# **Задание 1**

Используя Vagrant или VirtualBox, создайте виртуальную машину и установите RabbitMQ. Добавьте management plug-in и зайдите в веб-интерфейс.

*Итогом выполнения домашнего задания будет приложенный скриншот веб-интерфейса RabbitMQ.*

# **Решение 1**

docker-compose:

version: '3'

services:

rabbitmq-server:

image: rabbitmq:3.9

container\_name: rabbitmq-server

ports:

- "5672:5672"

networks:

- rabbitmqnet

rabbitmq-managment:

image: rabbitmq:3.9-management

container\_name: rabbitmq-management

ports:

- "8080:15672"

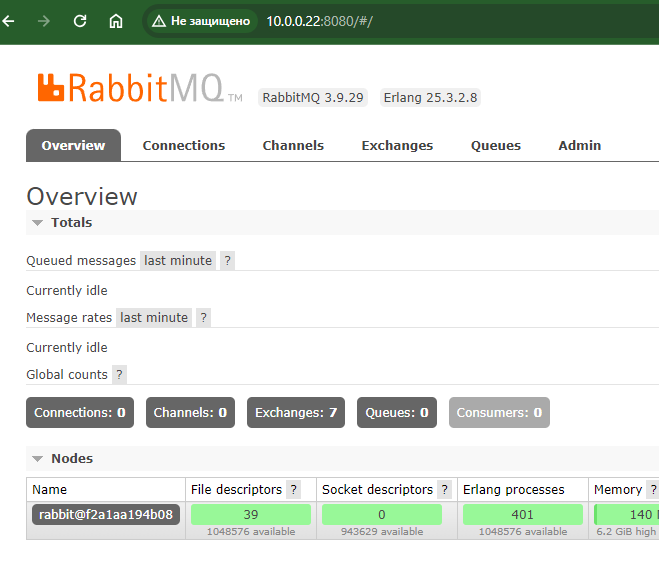
networks:

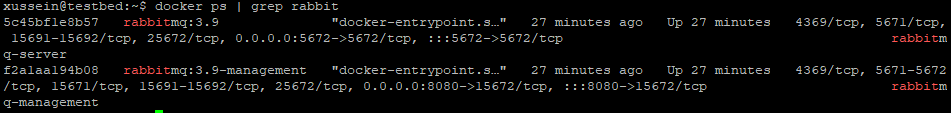
- rabbitmqnet

networks:

rabbitmqnet:

driver: bridge





# **Задание 2**

Используя приложенные скрипты, проведите тестовую отправку и получение сообщения. Для отправки сообщений необходимо запустить скрипт producer.py.

Для работы скриптов вам необходимо установить Python версии 3 и библиотеку Pika. Также в скриптах нужно указать IP-адрес машины, на которой запущен RabbitMQ, заменив localhost на нужный IP.

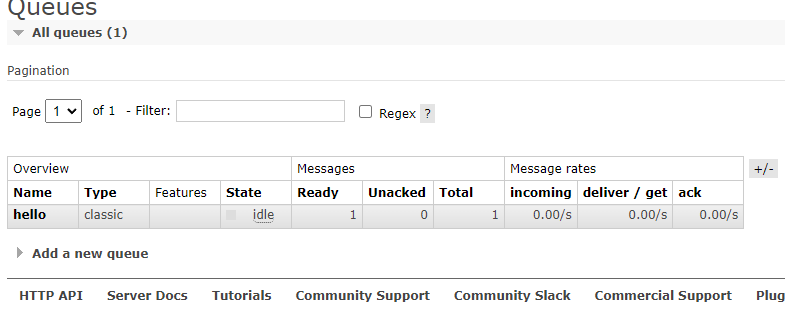
$ pip install pika

Зайдите в веб-интерфейс, найдите очередь под названием hello и сделайте скриншот. После чего запустите второй скрипт consumer.py и сделайте скриншот результата выполнения скрипта

*В качестве решения домашнего задания приложите оба скриншота, сделанных на этапе выполнения.*

Для закрепления материала можете попробовать модифицировать скрипты, чтобы поменять название очереди и отправляемое сообщение.

# **Решение 2**





# **Задание 3**

Используя Vagrant или VirtualBox, создайте вторую виртуальную машину и установите RabbitMQ. Добавьте в файл hosts название и IP-адрес каждой машины, чтобы машины могли видеть друг друга по имени.

