金蝶Apusic分布式消息队列（ADMQ）V2 命令行安装手册

深圳市金蝶天燕云计算股份有限公司

发布日期：2023年7月25日

文档版本：v1.0

目录

[目录 2](#_Toc143010369)

[第1章 总体说明 4](#_Toc143010370)

[第2章 前期准备工作 5](#_Toc143010371)

[第3章 部署管控台 6](#_Toc143010372)

[第4章 部署ADMQ核心引擎 8](#_Toc143010373)

[4.1 单机部署 8](#_Toc143010374)

[4.2 集群部署 9](#_Toc143010375)

[4.2.1 部署协调器 9](#_Toc143010376)

[4.2.2 初始化元数据 10](#_Toc143010377)

[4.2.3 存储组件 11](#_Toc143010378)

[4.2.4 计算组件 12](#_Toc143010379)

[4.3 伪集群部署 13](#_Toc143010380)

[4.3.1 创建目录并拷贝安装包 13](#_Toc143010381)

[4.3.2 修改节点配置 14](#_Toc143010382)

[4.3.3 拷贝安装包到其他节点目录并修改端口 15](#_Toc143010383)

[4.3.4 启动协调器并初始化集群 16](#_Toc143010384)

[4.3.5 启动存储节点和计算节点 17](#_Toc143010385)

[4.3.6 验证是否启动成功 17](#_Toc143010386)

[4.4 加载插件 17](#_Toc143010387)

[4.4.1 Kafka插件 18](#_Toc143010388)

[4.4.2 RocketMQ插件 19](#_Toc143010389)

[4.4.3 AMQP（RabbitMQ）插件 19](#_Toc143010390)

[4.4.4 MQTT插件 20](#_Toc143010391)

[第5章 部署RocketMQ引擎 21](#_Toc143010392)

[5.1 单机部署 21](#_Toc143010393)

[5.2 集群部署 21](#_Toc143010394)

[5.2.1 部署namesrv 21](#_Toc143010395)

[5.2.2 部署broker 22](#_Toc143010396)

[5.3 主从部署 25](#_Toc143010397)

[第6章 管控台注册引擎服务 26](#_Toc143010398)

[第7章 常见问题 27](#_Toc143010399)

[7.1 协调器启动失败 27](#_Toc143010400)

[7.1.1 服务器多IP 27](#_Toc143010401)

[7.2 rocketmq nameserver启动失败 27](#_Toc143010402)

[7.2.1 jdk版本过高 27](#_Toc143010403)

# 总体说明

ADMQ包括管控台和引擎两部分，通常手动部署管控台，然后通过管控台部署引擎。但在某些场景下需要单独部署引擎，然后通过其他方式管理或者注册到ADMQ管控台管理。

本文档给出命令行方式部署ADMQ引擎的流程，用户可依据此文档完成ADMQ单机和集群的部署。

# 前期准备工作

在部署实施之前，首先获取软件包，准备部署环境。

**软件环境**

|  |  |  |
| --- | --- | --- |
| **软件** | **作用** | **其他说明** |
| admq-manager-Vx.x.x | admq管控台，提供集群、主题等资源管理功能 |  |
| admq-Vx.x.x | admq核心引擎包，提供admq、rocketmq、kafka、rabbitmq、mqtt和jms等类型的消息收发服务。 |  |
| rocketmq-Vx.x.x | rocketmq安装包，只提供rocketmq服务 |  |
| jdk1.8及以上 | 软件运行依赖环境 |  |
| mysql 5.7.26及以上 | 提供管控台数据存储服务 | 可选。默认使用h2内存数据库，生产环境建议mysql、tdsql等数据库。 |

**硬件环境**

服务器

内存：2G及以上

硬盘：50G及以上

数量：单机1台，集群至少3台

网络：千兆以上网卡

# 部署管控台

管控台（admq-manager）是一个spring boot程序，可以直接解压运行。

**获取软件包后，解压并进入解压目录**

tar zxvf admq-manager-V2.4.1.tar.gz

cd admq-manager-V2.4.1

其中的目录及作用如下：

|  |  |  |
| --- | --- | --- |
| 目录 | **作用** | **其他说明** |
| bin | 启动、停止等脚本目录 |  |
| config | 配置文件目录，核心配置文件为application.properties |  |
| data | 数据存储目录，报错上传的软件包、部署配置等信息 |  |
| lib | 程序依赖的所有jar包 |  |
| licenses | 存放license文件 |  |
| logs | 存放程序运行日志文件 |  |
| package | 存放引擎服务安装包，程序第一次启动时会自动加载该目录中的安装包 |  |
| ui | 前端页面文件 |  |

