
    image: localhost:5000/stackdemo_web:1.0
    ports:
      - "8000:5000"
    deploy:
      replicas: 3
    links:
      - redis
    networks:
      - mynet
  redis:
    image: redis
    expose:
      - "6379"
    networks:
      - mynet
networks:
  mynet:
```

Step4: Install docker compose

Sudo apt update && sudo apt install docker-compose

Subhakanta Mishra
Subhamishra.in@gmail.com

Step5: Creates a **private Docker registry** as a Swarm service.

docker service create --name registry --publish target=5000,published=5000 registry:2

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker service create --name registry --publish target=5000,published=5000 registry:2
xcbmd7dm37s3kv2ep9mrm4mka
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service xcbmd7dm37s3kv2ep9mrm4mka converged
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step6: Verifies that the local registry is running.

curl localhost:5000/v2/_catalog

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:5000/v2/_catalog
{"repositories":[]}
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step7: Building image using the local Dockerfile and tags it as stackdemo_web.

docker-compose -f compose.yaml build

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker-compose -f compose.yaml build
redis uses an image, skipping
Building web
[+] Building 31.9s (8/8) FINISHED                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                                  0.0s
 => => transferring dockerfile: 148B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/python:2.7                                                         2.7s
 => [internal] load .dockerignore                                                                                     0.0s
 => => transferring context: 2B                                                                                       0.0s
 => [internal] load build context                                                                                     0.0s
 => => transferring context: 1.01kB                                                                                   0.0s
 => [1/3] FROM docker.io/library/python:2.7@sha256:cfa62318c459b1fde9e0841c619906d15ada5910d625176e24bf692cf8a2601d  20.2s
 => => resolve docker.io/library/python:2.7@sha256:cfa62318c459b1fde9e0841c619906d15ada5910d625176e24bf692cf8a2601d  0.0s
 => => sha256:de52085267e18aba7083768f283765fcd04f753aba32cc85e62a39c77670cf4b 2.22kB / 2.22kB                       0.0s
 => => sha256:d080dad46627d4103daddf54c89f2ef0a0b72ba8270066c50e95ffbf4a79362e 49.17MB / 49.17MB                     1.5s
 => => sha256:454b6b37aa5fec6dcadc18e45dc6d58e4cf4b967540152c4f44030737e2f5351 9.98MB / 9.98MB                       1.2s
 => => sha256:cfa62318c459b1fde9e0841c619906d15ada5910d625176e24bf692cf8a2601d 2.14kB / 2.14kB                       0.0s
 => => sha256:c7f4559769f8892b260cf86e99ed89b11abf2619c66e7cdc9d83c9c971ef62fd 8.96kB / 8.96kB                       0.0s
 => => sha256:0d8c76f347939ef96b2815e45f9fa43995940a8d4a01525012b2bce1adcbfc3a 7.68MB / 7.68MB                       0.6s
 => => sha256:0caa3b2294305ed3cf1d91d1fe2e2dbf3e294a694a4f636dea6bf733975e237a 52.16MB / 52.16MB                     2.1s
 => => sha256:3b47ba45af89a4c13c0c56e12431cbedfa1e0836733d37eb58461934ebca5619 183.71MB / 183.71MB                   4.6s
 => => extracting sha256:d080dad46627d4103daddf54c89f2ef0a0b72ba8270066c50e95ffbf4a79362e                            3.6s
 => => sha256:7265f9167afe68b7ec760a436cba685022ddddf6d09978e92d73fa39ad7fa13e 5.91MB / 5.91MB                       3.1s
 => => sha256:2bfc998b0cd02511fa6e90b283528652ebd6256a58d309a0f2a021bd737953e0 17.99MB / 17.99MB                     2.9s
 => => sha256:089c17f1d7bf843f324823e5517baf6f00f8d4c5db8e442a17880d62b3a8120a 1.89MB / 1.89MB                       3.7s
 => => sha256:1ce677d6f0522924f772aa7be36efd6365f59ef061c92a1512d91122f473d3fa 7.68MB / 7.68MB                       3.8s
 => => extracting sha256:0d8c76f347939ef96b2815e45f9fa43995940a8d4a01525012b2bce1adcbfc3a                            0.3s
 => => extracting sha256:454b6b37aa5fec6dcadc18e45dc6d58e4cf4b967540152c4f44030737e2f5351                            0.2s
 => => extracting sha256:0caa3b2294305ed3cf1d91d1fe2e2dbf3e294a694a4f636dea6bf733975e237a                            3.0s
 => => extracting sha256:3b47ba45af89a4c13c0c56e12431cbedfa1e0836733d37eb58461934ebca5619                            8.8s
 => => extracting sha256:7265f9167afe68b7ec760a436cba685022ddddf6d09978e92d73fa39ad7fa13e                            0.3s
 => => extracting sha256:2bfc998b0cd02511fa6e90b283528652ebd6256a58d309a0f2a021bd737953e0                            0.8s
 => => extracting sha256:089c17f1d7bf843f324823e5517baf6f00f8d4c5db8e442a17880d62b3a8120a                            0.2s
 => => extracting sha256:1ce677d6f0522924f772aa7be36efd6365f59ef061c92a1512d91122f473d3fa                            0.4s
 => [2/3] COPY . /tmp                                                                                                 5.5s
 => [3/3] RUN pip install -r /tmp/requirements.txt                                                                    3.3s
 => exporting to image                                                                                                0.1s
 => => exporting layers                                                                                               0.1s
 => => writing image sha256:83589b1b9a6043f952210d2bdb40ed7a7731923ed3fca37722bc14c9c11507d2                         0.0s
 => => naming to localhost:5000/stackdemo_web:1.0                                                                     0.0s
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Subhakanta Mishra
Subhamishra.in@gmail.com

Step8: Verifies the image was built successfully.