Пример содержимого hosts файла:

$ cat /etc/hosts

192.168.0.10 rmq01

192.168.0.11 rmq02

После этого ваши машины могут пинговаться по имени.

Затем объедините две машины в кластер и создайте политику ha-all на все очереди.

*В качестве решения домашнего задания приложите скриншоты из веб-интерфейса с информацией о доступных нодах в кластере и включённой политикой.*

Также приложите вывод команды с двух нод:

$ rabbitmqctl cluster\_status

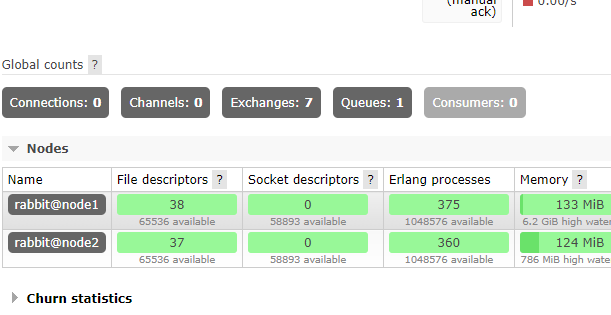
Для закрепления материала снова запустите скрипт producer.py и приложите скриншот выполнения команды на каждой из нод:

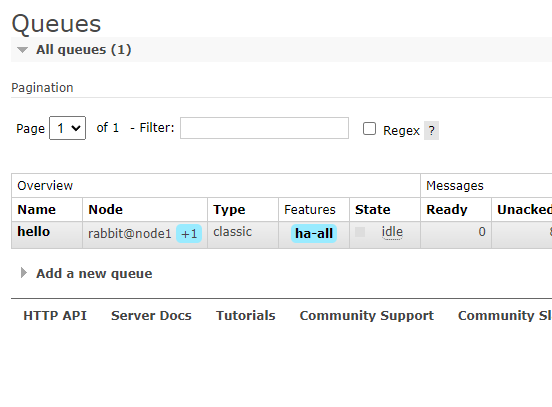
$ rabbitmqadmin get queue='hello'

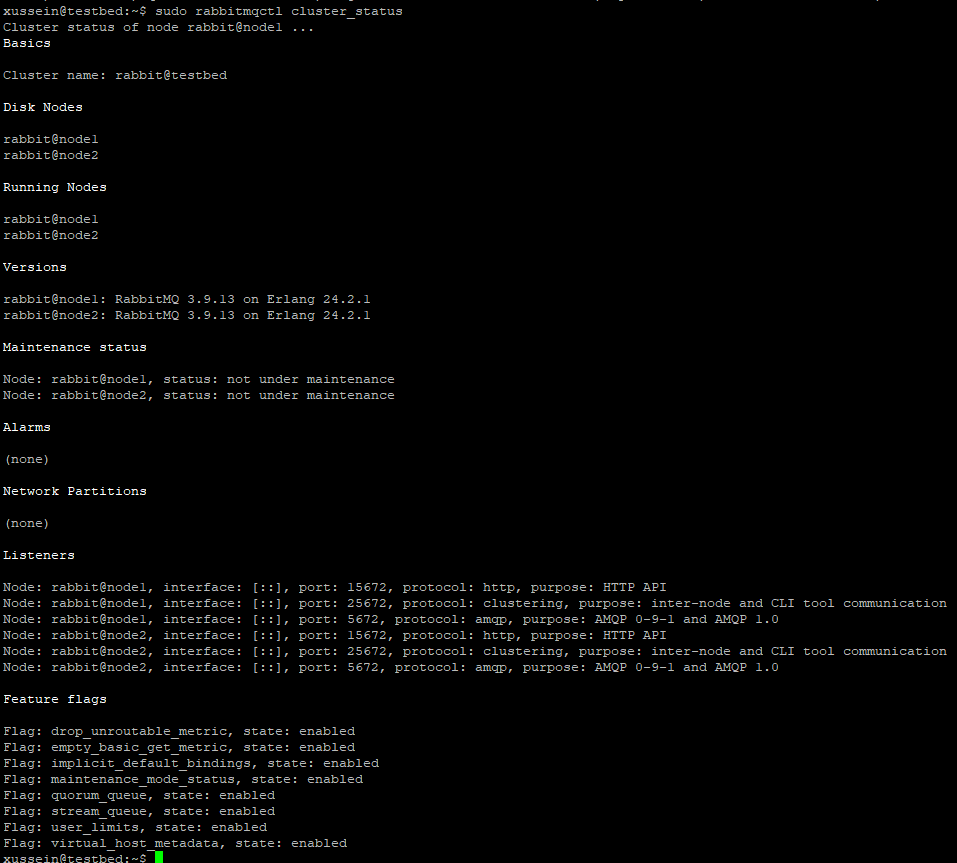
После чего попробуйте отключить одну из нод, желательно ту, к которой подключались из скрипта, затем поправьте параметры подключения в скрипте consumer.py на вторую ноду и запустите его.

