安装 Dapr CLI 脚手架工具 :

拷贝的tar包，放在 /usr/local/bin

初始化 Dapr :

dapr init

验证dapr版本 :

dapr --version

注意 : dapr初始化在/root/.dapr/bin未生成dashboard\_linux\_amd64.tar.gz和daprd文件，需要手动添加下并授权 chmod 777 daprd，

启动docker :

systemctl start docker

查看docker服务状态 :

systemctl status docker

验证容器正在运行

docker ps

启动rabbitmq :

docker run -d --hostname my-rabbit --name some-rabbit -p 5672:5672 -p 15672:15672 rabbitmq:3-management

重启rabbitmq :

docker restart some-rabbit

cd install/yifu\_deploy/talents-database/

测试使用python文件，提前安装python环境  
安装依赖  
pip3 install flask  
pip3 install flask\_cors

发布--

dapr run --app-id testpubsub --components-path ./talents-component --dapr-http-port 3500

订阅

dapr --app-id app1 --app-port 5001 --components-path ./talents-component run python3 app1.py

测试发布消息

1、Dapr CLI :

dapr publish --publish-app-id testpubsub --pubsub talents-database-rabbitmq-pubsub --topic deathStarStatus --data '{"status": "completed","data":"list"}'

2、HTTP API(BASH)

Curl -X POST http://localhost:3500/v1.0/publish/talents-database-rabbitmq-pubsub/deathStarStatus -H "Content-Type: application/json" -d '{ "status": "completed"}'

订阅响应输出

==APP== {'id': '307bf301-0c46-4cbb-90e7-f1a72c3c7db6', 'specversion': '1.0', 'source': 'testpubsub', 'pubsubname': 'talents-database-rabbitmq-

pubsub', 'data': {'status': 'completed', 'data': 'list'}, 'datacontenttype': 'application/json', 'type': 'com.dapr.event.sent', 'topic': 'deathStarStatus',

'traceid': '00-c8ab56c0df71ed81a0d43554df26d868-7735062e199b2ac3-01'}

== APP == 127.0.0.1 - - [02/Nov/2021 17:47:06] "POST /dsstatus HTTP/1.1" 200 -

curl -X POST http://localhost:5001/dsstatus

[{"pubsubname": "pubsub", "topic": "newOrder", "route": "/orders", "metadata": {"rawPayload": "true"}}]

== APP == None

== APP == 127.0.0.1 - - [03/Nov/2021 09:40:09] "POST /dsstatus HTTP/1.1" 200 -

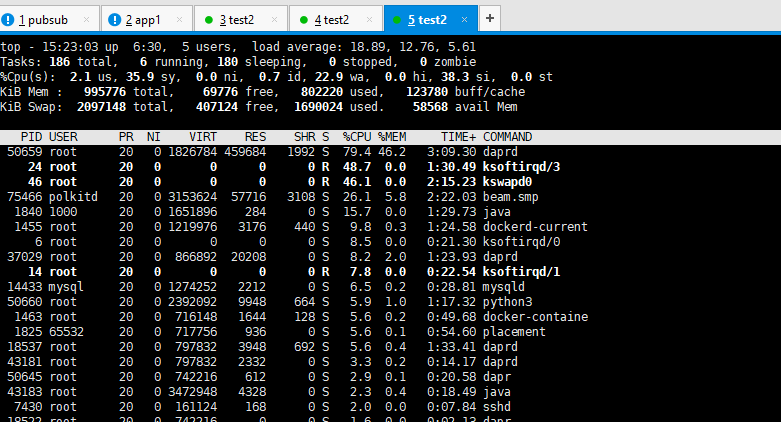
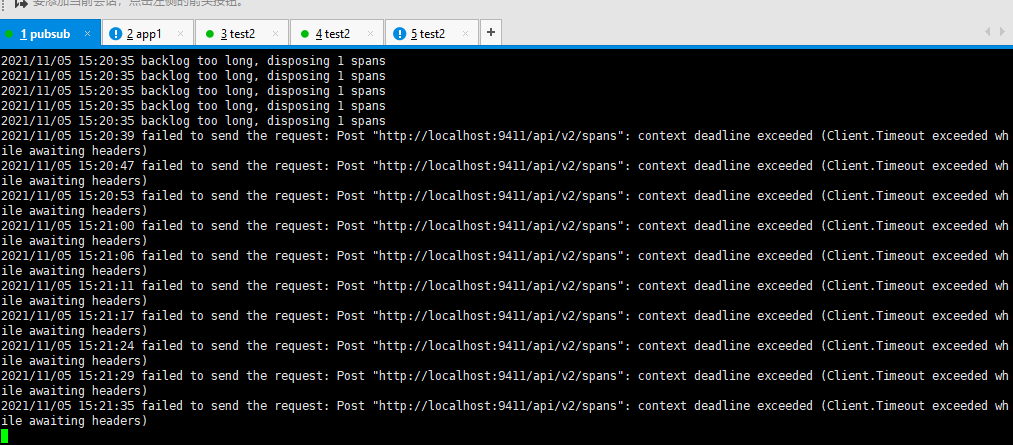
注意：代码运行添加防火墙端口：

