

Installation and Configuration of RabbitMQ on Amazon EC2

1. Link to install Docker on an EC2.
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-basics.html#install_docker
2. Link to create a RabbitMQ server onto a Docker container.
<https://docs.docker.com/samples/library/rabbitmq/>

Main command for running container:

docker run -d --name <CONTAINER_NAME> -p 5672:5672 -p 15672:15672 rabbitmq:3-management

- “-d” is a flag to tell docker to run the container in the background
- “--name <CONTAINER_NAME>” is command to assign a name to container. Without this flag, docker will assign a random name to the container
- “-p {ec2Port:containerPort}”

The most **important** flag of the run command. This is needed in order to tell the ec2 what request on its ports need to be mapped to the container. Without this the docker container simply will not receive any request. So it's good to keep them the same as defined for rabbitmq but it is possible to change these

Ex: -p 8080:15672 will map all request to the ec2 on port 8080 to containers personal listening port of 15672. ec2ip:8080 => dockercontainer:15672.

3. Accessing the RabbitMQ is possible through the enabled management plugin. This can be done by entering “<http://ec2ip:port>” into a web browser and using the default credentials.
Username: “guest”, Password: “guest”



4. Once logged into the configuration there is an option at the bottom that allows you to import the definitions which is included in the configuration service's src/main/resources called rabbit_hydra-rabbit_2018-3-14 of the first iteration of Hydra.

The screenshot shows the RabbitMQ Overview page. At the top, it displays the version (3.7.3) and Erlang version (20.2.2). The navigation bar includes Overview, Connections, Channels, Exchanges, Queues, and Admin. Below the navigation bar, there are global counts: Connections: 17, Channels: 47, Exchanges: 11, Queues: 47, and Consumers: 43. The 'Nodes' section shows a table with details for the node 'rabbit@hydra-rabbit', including file descriptors, socket descriptors, Erlang processes, memory usage, disk space, and uptime. The 'Export definitions' section has a button to 'Download broker definitions'. The 'Import definitions' section has a button to 'Upload broker definitions'.

Name	File descriptors ?	Socket descriptors ?	Erlang processes	Memory ?	Disk space	Uptime	Info	Reset stats
rabbit@hydra-rabbit	61 1024 available	19 829 available	794 1048576 available	105MB 397MB high watermark	6.2GB 48MB low watermark	5d 3h	basic disc 1 rss	This node All nodes

5. Importing of definitions is successful if exchanges and queues tab look similar to below screenshots

The screenshot shows the RabbitMQ Exchanges page. It displays a table of all exchanges (11). The table has columns for Name, Type, Features, Message rate in, and Message rate out. The exchanges listed are: (AMQP default), amq.direct, amq.fanout, amq.headers, amq.match, amq.rabbitmq.trace, amq.topic, revature.hydra.dto, revature.hydra.repos, revature.hydra.service, and springCloudHystrixStream.

Name	Type	Features	Message rate in	Message rate out
(AMQP default)	direct	D	0.40/s	
amq.direct	direct	D		
amq.fanout	fanout	D		
amq.headers	headers	D		
amq.match	headers	D		
amq.rabbitmq.trace	topic	D I		
amq.topic	topic	D		
revature.hydra.dto	topic	D		
revature.hydra.repos	topic	D	0.00/s	0.00/s
revature.hydra.service	topic	D		
springCloudHystrixStream	topic	D		

The screenshot shows the RabbitMQ Queues page. It displays a table of all queues (47). The table has columns for Name, Features, State, Ready, Unacked, and Total. The queues listed are: revature.hydra.dto.assessment, revature.hydra.dto.grade, revature.hydra.repos.address, revature.hydra.repos.address.list, revature.hydra.repos.assessment, revature.hydra.repos.assessment.list, revature.hydra.repos.batch, revature.hydra.repos.batch.list, revature.hydra.repos.curriculum, revature.hydra.repos.curriculum.list, and revature.hydra.repos.grade.

Name	Features	State	Ready	Unacked	Total
revature.hydra.dto.assessment	D	idle	0	0	
revature.hydra.dto.grade	D	idle	0	0	
revature.hydra.repos.address	D	idle	0	0	
revature.hydra.repos.address.list	D	idle	0	0	
revature.hydra.repos.assessment	D	idle	0	0	
revature.hydra.repos.assessment.list	D	idle	0	0	
revature.hydra.repos.batch	D	idle	0	0	
revature.hydra.repos.batch.list	D	idle	0	0	
revature.hydra.repos.curriculum	D	idle	0	0	
revature.hydra.repos.curriculum.list	D	idle	0	0	
revature.hydra.repos.grade	D	idle	0	0	

Adding more exchanges/queues to RabbitMQ.

1. Click on the appropriate tab to open either an exchange or queue.
2. At the bottom there are options to fill out a new exchange/queue.
3. Make sure to follow proper naming conventions (revature.)
4. Arguments are not necessary but allow for customization of the exchange or queue.
5. After the exchange or queue fits specifications then click the add button.
6. Messaging is handled in each microservice within their respective service package that you can reference.
 - a. Example for Batch Service: (located in com.revature.hydra.batch.service)
“BatchCompositionMessageService” contains calls to RabbitMQ Server exchange and queues
“BatchRepositoryMessageService” contains listeners to RabbitMQ Server queue

Other useful Docker commands:

List Containers:

```
docker ps
```

Delete container

```
docker rm <CONTAINER_ID>
```

List all exited containers

```
docker ps -aq -f status=exited
```

Remove stopped containers

```
docker ps -aq --no-trunc | xargs docker rm
```

To bash into running container

```
docker exec -t -i <CONTAINER_NAME> /bin/bash
```

Start an existing container

```
docker start <CONTAINER_NAME>
```

Get information about the container

```
docker inspect <CONTAINER_ID>
```

Start Docker Daemon

```
sudo service docker start
```