**修改配置文件**

进入config目录，打开application.properties文件，修改服务端口和数据库地址等信息。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| server.port | 管控台访问端口，仅支持https访问 | 12306 |
| server.http.port | 管控台访问端口，仅支持http访问。建议不要使用此端口访问，容易造成安全问题。 | 12305 |
| config.dir | 配置文件目录 | ./config/ |
| spring.datasource.driver-class-name | 使用的驱动名称，默认使用h2。生产环境建议使用mysql、tdsql等 | org.h2.Driver |
| spring.datasource.url | 数据库链接地址 | jdbc:mysql://ip:port?useUnicode=true&characterEncoding=utf-8&allowMultiQueries=true&useSSL=false |
| spring.sql.init.schema-locations | 数据库初始化sql文件路径 | classpath:/META-INF/sql/h2-schema.sql |
| spring.datasource.username | 数据库用户名称 | sa |
| spring.datasource.password | 数据库用户密码。使用ENC则表示加密，否则不加密。 | ENC(79ZwI6lMamz8n3KxETrheBUA5uuWDwevUF6ZWIE1X4jXvJWQ+m0Mqb16JCWk9B6B) |
|  |  |  |

主要是修改数据库配置（测试环境可以不修改）。

**启动**

bin/admq-manager start

**检查状态**

浏览器访问<https://ip:12306>，初始用户名密码：admq/11111111

**停止**

bin/admq-manager stop

# 部署ADMQ核心引擎

核心引擎包含三个组件：计算组件（broker）、存储组件（storage）和协调器组件，单机部署时三个组件的功能集成在一个进程中，不需要单独部署。但是集群部署时，需要单独部署、启动三个组件。

计算组件提供客户端连接服务，接收到消息后存储到存储组件中。协调器则负责保存topic等元数据信息。

## 单机部署

**解压软件包并进入安装目录**

tar zxvf admq-V2.4.2-all.tar.gz

cd admq-V2.4.2

**修改配置文件**

进入conf目录，打开standalone.conf文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| webServicePort | 管理服务端口 | 8080 |
| deployServerAddress | 本机IP | 127.0.0.1 |
| advertisedAddress | 本机IP | 127.0.0.1 |
| clusterName | 集群名称 | apusic-mq-standalone |

**放置License文件**

把license文件拷贝到admq-licenses目录。

**启动**

bin/admq-daemon start standalone

**检查状态**

bin/admqctl admin clusters list

如果有结果则表示正常。

**停止**

bin/admq-daemon stop standalone

## 集群部署

组件部署顺序：协调器、存储组件、计算组件。

集群部署至少需要三台服务器，可以在每台服务器上都部署三种类型的组件。

### 部署协调器

协调器的内核是zookeeper，主要用于保存集群元数据信息。

**创建部署目录并拷贝安装包**

mkdir /admq/v2.4.2/zk

cp admq-V2.4.2-all.tar.gz /admq/v2.4.2/zk

**解压软件包并进入目录**

tar zxvf admq-V2.4.2-all.tar.gz

cd admq-V2.4.2

**创建数据存储目录并生成节点ID（myid文件）**

mkdir -p local\_zookeeper/data

mkdir -p local\_zookeeper/logs

echo 1 > local\_zookeeper/data/myid

注：三个节点的id不能相同，而且需要和配置文件中保持一致。一般设置为1、2、3

**修改配置文件**

进入conf目录，打开zookeeper.conf文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| clientPort | 服务端口 | 2181 |
| server.1 | 第一个节点的地址 | 192.168.0.1:2888:3888 |
| server.2 | 第二个节点的地址 | 192.168.0.2:2888:3888 |
| server.3 | 第三个节点的地址 | 192.168.0.3:2888:3888 |
| forceSync | 是否同步刷新，如果磁盘读写能力不好可设置成false，避免因此连接超时 | true |
| metricsProvider.httpPort | Prometheus监控端口 | 18800 |