-A INPUT -m state --state NEW -m tcp -p tcp --dport 3500 -j ACCEPT

在 talents-database-rabbitmq-pubsub.yaml 发布配置文件修改 deletedWhenUnused 为true,启动发布程序报异常。

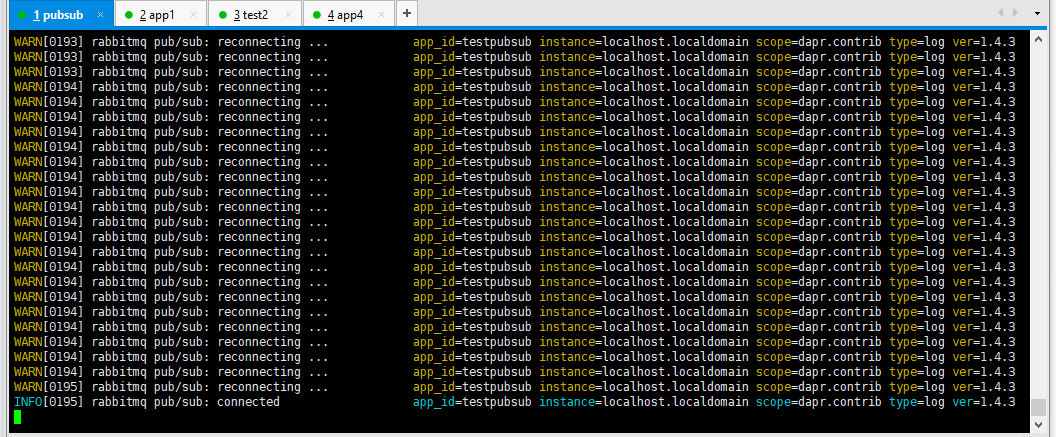
在 发布请求添加参数 metadata.ttlInSeconds 为生命周期

