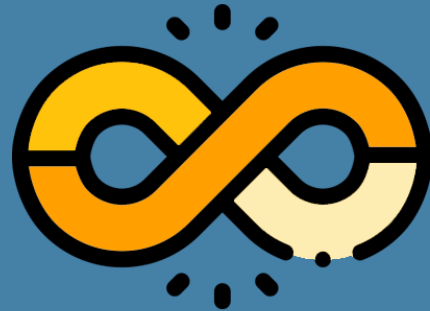


DevOps -
Infrastructure &
Configuration
Management



DevOps

Infrastructure & Configuration Management

April 2022

Message Brokering with RabbitMQ

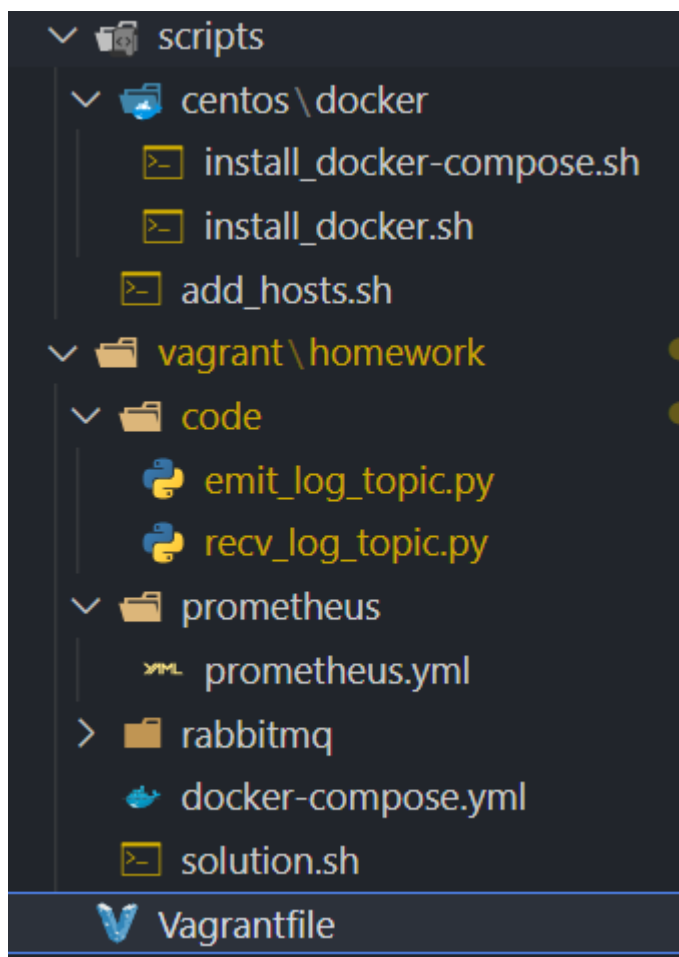
Home Work

Stefan Veselinov

Assignment

You are expected to execute the following:

1. Create a **single machine** with **Docker** installed
 - Create a **three-node RabbitMQ** cluster
 - Add a policy to distribute/replicate queues to all nodes
 - Run the last example (Topics) either on the command line on the Docker host or as containers
 - i. One producer
 - ii. One consumer subscribed to all warn and crit messages
 - iii. One consumer subscribed to all ram related messages
 - Deploy **Prometheus** as a container and make it collect metrics for all three nodes of the cluster
 - Deploy **Grafana** as a container and add Prometheus as a data source
 - Prepare at least one visualization to display any of the metrics exposed by the nodes



Solution Structure

- scripts - provisioning scripts for initial setup, installation, and config. Scripts for Docker and Docker-Compose installation
- vagrant – shared folder, contains code for emitter and receiver, Prometheus configurations, Volumes for docker containers, main docker-compose script