**启动**

bin/admq-daemon start zookeeper

**停止**

bin/admq-daemon stop zookeeper

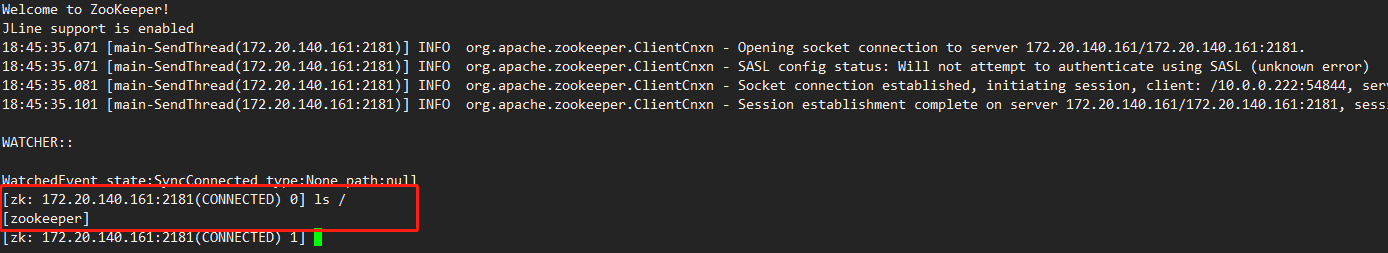
**修改另外两个节点的配置并启动**

只有myid的内容有区别，分别为1、2、3，其他都一样。

**检查状态**

bin/admqctl zk-cli -server ip1:2181

能连接则表示正常。



### 初始化元数据

需要把集群信息提前写入协调器中，执行以下命令：

bin/admqctl initialize-cluster-metadata \

--cluster apusic-mq \

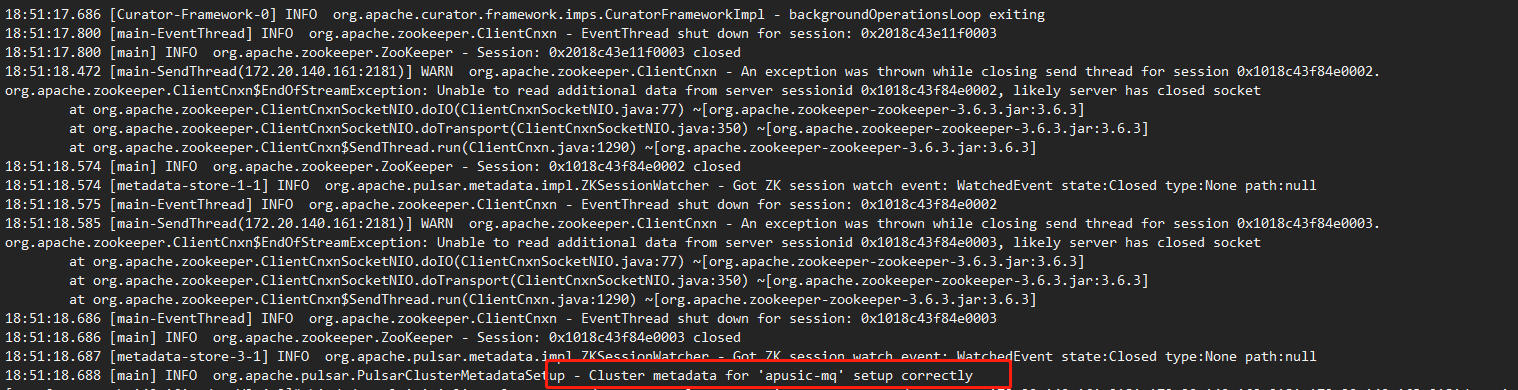
--metadata-store 172.20.140.161:2181,172.20.140.162:2181,172.20.140.163:2181/apusic-mq \

--configuration-metadata-store 172.20.140.161:2181,172.20.140.162:2181,172.20.140.163:2181/apusic-mq \

--web-service-url http://172.20.140.161:8080,172.20.140.162:8080,172.20.140.163:8080 \

--broker-service-url pulsar://172.20.140.161:6650,172.20.140.162:6650,172.20.140.163:6650

出现“Cluster metadata for 'apusic-mq' setup correctly”则表示执行成功



### 存储组件

存储组件主要用于保存集群数据。

**创建部署目录并拷贝安装包**

mkdir /admq/v2.4.2/storage

cp admq-V2.4.2-all.tar.gz /admq/v2.4.2/storage

**解压软件包并进入目录**

tar zxvf admq-V2.4.2-all.tar.gz

cd admq-V2.4.2

**放置License文件**

把license文件拷贝到admq-licenses目录。

**修改配置文件**

进入conf目录，打开storage.conf文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| bookiePort | 对外提供的服务端口 | 3181 |
| deployServerAddress | 本机IP | 127.0.0.1 |
| advertisedAddress | 本机IP | 127.0.0.1 |
| journalDirectories | 数据存储目录 | /admq\_journal |
| ledgerDirectories | 数据存储目录 | /admq\_data |
| metadataServiceUri | 协调器连接地址，其中的admq换成集群名称 | zk+hierarchical://192.168.0.1:2181;192.168.0.2:2181;192.168.0.3:2181/admq/ledgers |
| clusterName | 集群名称；需要和元数据初始化中的保持一致 | apusic-mq |
| httpServerPort | Prometheus监控服务端口 | 8000 |

