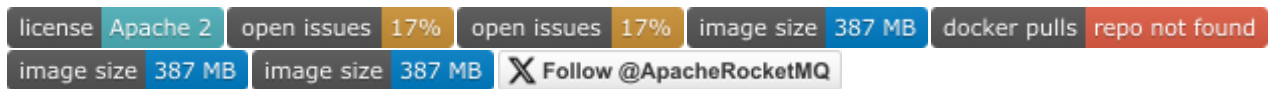

RocketMQ-Docker



This is the Git repo of the Docker Image for Apache RocketMQ and official docker hub repo: <https://hub.docker.com/repository/docker/apache/rocketmq> . You could run it through the following ways:

1. Generate a RocketMQ Docker image
2. Run the docker image with the below modes: 2.1. Single Node. 2.2. Cluster with docker-compose. 2.3. Cluster on Kubernetes. 2.4. Cluster of Dledger storage
3. TLS support
4. Generate a RocketMQ Dashboard Docker image

Prerequisites

The Docker images in this repository should support Docker version 1.12+, and Kubernetes version 1.9+.

Quick start

A. Generate a RocketMQ docker image

Note: This is an experimented code to allow users to build docker image locally according to a given RocketMQ version. Actually the formal images have been generated by RocketMQ official maintainer and stored in docker hub. Suggest common users to use these remote images directly.

```
1 cd image-build
2 sh build-image.sh RMQ-VERSION BASE-IMAGE
```

Tip: The supported RMQ-VERSIONs can be obtained from [here](#). The supported BASE-IMAGEs are [centos, alpine]. For example: `sh build-image.sh 4.5.0 alpine`

B. Stage a specific version

Users can generate a runtime (stage) directory based on a specific version and docker style operate the RocketMQ cluster/server/nameserver beneath the directory.

```
1 sh stage.sh RMQ-VERSION
```

Note: RMQ-VERSION is the tag of the RocketMQ image. After executing the above shell script, (e.g. sh stage.sh 4.5.0), it will generate a stage directory (./stages/4.5.0). User can do the following works under the directory, assuming the RMQ-version is defined with 4.5.0.

2.1 Single Node Run:

```
1 cd stages/4.5.0
2
3 ./play-docker.sh alpine
```

2.2 Cluster with docker-compose Run:

```
1 cd stages/4.5.0
2
3 ./play-docker-compose.sh
```

2.3 Cluster on Kubernetes Run:

```
1 cd stages/4.5.0
2
3 ./play-kubernetes.sh
```

2.4 Cluster of Dledger storage Run: (Note: This feature needs RMQ version is 4.4.0 or above)

```
1 cd stages/4.5.0
2
3 ./play-docker-dledger.sh
```

3. TLS support

Run: (It will startup nameserver and broker with SSL enabled style. The client will not invoke nameserver or broker until related SSL client is configured.)

You can see detailed TLS config instruction from [here](#)

```
1 cd stages/4.5.0
2
3 ./play-docker-tls.sh
4
5 # Once nameserver and broker startup correctly, you still can use the
  following script to test produce/consume in SSL mode, why, due to
  they still use the SSL setting which exists in JAVA-OPT of the
  docker rmqbroker container.
```

```
6 ./play-producer.sh
7 ./play-consumer.sh
```

4. Generate a RocketMQ Dashboard Docker image

- 4.1 build command

```
1 cd image-build && sh build-image-dashboard.sh `VERSION` centos
2
3 demo: sh build-image-dashboard.sh 1.0.0 centos
```

- 4.2 start command

```
1 sh product/start-dashboard.sh `VERSION`
2
3 demo: sh product/start-dashboard.sh 1.0.0
```

How to update RocketMQ image repository using update.sh

Run:

```
1 cd image-build
2 ./update.sh
```

This script will get the latest release version of RocketMQ and build the docker images based on [alpine](#) and [centos](#) respectively, then push the new images to the current official repository [apache/rocketmq](#).

How to verify RocketMQ works well

Verify with Docker and docker-compose

1. Use `docker ps | grep rmqbroker` to find your RocketMQ broker container id.
2. Use `docker exec -it {container_id} ./mqadmin clusterList -n {nameserver_ip}:9876` to verify if RocketMQ broker works, for example:

```
1 root$ docker exec -it 63950574b491 ./mqadmin clusterList -n
    192.168.43.56:9876
2 OpenJDK 64-Bit Server VM warning: ignoring option PermSize=128m;
    support was removed in 8.0
3 OpenJDK 64-Bit Server VM warning: ignoring option MaxPermSize=128m;
    support was removed in 8.0
```

4	#Cluster Name	#Broker Name	#BID	#Addr	
	#Version	#InTPS(LOAD)		#OutTPS(LOAD)	#PCWait(ms)
	#Hour #SPACE				
5	DefaultCluster	63950574b491	0	172.17.0.3:10911	
	V4_3_0	0.00(0,0ms)		0.00(0,0ms)	0
	429398.92 -1.0000				