Solution Setup

- Enter solution directory (where Vagrant file is placed).
 - Execute – vagrant up
- Vagrant is going to create 1-vm
 - Docker – Fully running docker and docker compose instances
 - Docker Compose is going to be used for creation of RabbitMQ cluster and monitoring by Prometheus and Grafana
- Vagrant ssh docker
 - Execute row by row commands from /vagrant/homework/solution.sh

```
echo "*** Installing Packages"
sudo dnf install -y python3
sudo python3 -m pip install pika --upgrade

echo "*** Starting RabbitMQ Containers"
docker-compose -f /vagrant/homework/docker-compose.yml up -d --build

echo "*** Adding Pluggins to RabbitMQ Containers"
docker container exec -it rabbitmq-1 rabbitmq-plugins enable rabbitmq_federation
docker container exec -it rabbitmq-2 rabbitmq-plugins enable rabbitmq_federation
docker container exec -it rabbitmq-3 rabbitmq-plugins enable rabbitmq_federation

docker container exec -it rabbitmq-1 rabbitmq-plugins enable rabbitmq_prometheus
docker container exec -it rabbitmq-2 rabbitmq-plugins enable rabbitmq_prometheus
docker container exec -it rabbitmq-3 rabbitmq-plugins enable rabbitmq_prometheus

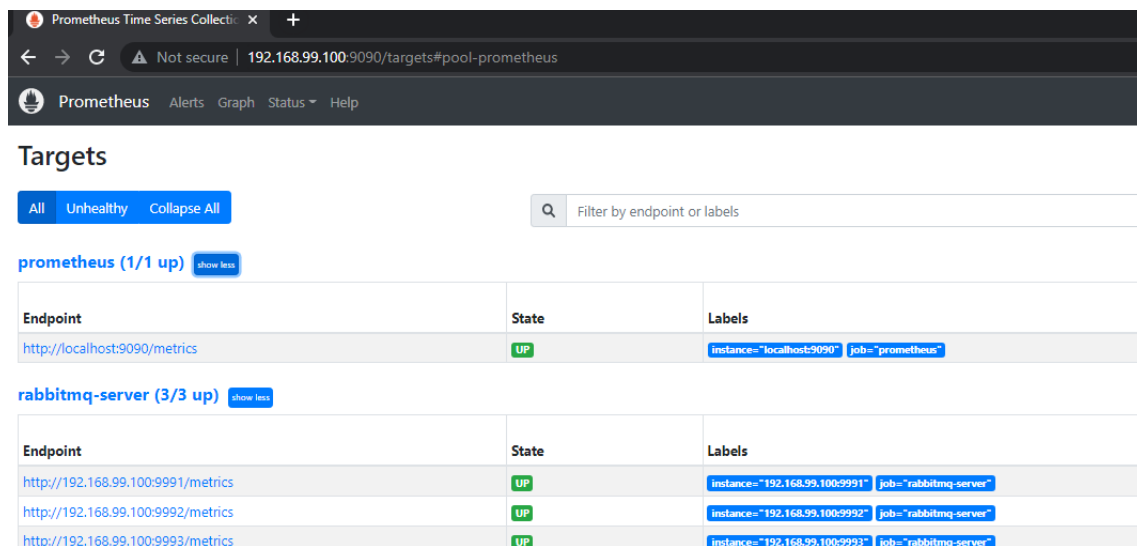
echo "*** Adding Policy"
docker exec -it rabbitmq-1 rabbitmqctl set_policy ha-fed ".*" '{"ha-sync-mode":"automatic", "ha-mode":"nodes", "ha-params":["rabbit@rabbitmq-1","rabbit@rabbitmq-2","rabbit@rabbitmq-3"]}' --priority 1 --apply-to queues

echo "*** Starting Emitter"
python3 /vagrant/homework/code/emit_log_topic.

echo "*** Starting Consumers"
python3 /vagrant/homework/code/recv_log_topic.py ".*.warm" ".*.crit"

echo "*** Starting Consumers"
python3 /vagrant/homework/code/recv_log_topic.py "ram.*"
```

- Check Prometheus Targets



The screenshot shows the Prometheus web interface at the URL `192.168.99.100:9090/targets#pool-prometheus`. The page title is "Prometheus Time Series Collector". The main heading is "Targets". There are buttons for "All", "Unhealthy", and "Collapse All". A search bar says "Filter by endpoint or labels".