需要创建journalDirectories和ledgerDirectories配置的目录，或者把上述配置改成已有目录。

**启动**

bin/admq-daemon start storage

**停止**

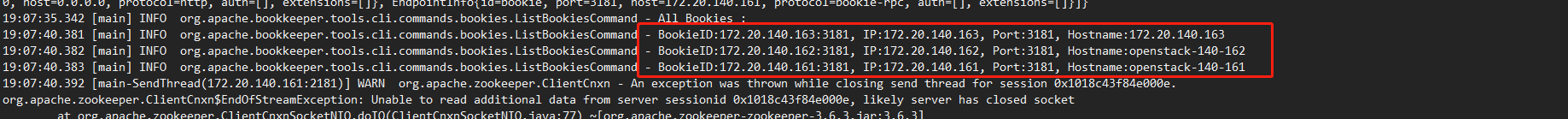
bin/admq-daemon stop storage

**修改另外两个节点的配置并启动**

只有IP地址有区别，其他都一样。

**检查状态**

bin/admqctl store-shell listbookies -a



可以查询到三个storage节点

### 计算组件

计算组件主要用于提供客户端连接服务。

**创建部署目录并拷贝安装包**

mkdir /admq/v2.4.2/broker

cp admq-V2.4.2-all.tar.gz /admq/v2.4.2/broker

**解压软件包并进入目录**

tar zxvf admq-V2.4.2-all.tar.gz

cd admq-V2.4.2

**放置License文件**

把license文件拷贝到admq-licenses目录。

**修改配置文件**

进入conf目录，打开broker.conf文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| metadataStoreUrl | 协调器地址；需要把地址换成实际地址，同时把admq换成集群名称（和元数据初始化时保持一致）。 | 192.168.0.1:2181,192.168.0.2:2181,192.168.0.3:2181/admq |
| webServicePort | 管理服务端口 | 8080 |
| deployServerAddress | 本机IP | 127.0.0.1 |
| advertisedAddress | 本机IP | 127.0.0.1 |
| clusterName | 集群名称 | apusic-mq-standalone |

**启动**

bin/admq-daemon start broker

**停止**

bin/admq-daemon stop broker

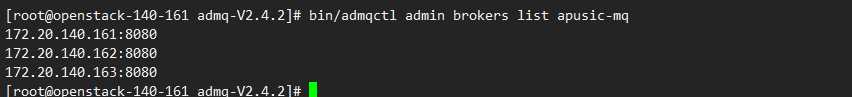
**修改另外两个节点的配置并启动**

只有IP地址有区别，其他都一样。

**检查状态**

bin/admqctl admin brokers list apusic-mq

其中apusic-mq是集群名称。



## 伪集群部署

在有些场景下需要在1台或者2台服务器上部署admq，这时候就会出现同一台服务器上多个相同的组件同时在运行。这种方式虽然也是集群部署，但可靠性相对于多台服务器部署的集群要低很多，因为服务器故障后会导致整个集群不可用。

下面针对单台服务器部署集群的方式进行说明。

### 创建目录并拷贝安装包

多服务器部署时，每一类进程在每台服务器上的目录基本都是相同的，但是单服务器部署集群时需要每个进程一个单独的目录。

部署一个三节点集群时，对于协调器、计算节点、存储节点分别创建对应的目录，如下：

mkdir -p node1/localZk

mkdir -p node2/localZk

mkdir -p node3/localZk

mkdir -p node1/broker

mkdir -p node2/broker

mkdir -p node3/broker

mkdir -p node1/storage

mkdir -p node2/storage

mkdir -p node3/storage

目录创建完成后解压安装包并把解压后的文件拷贝到node1目录中：

tar zxvf admq-V2.4.2-all.tar.gz

cp -r admq-V2.4.2 node1/localZk

cp -r admq-V2.4.2 node1/broker

cp -r admq-V2.4.2 node1/storage

### 修改节点配置

伪集群部署时大部分配置都是相同的，因此可以只修改node1下三个节点的配置，然后把配置拷贝到node2和node3目录中。

**修改协调器配置**

首先创建存储目录

mkdir -p node1/localZk/admq-V2.4.2/local\_zookeeper/data

mkdir -p node1/localZk/admq-V2.4.2/local\_zookeeper/logs

生成节点标识：myid文件

echo 1 > node1/localZk/admq-V2.4.2/local\_zookeeper/data/myid

修改配置文件，修改其中的ip和端口，具体修改项如下

vi node1/localZk/admq-V2.4.2/config/zookeeper.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **默认值** | **修改后的值** |
| server.1 | 192.168.0.1:2888:3888 | 主机IP:12888:13888 |
| server.2 | 192.168.0.2:2888:3888 | 主机IP:22888:23888 |
| server.3 | 192.168.0.3:2888:3888 | 主机IP:32888:33888 |
| clientPort | 2181 | 12181 |