Verify with Kubernetes

1. Use `kubectl get pods | grep rocketmq` to find your RocketMQ broker Pod id, for example:

```
1 [root@k8s-master rocketmq]# kubectl get pods |grep rocketmq
2 rocketmq-7697d9d574-b5z7g          2/2      Running      0
   2d
```

2. Use `kubectl -n {namespace} exec -it {pod_id} -c broker bash` to login the broker pod, for example:

```
1 [root@k8s-master rocketmq]# kubectl -n default exec -it rocketmq-7697
   d9d574-b5z7g -c broker bash
2 [root@rocketmq-7697d9d574-b5z7g bin]#
```

3. Use `mqadmin clusterList -n {nameserver_ip}:9876` to verify if RocketMQ broker works, for example:

```
1 [root@rocketmq-7697d9d574-b5z7g bin]# ./mqadmin clusterList -n
   localhost:9876
2 OpenJDK 64-Bit Server VM warning: ignoring option PermSize=128m;
   support was removed in 8.0
3 OpenJDK 64-Bit Server VM warning: ignoring option MaxPermSize=128m;
   support was removed in 8.0
4 #Cluster Name      #Broker Name      #BID  #Addr
   #Version          #InTPS(LOAD)      #OutTPS(LOAD) #PCWait(ms)
   #Hour #SPACE
5 DefaultCluster    rocketmq-7697d9d574-b5z7g  0      192.168.196.14:10911
   V4_3_0            0.00(0,0ms)        0.00(0,0ms)
   0 429399.44 -1.0000
```

So you will find it works, enjoy !

C. Product level configuration

The project also provides a usage reference for product level cluster docker configuration and startup. Please see the README.md details in /product directory.

FAQ

1. If I want the broker container to load my customized configuration file (which means `broker.conf`) when it starts, how can I achieve this? First, create the customized `broker.conf`, like below:

```
1 brokerClusterName = DefaultCluster
2 brokerName = broker-a
3 brokerId = 0
4 deleteWhen = 04
5 fileReservedTime = 48
6 brokerRole = ASYNC_MASTER
7 flushDiskType = ASYNC_FLUSH
8 #set `brokerIP1` if you want to set physical IP as broker IP.
9 brokerIP1=10.10.101.80 #change you own physical IP Address
```

And put the customized `broker.conf` file at a specific path, like “`pwd/data/broker/conf/broker.conf`”.

Then we can modify the `play-docker.sh` and volume this file to the broker container when it starts. For example:

```
1 docker run -d -p 10911:10911 -p 10909:10909 -v `pwd`/data/broker/logs:/
  root/logs -v `pwd`/data/broker/store:/root/store -v `pwd`/data/
  broker/conf/broker.conf:/home/rocketmq/rocketmq-4.5.0/conf/broker.
  conf --name rmqbroker --link rmqnamesrv:namesrv -e "NAMESRV_ADDR=
  namesrv:9876" apache/rocketmq:4.5.0 sh mqbroker -c /home/rocketmq/
  rocketmq-4.5.0/conf/broker.conf
```

Finally we can find the customized `broker.conf` has been used in the broker container. For example:

```
1 MacBook-Pro:4.5.0 huan$ docker ps |grep mqbroker
2 a32c67aed6dd      apache/rocketmq:4.5.0      "sh mqbroker"          20
   minutes ago      Up 20 minutes             0.0.0.0:10909->10909/tcp, 9876/
   tcp, 0.0.0.0:10911->10911/tcp    rmqbroker
3 MacBook-Pro:4.5.0 $ docker exec -it a32c67aed6dd cat /home/rocketmq/
  rocketmq-4.5.0/conf/broker.conf
4 brokerClusterName = DefaultCluster
5 brokerName = broker-a
6 brokerId = 0
7 deleteWhen = 04
8 fileReservedTime = 48
9 brokerRole = ASYNC_MASTER
10 flushDiskType = ASYNC_FLUSH
11 #set `brokerIP1` if you want to set physical IP as broker IP.
12 brokerIP1=10.10.101.80 #change you own physical IP Address
```

In the case of docker-compose, change the `docker-compose.yml` like following:

```
1 version: '2'
2 services:
3   namesrv:
4     image: apache/rocketmq:4.5.0
5     container_name: rmqnamesrv
6     ports:
7       - 9876:9876
8     volumes:
9       - ./data/namesrv/logs:/home/rocketmq/logs
10    command: sh mqnamesrv
11  broker:
12    image: apache/rocketmq:4.5.0
13    container_name: rmqbroker
14    ports:
15      - 10909:10909
16      - 10911:10911
17      - 10912:10912
18    volumes:
19      - ./data/broker/logs:/home/rocketmq/logs
20      - ./data/broker/store:/home/rocketmq/store
21      - ./data/broker/conf/broker.conf:/home/rocketmq/rocketmq-4.5.0/
22        conf/broker.conf
23    command: sh mqbroker -n namesrv:9876 -c ../conf/broker.conf
24    depends_on:
25      - namesrv
```