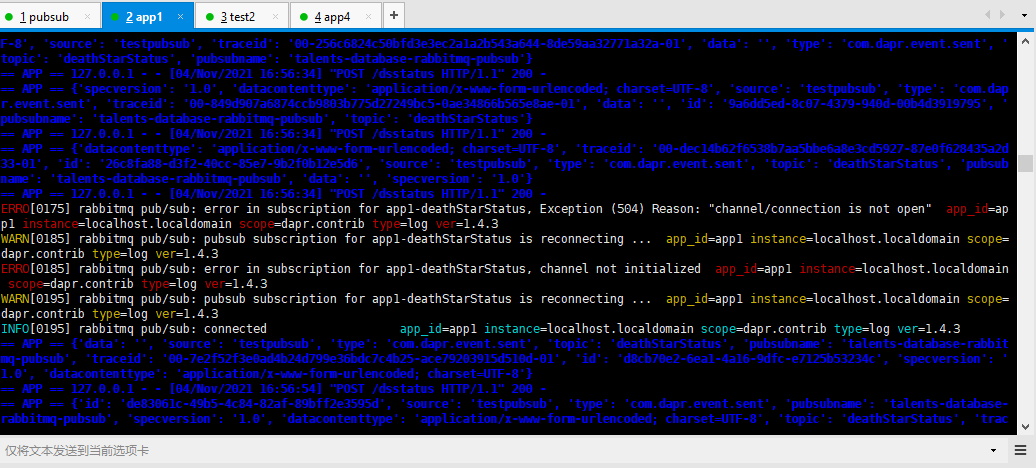
压力测试

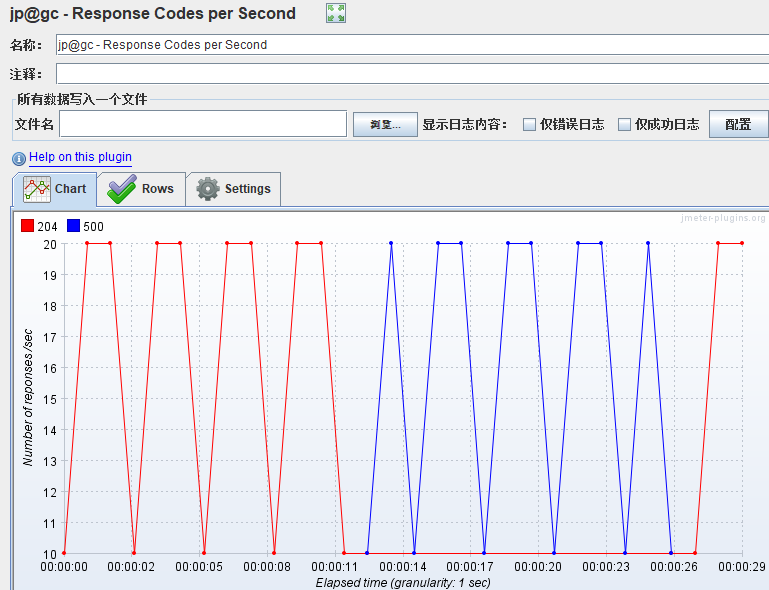
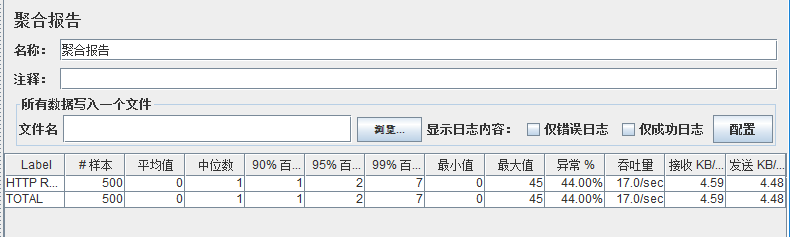


混沌测试（50个线程，30秒，循环10次）：

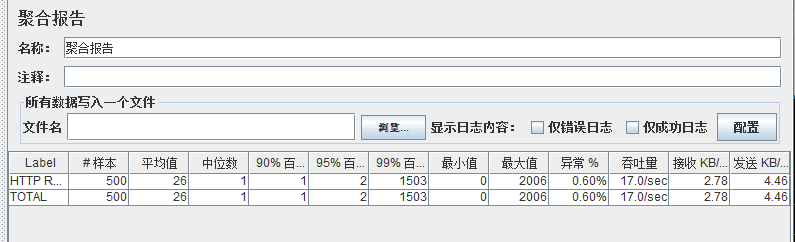
1、mq中断（发布立刻尝试重新连接，订阅出现错误提示），重启mq,发布连接成功，订阅正常接收。