**修改计算节点配置**

vi node1/broker/admq-V2.4.2/config/broker.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **默认值** | **修改后的值** |
| metadataStoreUrl | 192.168.0.1:2181,192.168.0.2:2181,192.168.0.3:2181/admq | 主机IP:12181, 主机IP:22181, 主机IP:32181/apusic-mq |
| brokerServicePort | 6650 | 16650 |
| webServicePort | 8080 | 18080 |
| advertisedAddress | 127.0.0.1 | 主机IP |

**修改存储节点配置**

vi node1/storage/admq-V2.4.2/config/storage.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **默认值** | **修改后的值** |
| advertisedAddress | 127.0.0.1 | 主机IP |
| bookiePort | 3181 | 13181 |
| journalDirectories | /admq\_journal | data/journal |
| ledgerDirectories | /admq\_data | data/data |
| metadataServiceUri | zk+hierarchical://192.168.0.1:2181;192.168.0.2:2181;192.168.0.3:2181/admq/ledgers | zk+hierarchical://主机IP:12181;主机IP:22181;主机IP:32181/apusic-mq/ledgers |
| httpServerPort | 8000 | 18000 |

### 拷贝安装包到其他节点目录并修改端口

可以把node1下三个组件所有配置拷贝到node2和node3下，然后再修改端口即可。

cp -r node1/\* node2/

cp -r node1/\* node3/

**修改端口等信息，解决冲突问题**

修改协调器端口

vi node2/localZk/admq-V2.4.2/config/zookeeper.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| clientPort | 12181 | 22181 |
| admin.serverPort | 19990 | 29990 |
| metricsProvider.httpPort | 18800 | 28800 |

vi node3/localZk/admq-V2.4.2/config/zookeeper.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| clientPort | 12181 | 32181 |
| admin.serverPort | 19990 | 39990 |
| metricsProvider.httpPort | 18800 | 38800 |

生成myid文件

echo 2 > node2/localZk/admq-V2.4.2/local\_zookeeper/data/myid

echo 3 > node3/localZk/admq-V2.4.2/local\_zookeeper/data/myid

修改计算节点端口

vi node2/broker/admq-V2.4.2/config/broker.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| brokerServicePort | 16650 | 26650 |
| webServicePort | 18080 | 28080 |

vi node3/broker/admq-V2.4.2/config/broker.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| brokerServicePort | 16650 | 36650 |
| webServicePort | 18080 | 38080 |

修改存储节点端口

vi node2/storage/admq-V2.4.2/config/storage.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| bookiePort | 13181 | 23181 |
| httpServerPort | 18000 | 28000 |

vi node3/storage/admq-V2.4.2/config/storage.conf

|  |  |  |
| --- | --- | --- |
| **配置名称** | **当前值** | **修改后的值** |
| bookiePort | 13181 | 33181 |
| httpServerPort | 18000 | 38000 |

### 启动协调器并初始化集群

分别进入三个协调器目录执行启动命令：

cd node1/localZk/admq-V2.4.2/ && bin/admq-daemon start zookeeper

cd node2/localZk/admq-V2.4.2/ && bin/admq-daemon start zookeeper

cd node3/localZk/admq-V2.4.2/ && bin/admq-daemon start zookeeper

初始化集群

cd node1/localZk/admq-V2.4.2

bin/admqctl initialize-cluster-metadata \

--cluster apusic-mq \

--zookeeper 主机IP:12181, 主机IP:22181, 主机IP:32181/apusic-mq \

--configuration-store 主机IP:12181, 主机IP:22181, 主机IP:32181/apusic-mq \

--web-service-url http://主机IP:18080,主机IP:28080,主机IP:38080 \

--broker-service-url pulsar://主机IP:16650,主机IP:26650,主机IP:36650

### 启动存储节点和计算节点

集群初始化完成后即可启动存储和计算节点，先启动存储在启动计算节点。

cd node1/storage/admq-V2.4.2/ && bin/admq-daemon start storage

cd node2/storage/admq-V2.4.2/ && bin/admq-daemon start storage

cd node3/storage/admq-V2.4.2/ && bin/admq-daemon start storage

cd node1/broker/admq-V2.4.2/ && bin/admq-daemon start broker

cd node2/broker/admq-V2.4.2/ && bin/admq-daemon start broker

cd node3/broker/admq-V2.4.2/ && bin/admq-daemon start broker

### 验证是否启动成功

执行以下命令可分别查看对应节点进程是否存在

ps -ef |grep QuorumPeerMain

ps -ef |grep MQStorage

ps -ef |grep MQBroker

随便进入一个安装目录，执行命令查看租户状态，如果正常输出则表示集群正常

cd node1/broker/admq-V2.4.2

bin/admqctl admin tenants list

## 加载插件

通过修改config/broker.conf文件加载不同的插件。通用配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| messagingProtocols | 需要开启的插件；支持：kafka,mqtt,amqp,rocketmq四种 |  |
| protocolHandlerDirectory | 插件目录 | ./protocols |
| narExtractionDirectory | 插件解压目录 | ./nar |