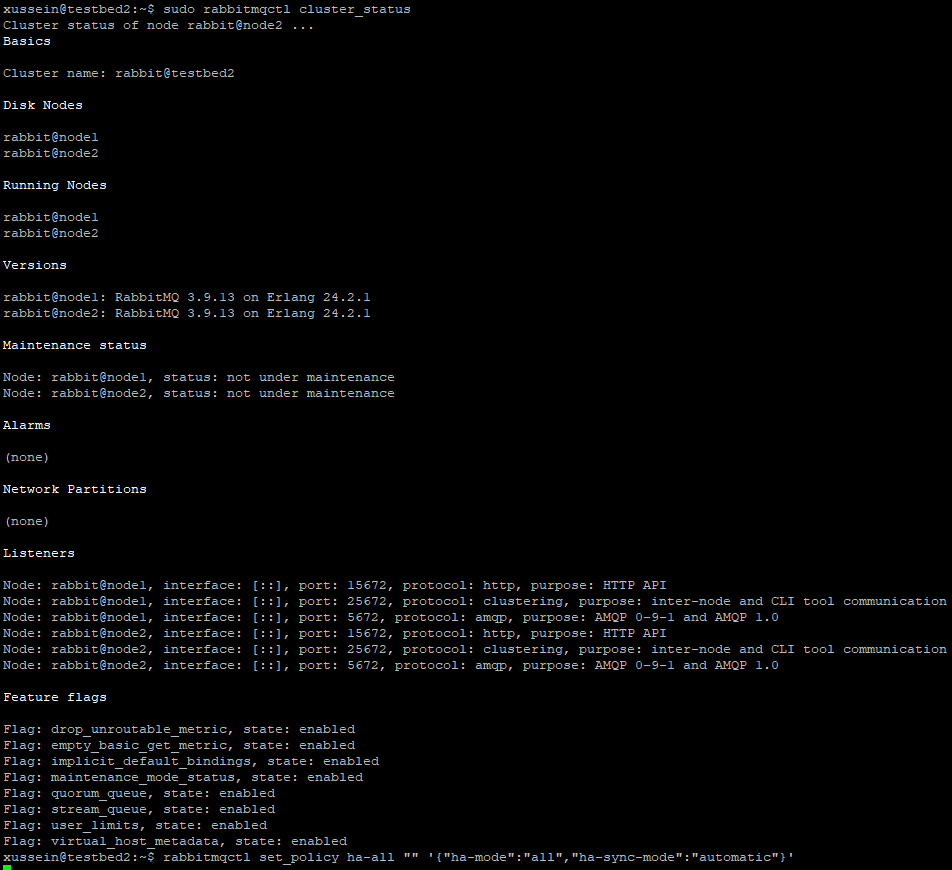
*Приложите скриншот результата работы второго скрипта.*

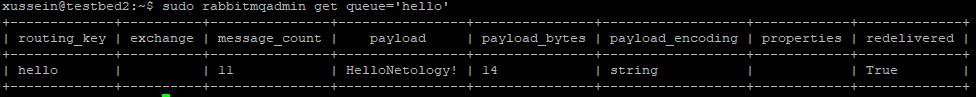
# **Решение 3**

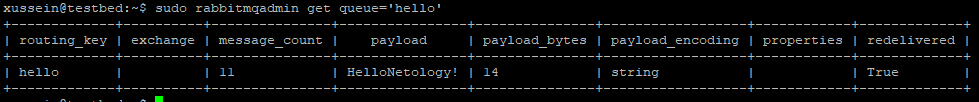












Отключил node1 (192.168.0.90), перевел код питона на вторую, консьюмер:

#!/usr/bin/env python

# coding=utf-8

import pika

credentials = pika.PlainCredentials('test', 'test')

parameters = pika.ConnectionParameters('192.168.0.89',

5672,

'/',

credentials)

connection = pika.BlockingConnection(parameters)

channel = connection.channel()

channel.queue\_declare(queue='hello')

def callback(ch, method, properties, body):

print (" [x] Received %r" % body)

channel.basic\_consume(queue='hello', on\_message\_callback=callback,auto\_ack=False)

channel.start\_consuming()

продюсер:

#!/usr/bin/env python

# coding=utf-8

import pika

credentials = pika.PlainCredentials('test', 'test')

parameters = pika.ConnectionParameters('192.168.0.89',

5672,

'/',

credentials)

connection = pika.BlockingConnection(parameters)

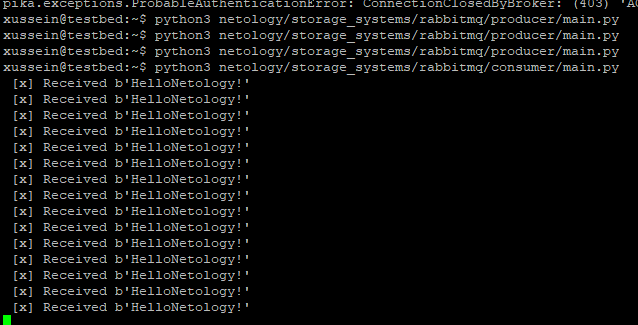
channel = connection.channel()

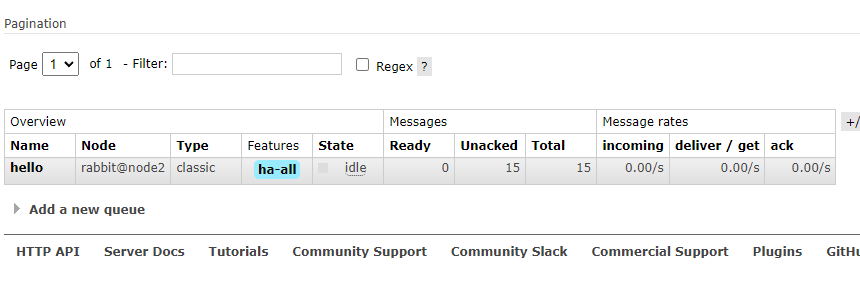
channel.queue\_declare(queue='hello')

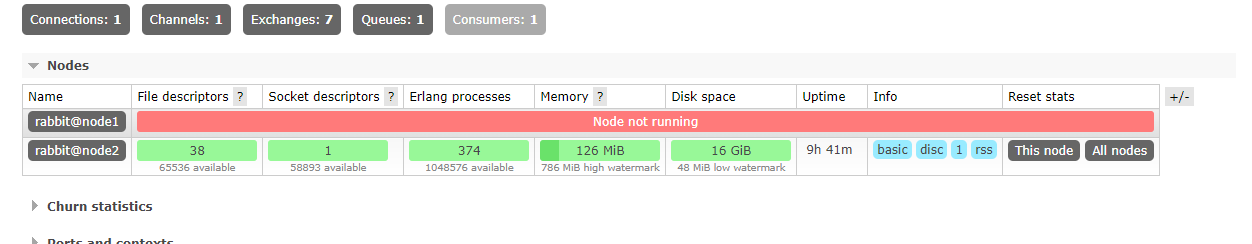
channel.basic\_publish(exchange='', routing\_key='hello', body='HelloNetology!')

connection.close()

При запуске с node1 кода питона на вторую ноду уходят сообщения:







# **Задание 4**

Напишите плейбук, который будет производить установку RabbitMQ на любое количество нод и объединять их в кластер. При этом будет автоматически создавать политику ha-all.

*Готовый плейбук разместите в своём репозитории.*

# **Решение 4**

Не успеваю 🙁

Успокаиваю себя только тем, что знаю как это делается и надеюсь что время найдется, чтобы задание выполнить. С автоматической кластеризацией и прочими прелестями жизни.