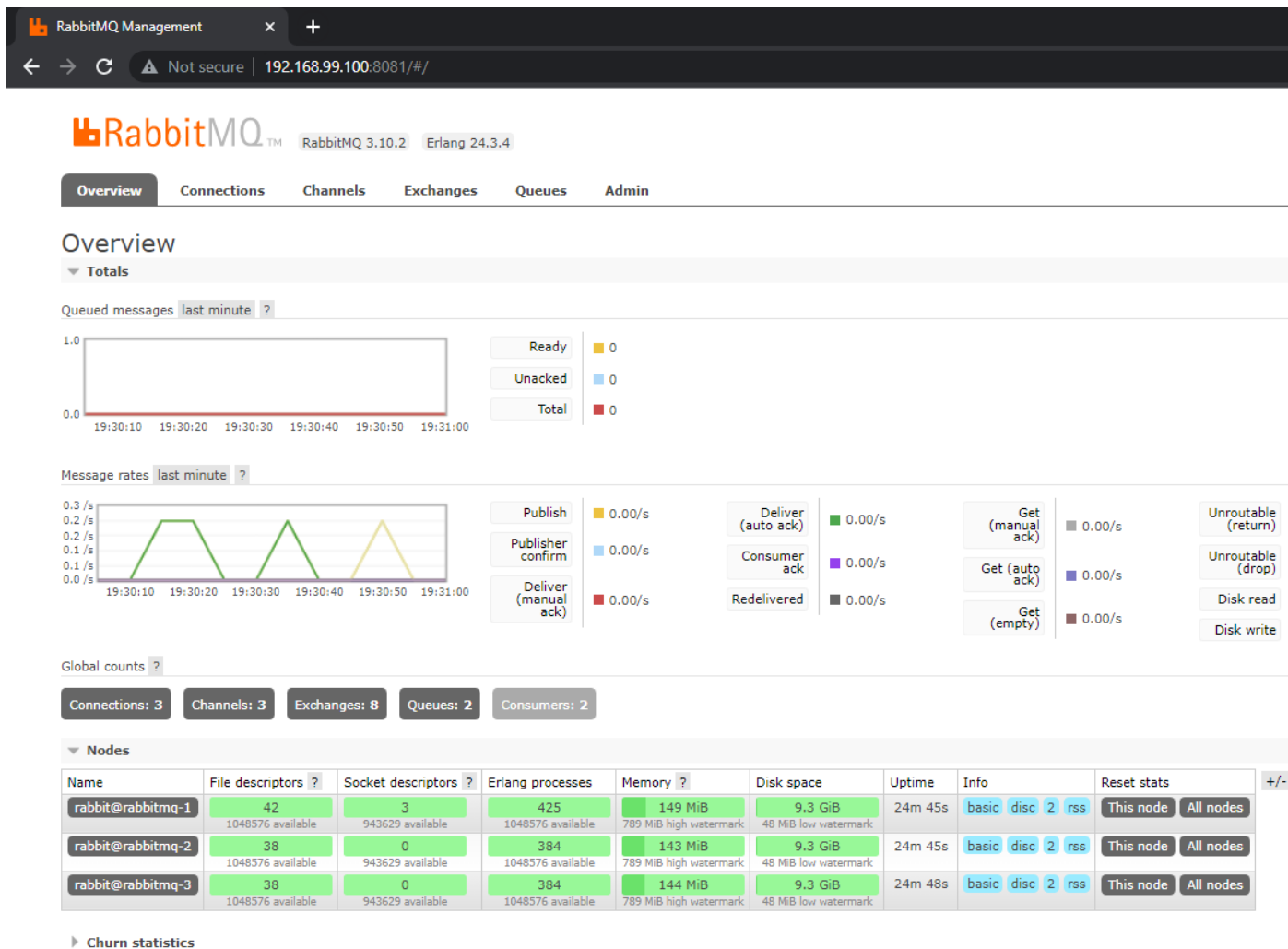
Under the heading "prometheus (1/1 up)", there is a table with one row:

Endpoint	State	Labels
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"

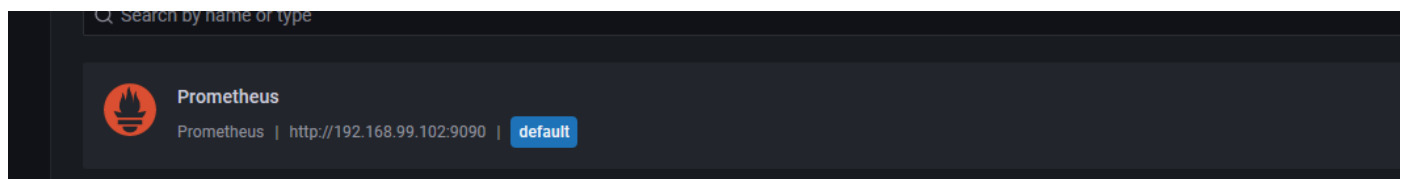
Under the heading "rabbitmq-server (3/3 up)", there is a table with three rows:

Endpoint	State	Labels
http://192.168.99.100:9991/metrics	UP	instance="192.168.99.100:9991" job="rabbitmq-server"
http://192.168.99.100:9992/metrics	UP	instance="192.168.99.100:9992" job="rabbitmq-server"
http://192.168.99.100:9993/metrics	UP	instance="192.168.99.100:9993" job="rabbitmq-server"

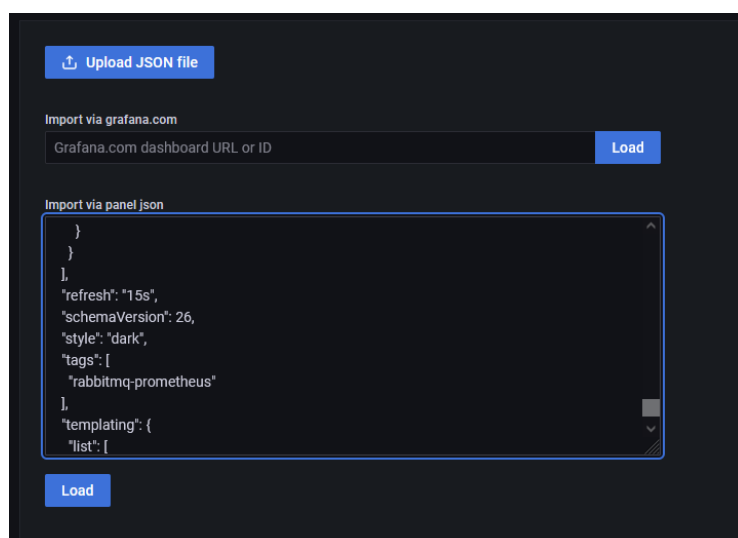
- Check RabbitMq



- Login to Grafana - <http://192.168.99.100:3000>
- Set Grafana data source



- Import Grafana Dashboard



JSON: [Source](#)

- Go to Dashboard