如果需要加载插件，首先要把上述配置加到standalone.conf或者broker.conf中。**添加前先检查配置项是否存在，如果存在则直接修改。**

加载插件后需要重启broker节点。

### Kafka插件

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| kafkaTenant | 数据保存租户 | kafka-data |
| kafkaMetadataTenant | 元数据保存租户 | kafka-meta |
| kafkaListeners | 监听地址 | kafka://127.0.0.1:9092 |
| kafkaProtocolMap | 协议转换地址 | kafka:PLAINTEXT |
| kafkaAdvertisedListeners | 对外暴露地址 | kafka://127.0.0.1:9092 |
| entryFormat | 数据转换类型 |  |
| brokerEntryMetadataInterceptors | 数据存储之前的拦截器 | 空 |
| allowAutoTopicCreationType | 自动创建的topic类型，需要设置成：partitioned | 空 |
| kopEnableGroupLevelConsumerMetrics | 是否开启订阅组监控 | true |
| saslAllowedMechanisms | 设置成PLAIN则开启sasl认证，设置成空则不开启 | 空 |

下面是一组示例配置：

kafkaTenant=kafka-data

kafkaMetadataTenant=kafka-meta

kafkaListeners=kafka://172.24.4.217:9092

kafkaProtocolMap=kafka:PLAINTEXT

kafkaAdvertisedListeners=kafka://172.24.4.217:9092

entryFormat=kafka

brokerEntryMetadataInterceptors=org.apache.pulsar.common.intercept.AppendIndexMetadataInterceptor

allowAutoTopicCreationType=partitioned

kopEnableGroupLevelConsumerMetrics=true

saslAllowedMechanisms=

### RocketMQ插件

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| rocketmqTenant | 数据保存租户 | rocketmq-data |
| rocketmqMetadataTenant | 元数据保存租户 | rocketmq-meta |
| rocketmqListeners | 监听地址 | rocketmq://127.0.0.1:9876 |
| rocketmqListenerPortMap | 端口转换 | 9876:rocketmq |
| loadManagerClassName | 数据加载管理类 | 空 |
| defaultRetentionTimeInMinutes | 数据保存时间 | 空 |
| defaultRetentionSizeInMB | 数据保存大小 | 空 |
| brokerEntryMetadataInterceptors | 数据存储前的拦截器 | 空 |
| allowAutoTopicCreationType | 自动创建topic类型；需要设置成partitioned | non-partitioned |
| ropBrokerReplicationNum | broker replication数量，需要设置成1 | 1 |
| ropTraceTopicEnable | 是否开启轨迹 | false |
| ropRestServerPort | 监控端口 | 9888 |
| ropAclEnable | 是否开启acl | false |
| advertisedListeners | 对外暴露地址 |  |

下面是一组示例配置：

rocketmqTenant=rocketmq-data

rocketmqMetadataTenant=rocketmq-meta

rocketmqListeners=rocketmq://172.20.140.140:9876

rocketmqListenerPortMap=9876:rocketmq

loadManagerClassName=org.apache.pulsar.broker.loadbalance.impl.ModularLoadManagerImpl

brokerEntryMetadataInterceptors=org.apache.pulsar.common.intercept.AppendIndexMetadataInterceptor

defaultRetentionTimeInMinutes=2400

defaultRetentionSizeInMB=16384

allowAutoTopicCreationType=partitioned

advertisedListeners=INTERNAL:pulsar://172.20.140.140:6650,rocketmq:pulsar://172.20.140.140:9876

ropBrokerReplicationNum=1

ropTraceTopicEnable=false

ropRestServerPort=9888

ropAclEnable=false

### AMQP（RabbitMQ）插件

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| amqpTenant | 数据保存租户 | amqp-data |
| amqpMetadataTenant | 元数据保存租户 | amqp-meta |
| amqpListeners | 监听地址 | amqp://127.0.0.1:5672 |
| amqpMaxNoOfChannels | 最大channel数量 | 2047 |
| amqpMaxFrameSize | 单个frame最大长度 | 4194304 |
| amqpProxyEnable | 是否开启代理 | true |
| amqpProxyPort | 代理服务端口 | 6672 |
| amqpAuthenticationEnabled | 是否开启认证 | false |
| amqpAuthorizationEnabled | 是否开启授权 | false |
| amqpAdminPort | 管理服务端口 | 15673 |
| loadBalancerAutoBundleSplitEnabled | 是否开启bundle自动分裂；需要设置成false | true |
| loadBalancerAutoUnloadSplitBundlesEnabled | 是否开启开启bundle自动卸载；需要设置成false | true |