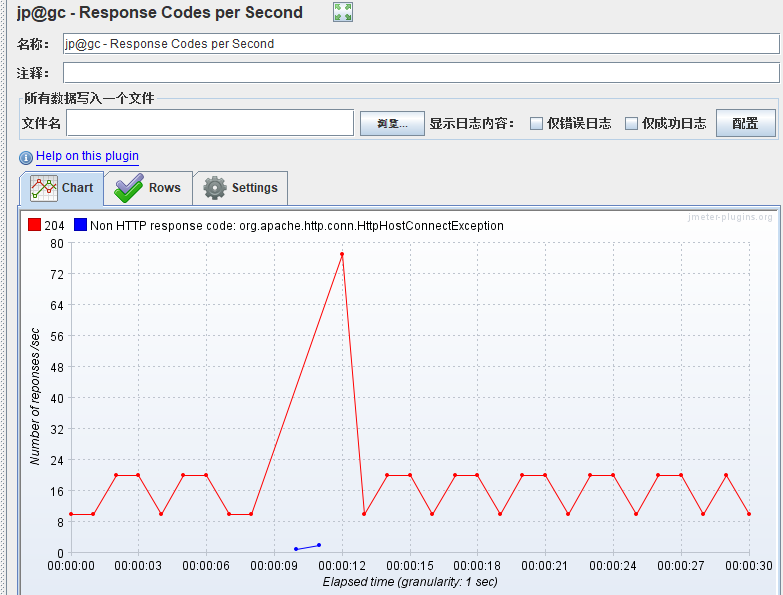




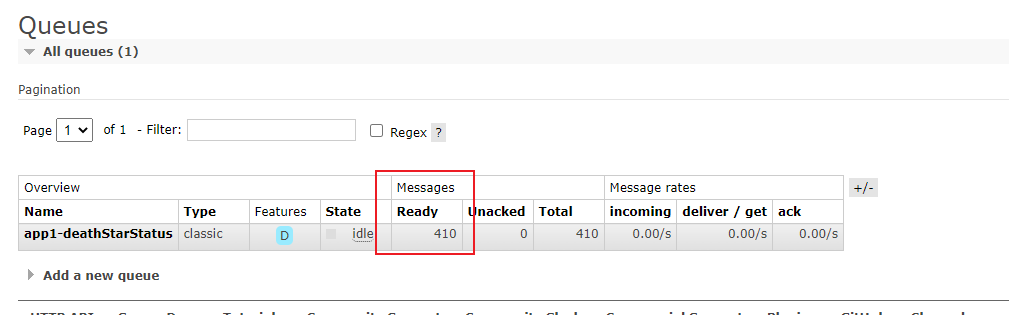
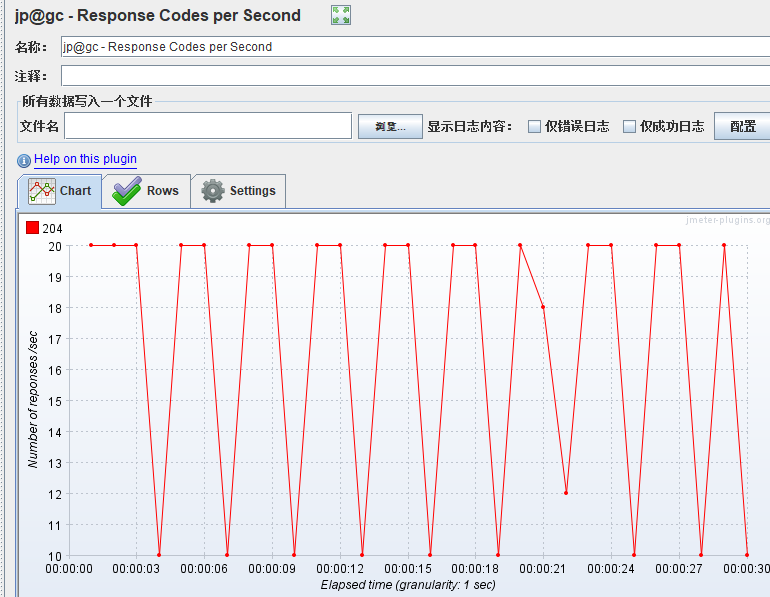
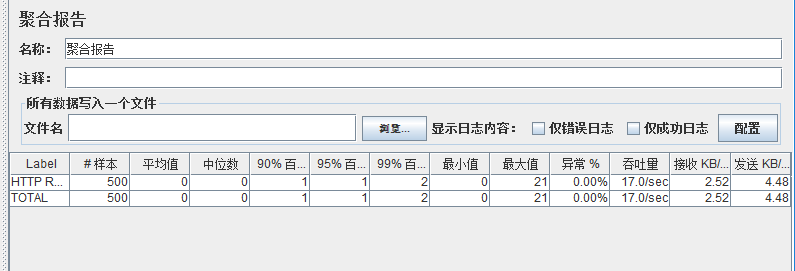


1. 发布中断（订阅停止接收发布消息，发布者重新启动，订阅可继续接收）





1. 订阅中断（订阅重启可继续接收）



容易出现下图卡死情况

