目录

目录 1
准备工作 2
注册金山云账号 2
    权限开通 2
    购买方式 2
创建集群 2
    前提条件 2
    操作步骤 2
公网访问 3
    基础环境 3
        绑定公网IP 3
        开放安全组端口 3
            示例：操作指南 3
修改 即可 4
填入 ACL 用户名密码 4
!/usr/bin/env python 4
-- coding: utf-8 -- 4
修改 即可, 根据业务自定义 5
填入 ACL 用户名密码 5
!/usr/bin/env python 5
-- coding: utf-8 -- 5

准备工作

本文介绍创建Kafka集群前所需的准备工作。

注册金山云账号

如果您还没有注册金山云的账号，请注册金山云账号,并完成实名认证

权限开通

1.请您先保证账号下至少存在一对密钥。

由于Kafka调用访问的需要，您至少需要创建一个Access Key，创建步骤如下：

（1）登录金山云官网。

（2）单击右上角的控制台。

（3）在控制台界面，鼠标指向右上角的用户身份，然后点击Access Keys

（4） 在AK密钥管理页面，点击**新建密钥**。

注：如出现以上提示，请单击继续使用。

1.请联系您的商务，开通KS3权限及关联购买权限模板。

2.用户在购买或扩容按小时实时付费KMR服务的时候，要保证金山云账户中至少有100元的现金。因此，在新购或扩容按小时实时付费KMR集群时，请确认您的账户中已至少充值100元，否则会创建失败。前往充值[充值中心](https://uc.console.ksyun.com/pro/buy/#/account/chargeChina)

购买方式

1、登录金山云控制台

2、左侧导航栏选择 **大数据** > **托管Hadoop**。

3、在托管Hadoop服务页面点击 **新建集群** 进入创建页面。

详情请参考创建集群

创建集群

本文介绍如何在KMR控制台，快速创建一个Kafka集群。

前提条件

注册金山云账号，并完成实名认证，开通相关权限，具体操作请参见准备工作

操作步骤

1、登录金山云控制台

2、左侧导航栏选择 **大数据** > **托管Hadoop**。

3、在托管Hadoop服务页面点击 新建集群 进入创建页面

4、基本信息配置

配置项	说明
业务场景	消息中间件
集群类型	Kafka
数据中心	数据中心是物理的数据中心的地理区域，不同地域的云产品之间内网不互通。一旦创建不能修改。
可用区	可用区是指金山云在同一地域内电力和网络互相独立的物理数据中心。选择Kafka集群所在可用区。
产品版本	选择创建Kafka集群所用的产品版本，目前支持Kafka2.2.2和2.8.1
计费方式	选择符合需求的计费方式，详情请参见计费方式。
集群名称	建集群时，会根据系统时间戳生成一个默认名称。您也可以为kafka集群输入自定义名称。长度限制为1-25个字符，支持数字、字母、特殊符号（_和-），该名称不必是唯一的。
所属项目组	您可以将集群加入项目，保持EBS、EIP、SLB、主机、KMR集群、项目一致，详情请参见项目管理。
SSH密钥	如果需要通过SSH访问集群，点击绑定密钥，详细请参阅SSH密钥管理

5、选择并配置节点组

6、配置网络及其他

配置项	说明
VPC网络	您可以使用同一数据中心内任一VPC来创建集群，若该数据中心无VPC，请您去VPC控制台创建，同时创建该VPC下EndPoint子网及与集群同一可用区内VPC子网
VPC子网	VPC子网是VPC中用于管理云主机的网络单元，您可以使用该VPC下与集群同一可用区内任一VPC子网来创建集群
所属安全组	您可直接新建一个安全组，如果已有KMR安全组，则可以直接选择使用
标签	您可以使用已有的标签进行分类，也可以选择创建新的标签进行分类