下面是一组示例配置：

amqpTenant=amqp-data

amqpMetadataTenant=amqp-meta

amqpListeners=amqp://172.24.4.217:5672

amqpMaxNoOfChannels=2047

amqpMaxFrameSize=4194304

amqpProxyEnable=true

amqpProxyPort=6672

amqpAuthenticationEnabled=false

amqpAuthorizationEnabled=false

amqpAdminPort=15673

loadBalancerAutoBundleSplitEnabled=false

loadBalancerAutoUnloadSplitBundlesEnabled=false

### MQTT插件

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| mqttListeners | 监听地址 | mqtt://127.0.0.1:1883 |
| mqttProxyEnabled | 是否开启代理 | true |
| mqttProxyPort | 代理端口 | 5682 |

下面是一组示例配置：

mqttListeners=mqtt://172.20.140.140:1883

mqttProxyEnabled=true

mqttProxyPort=5682

# 部署RocketMQ引擎

## 单机部署

单机需要启动一个nameserver和一个broker，参考下面集群部署。

## 集群部署

集群部署建议至少需要启动2个nameserver，启动三个broker。

在每个broker中配置多个nameserver的地址。

### 部署namesrv

主要用于保存集群元数据信息。

**创建部署目录并拷贝安装包**

mkdir /admq/v4.9.2/nameserver

tar zxvf rocketmq-V4.9.2-all.tar.gz

cp rocketmq-V4.9.2-all.zip /admq/v4.9.2/nameserver

**解压软件包并进入目录**

unzip rocketmq-V4.9.2-all.zip

cd rocketmq-V4.9.2-all

**放置License文件**

mkdir admq-licenses

把license文件拷贝到admq-licenses目录。

**修改配置文件**

进入conf目录，打开namesrv.properties文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| listenPort | 监听端口 | 9876 |
| deployServerAddress | 本机IP | 127.0.0.1 |

**启动**

bin/rocketmq start nameserver

**停止**

bin/rocketmq stop nameserver

### 部署broker

主要用于保存集群元数据信息。

**创建部署目录并拷贝安装包**

mkdir /admq/v4.9.2/nameserver

tar zxvf rocketmq-V4.9.2-all.tar.gz

cp rocketmq-V4.9.2-all.zip /admq/v4.9.2/nameserver

**解压软件包并进入目录**

unzip rocketmq-V4.9.2-all.zip

cd rocketmq-V4.9.2-all

**放置License文件**

mkdir admq-licenses

把license文件拷贝到admq-licenses目录。

**修改配置文件**

进入conf目录，打开namesrv.properties文件。常用的配置如下：

|  |  |  |
| --- | --- | --- |
| **配置名称** | **作用** | **默认值** |
| listenPort | 监听端口 | 10911 |
| deployServerAddress | 本机IP | 127.0.0.1 |
| namesrvAddr | nameserver地址 | 空 |
| storePathRootDir | 数据保存目录 | 空 |
| storePathCommitLog | 数据保存目录 | 空 |
| brokerClusterName | 集群名称 | defaultCluster |
| aclEnable | 是否开启认证 | true |
| brokerName | 计算节点名称，主从模式下，属于同一个主节点的所有从节点，名字必须和主节点相同；  不同节点的名称不能相同，可以是broker-a，broker-b，broker-c | broker-a |
| brokerId | 主从模式下，各个计算节点的id，0表示主节点，非0表示从节点 |  |
| restServerPort | 监控端口 | 9888 |

下面是一组示例配置

deleteWhen=01

fileReservedTime=72

brokerRole=ASYNC\_MASTER

diskMaxUsedSpaceRatio=85

flushDiskType=ASYNC\_FLUSH

tenantRateLimit=true

sendRate=50000

receiveRate=150000

autoCreateTopicEnable=false

autoCreateSubscriptionGroup=false

aclEnable=true

traceTopicEnable=true

brokerClusterName=rocketmq

brokerName=broker-0

brokerId=0

restServerPort=9888

listenPort=10911

deployServerAddress=172.24.4.217

managerUrl=http://172.24.4.216:12307

tls.enable=false

tls.server.keyPath=conf/auth/server.key

tls.server.keyPassword=123456

tls.server.certPath=conf/auth/server.pem

tls.server.trustCertPath=conf/auth/ca.pem

tls.client.keyPath=conf/auth/client.key

tls.client.keyPassword=123456

tls.client.certPath=conf/auth/client.pem

tls.client.trustCertPath=conf/auth/ca.pem

namesrvAddr=172.24.4.217:9876

storePathRootDir=/admq/v2.4/data/rocketmq/rocketmq\_broker/1gX49DA8

storePathCommitLog=/admq/v2.4/data/rocketmq/rocketmq\_broker/1gX49DA8/commitlog

**配置ACL文件（可选）**

如果上边配置中开启了acl，则需要配置acl文件。修改conf/plan\_acl.yml文件，如下：

accounts:

- accessKey: eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJhZG1xIn0.ybJge7zTfy\_RDdAtB3w6nIPDHPT6-kbB6sNzgPt8sKQ

secretKey: admq123456

admin: true

globalWhiteRemoteAddresses:

- 172.20.140.161

- 172.20.140.162

- 172.20.140.163

最下面的白名单建议配置broker和nameserver的ip地址。

**启动**

bin/rocketmq start broker

**停止**

bin/rocketmq stop broker

## 主从部署

主从模式和集群模式类似，在启动主节点的前提下在部署从节点即可。

从节点的brokerName和主节点保持一致，brokerId从1递增。

配置完成后启动：

bin/rocketmq start broker

# 管控台注册引擎服务

引擎服务部署好后可以直接通过集群注册功能注册服务，在【集群管理】页面选择【非容器化注册】按照提示内容填写即可。

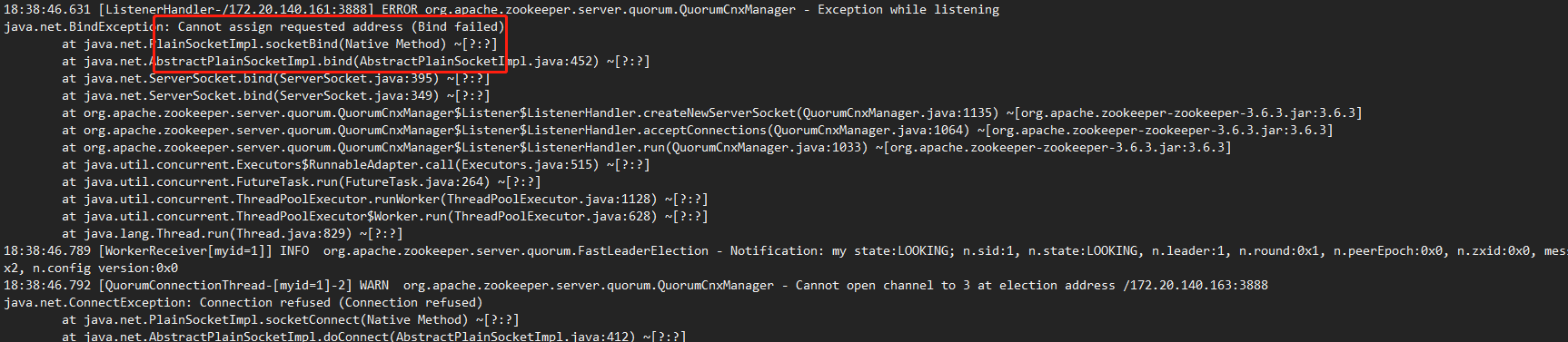
如果不想管理集群的启停和配置，则可以选择【容器化注册】。

# 常见问题

## 协调器启动失败

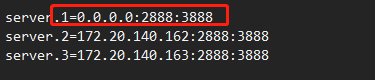
### 服务器多IP

查看logs/zookeeper—xx.log日志，出现以下日志则是该问题：



**解决办法**

修改conf/zookeeper.conf，把对应server的ip改成0.0.0.0，如下：



同理，在另外两台服务器上分别把对应id的ip改成0.0.0.0

## rocketmq nameserver启动失败

### jdk版本过高

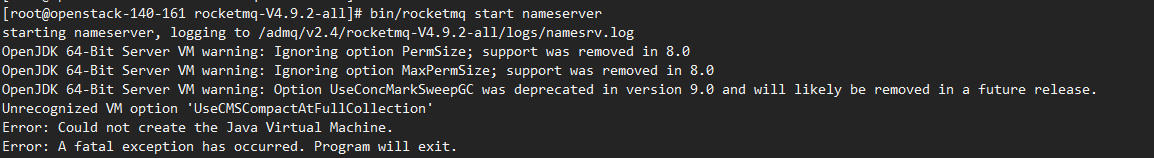
关键词：

Unrecognized VM option 'UseParNewGC'

Unrecognized VM option 'UseGCLogFileRotation'

Unrecognized VM option 'UseCMSCompactAtFullCollection'

如：



出现以上日志，则需要修改启动脚本：

bin/runserver.sh和bin/runbroker.sh中去掉上述不支持的jvm参数即可。