实践指导

1. 下载 Kafka，并上传至 ECS

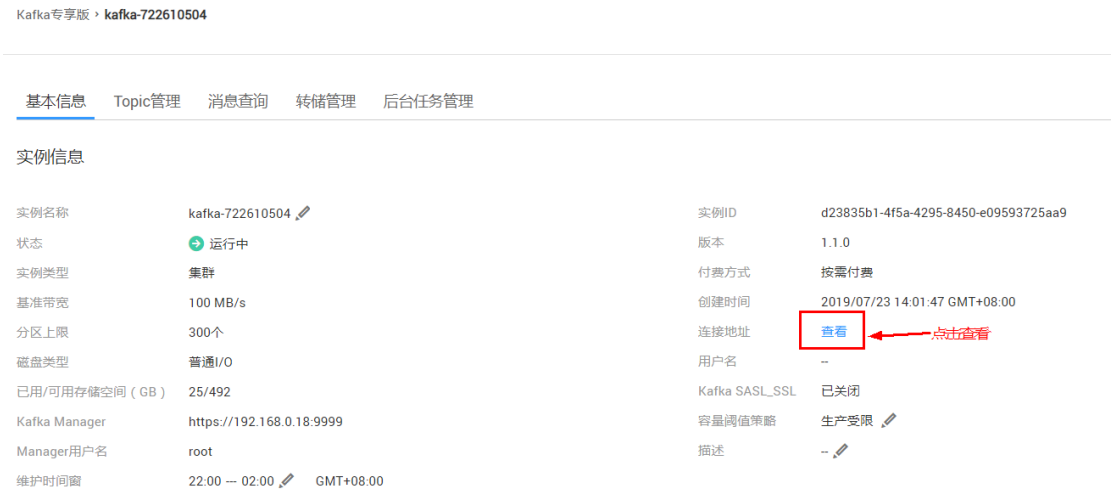
下载链接：https://archive.apache.org/dist/kafka/1.1.0/kafka_2.12-1.1.0.tgz

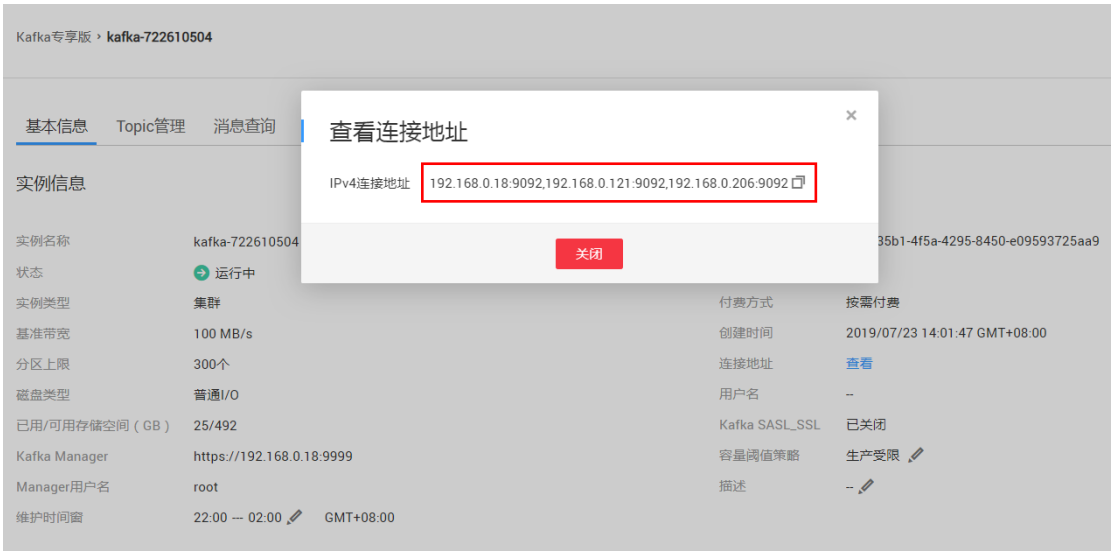
上传至ECS并解压：

```
[root@ecs-site-notdelete kafka_2.12-1.1.0]# pwd
/root/kafka_2.12-1.1.0
[root@ecs-site-notdelete kafka_2.12-1.1.0]# ll
total 52
drwxr-xr-x 3 root root  4096 Mar 24  2018 bin
drwxr-xr-x 2 root root  4096 Mar 24  2018 config
drwxr-xr-x 2 root root  4096 Jul 23 14:12 libs
-rw-r--r-- 1 root root 28824 Mar 24  2018 LICENSE
-rw-r--r-- 1 root root   336 Mar 24  2018 NOTICE
drwxr-xr-x 2 root root  4096 Mar 24  2018 site-docs
```