7、确认配置并购买

该页面会显示您所创建集群的配置清单，以及集群费用。根据不同的资源及计费信息，展示不同的价格信息。当所有信息确认正确有效之后，点击**购买**即可创建集群。

公网访问

本文介绍使用SDK完成生产、消费

基础环境

绑定公网IP

将集群所有Broker分别绑定弹性IP，具体操作是在云服务控制台绑定弹性IP，在云服务器控制台根据主机名搜索所有节点，设置方法详见更换IP。

开放安全组端口

选择创建集群时所选的安全组，并在该安全组内配置外网访问入站的ACL。开放对应客户端外网IP的访问，端口为9092

示例：操作指南

1.在客户端主机上配置/etc/hosts文件 需要将master1上的/etc/hosts文件内容添加至客户端主机上，并将每个ip替换成公网IP地址。

```
# kafka
120.92.33.214 kmr-0ba43848-gn-73607e20-master-1-001.ksc.com kmr-0ba43848-gn-73607e20-master-1-001
120.92.93.108 kmr-0ba43848-gn-73607e20-core-1-001.ksc.com kmr-0ba43848-gn-73607e20-core-1-001
120.131.13.117 kmr-0ba43848-gn-73607e20-core-1-002.ksc.com kmr-0ba43848-gn-73607e20-core-1-002
```

2.添加maven依赖。

```
// 消息队列 Kafka 版服务端版本为 0.10.1.0，建议的客户端版本为 0.10.2.2
<dependency>
```

```
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-clients</artifactId>
<version>0.10.2.2</version>
</dependency>
```

3.客户端添加配置文件。

```
##配置节点列表，即控制台中集群详情页中，节点配置项所有节点的内网ip列表，如120.92.33.214:6667,120.92.93.108:6667,120.131.13.117:6667
bootstrap.servers=xxxxxxxxxxxxxxxxxxxxx
##配置 Topic，可以在Kafka
topic=alikafka-topic-demo
##配置 Consumer Group
group.id=group-demo
```

加载配置文件：

```
public class JavaKafkaConfigurer {
    private static Properties properties;
    public synchronized static Properties getKafkaProperties() {
       if (null != properties) {
           return properties;
       }
       //获取配置文件 kafka.properties 的内容
       Properties kafkaProperties = new Properties();
       try {
           kafkaProperties.load(KafkaProducerDemo.class.getClassLoader().getResourceAsStream("kafka.properties"));
       } catch (Exception e) {
           // 没加载到文件，程序要考虑退出
           e.printStackTrace();
       }
       properties = kafkaProperties;
       return kafkaProperties;
    }
}
```

4. 使用Python SDK生产消息

```
# 修改 <ip> <port> <topic_name> 即可
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from kafka import KafkaProducer
import json
producer = KafkaProducer(
    bootstrap_servers = ['<ip>:<port>']) #多个ip以逗号隔开

message = "Hello kafka! Message test!"
msg = json.dumps(message).encode()
producer.send('<topic_name>',value = msg)
print("produce message " + message + " success.");
producer.close()
```

5. 使用Python SDK消费消息。

```
# 修改 <ip> <port> <topic_name> <group_id>  即可, <group_id> 根据业务自定义
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from kafka import KafkaConsumer

consumer = KafkaConsumer( '', group_id = "", bootstrap_servers = [':']) #多个ip以逗号隔开

for message in consumer: print ("Topic:[%s] Partition:[%d] Offset:[%d] Value:[%s]" % (message.topic,
message.partition, message.offset, message.value))
```

6. 开启ACL使用Python SDK生产消息

```
# 修改 即可

# 填入 ACL 用户名密码

!/usr/bin/env python

-- coding: utf-8 --
```

```
from kafka import KafkaProducer import json producer = KafkaProducer( bootstrap_servers = [':'], #多个ip以逗号隔开 security_protocol = "SASL_PLAINTEXT", sasl_mechanism = "SCRAM-SHA-256", sasl_plain_username = "", sasl_plain_password = "", )

message = "Hello kafka! Message test!" msg = json.dumps(message).encode() producer.send('',value = msg) print("produce message " + message + " success."); producer.close()
```

7. 开启ACL使用Python SDK消费消息

```
# 修改 即可，根据业务自定义

# 填入 ACL 用户名密码

!/usr/bin/env python

-- coding: utf-8 --

from kafka import KafkaConsumer

consumer = KafkaConsumer( '', group_id = "", bootstrap_servers = [':'], #多个ip以逗号隔开 security_protocol = "SASL_PLAINTEXT", sasl_mechanism = "SCRAM-SHA-256", sasl_plain_username = "", sasl_plain_password = "", )

for message in consumer: print ("Topic:[%s] Partition:[%d] Offset:[%d] Value:[%s]" % (message.topic, message.partition, message.offset, message.value))
```