docker images

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker images
REPOSITORY                      TAG       IMAGE ID       CREATED         SIZE
localhost:5000/stackdemo_web    1.0       83589b1b9a60   2 minutes ago   853MB
registry                        <none>    33eeff39e0aa   21 months ago   25MB
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step9: Pushes your built image to the **local registry at localhost:5000**

docker-compose -f compose.yaml push

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker-compose -f compose.yaml push
Pushing web (localhost:5000/stackdemo_web:1.0)...
The push refers to repository [localhost:5000/stackdemo_web]
0c403f830f87: Pushed
eae631242601: Pushed
e49104642ec4: Pushed
a4a2d56ebc0d: Pushed
2f1421ed21bd: Pushed
6291a9d15790: Pushed
4f148ec7a82f: Pushed
bb3103538786: Pushed
4a8b48ebe9d5: Pushed
745a007844b3: Pushed
7f79b7253b84: Pushed
1.0: digest: sha256:3f0080ce20e211e6503a5c084ee5b5b04cc0382b0befae164817ab94ea5307da size: 2639
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step10: Now should show your pushed image

curl localhost:5000/v2/_catalog

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:5000/v2/_catalog
{"repositories":["stackdemo_web"]}
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step11: Deploys the app as a Swarm stack named stackdemo.

docker stack deploy --compose-file compose.yaml stackdemo

Subhakanta Mishra
Subhamishra.in@gmail.com

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker stack deploy --compose-file compose.yaml stackdemo
Ignoring unsupported options: build, links

Ignoring deprecated options:

expose: Exposing ports is unnecessary - services on the same network can access each other's containers on any port.

Since --detach=false was not specified, tasks will be created in the background.
In a future release, --detach=false will become the default.
Creating network stackdemo_mynet
Creating service stackdemo_web
Creating service stackdemo_redis
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step12: Lists services created by the stack.

docker service ls

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker service ls
ID             NAME             MODE         REPLICAS   IMAGE                              PORTS
xcbmd7dm37s3   registry         replicated   1/1        registry:2                         *:5000->5000/tcp
edw54wuzoxij   stackdemo_redis  replicated   1/1        redis:latest
r4qfahtm4frv   stackdemo_web    replicated   3/3        localhost:5000/stackdemo_web:1.0   *:8000->5000/tcp
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step13: Accesses the running container via published port 8000

curl localhost:8000

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:8000
Hello Container World! I have been seen 1 times and my hostname is 2f4ba6855037.
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:8000
Hello Container World! I have been seen 2 times and my hostname is 4330041a44ae.
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:8000
Hello Container World! I have been seen 3 times and my hostname is 8df6c4336d3b.
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:8000
Hello Container World! I have been seen 4 times and my hostname is 2f4ba6855037.
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ curl localhost:8000
Hello Container World! I have been seen 5 times and my hostname is 4330041a44ae.
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Step14: Deploying Visualizer

docker service create --name=viz --publish=8080:8080/tcp --constraint=node.role==manager --mount=type=bind,src=/var/run/docker.sock,dst=/var/run/docker.sock alexellis2/visualizer-arm:latest

```
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$ docker service create   --name=viz   --publish=8080:8080/tcp   --constraint=node.role==manager
   --mount=type=bind,src=/var/run/docker.sock,dst=/var/run/docker.sock   alexellis2/visualizer-arm:latest
fgy0f1v1ohjk46bec4mw3pqc0
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service fgy0f1v1ohjk46bec4mw3pqc0 converged
(base) labuser@ip-172-31-19-12:~/k8s_material/docker_compose$
```

Subhakanta Mishra
Subhamishra.in@gmail.com

Step15: Open the browser to see the containers.