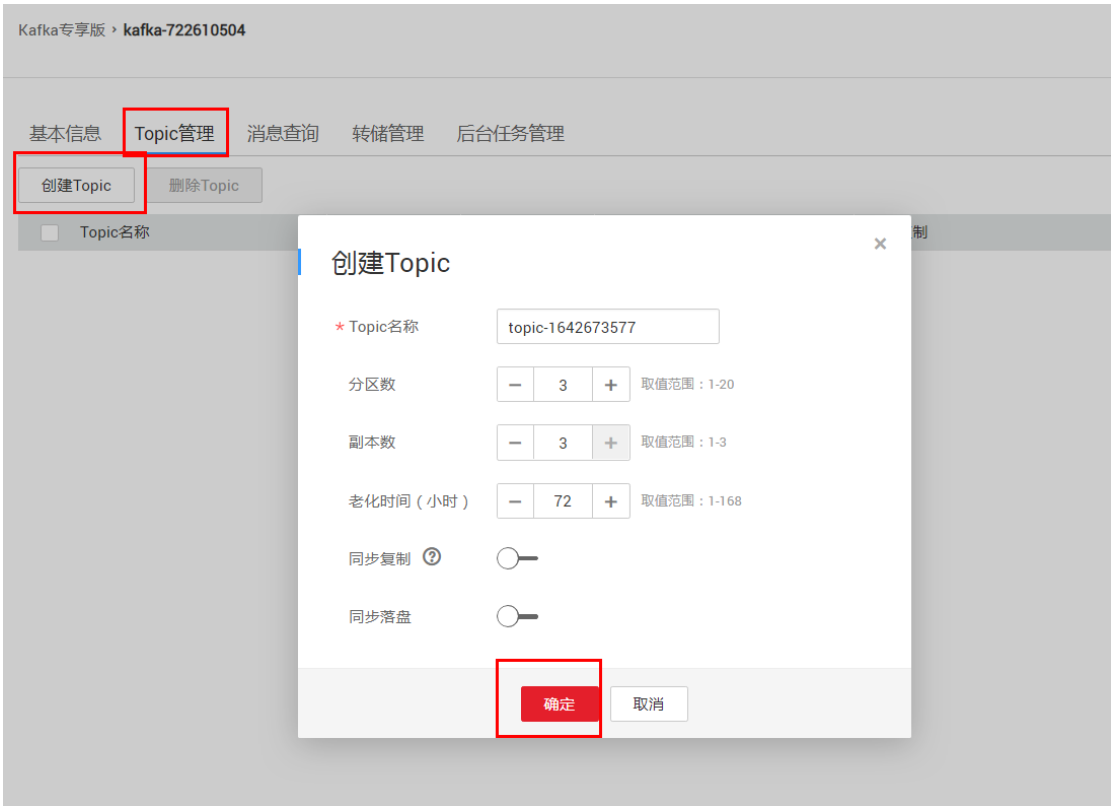
ECS要和Kafka实例在同一VPC

2. 在 Kafka 实例详情页获取 Kafka broker 地址：


Kafka专享版 › kafka-722610504
基本信息
Topic管理
消息查询
转储管理
后台任务管理
实例信息
实例名称
kafka-722610504
状态
运行中
实例类型
集群
基准带宽
100 MB/s
分区上限
300个
磁盘类型
普通I/O
已用/可用存储空间（GB）
25/492
Kafka Manager
https://192.168.0.18:9999
Manager用户名
root
维护时间窗
22:00 --- 02:00
GMT+08:00
实例ID
d23835b1-4f5a-4295-8450-e09593725aa9
版本
1.1.0
付费方式
按需付费
创建时间
2019/07/23 14:01:47 GMT+08:00
连接地址
查看
点击查看
用户名
--
Kafka SASL_SSL
已关闭
容量阈值策略
生产受限
描述
--


Kafka专享版 › kafka-722610504
基本信息
Topic管理
消息查询
实例信息
查看连接地址
IPv4连接地址
192.168.0.18:9092,192.168.0.121:9092,192.168.0.206:9092
关闭
实例名称
kafka-722610504
状态
运行中
实例类型
集群
基准带宽
100 MB/s
分区上限
300个
磁盘类型
普通I/O
已用/可用存储空间（GB）
25/492
Kafka Manager
https://192.168.0.18:9999
Manager用户名
root
维护时间窗
22:00 --- 02:00
GMT+08:00
付费方式
按需付费
创建时间
2019/07/23 14:01:47 GMT+08:00
连接地址
查看
用户名
--
Kafka SASL_SSL
已关闭
容量阈值策略
生产受限
描述
--

3. 创建 topic：



4. 将代码打包，上传至 Kafka libs 目录

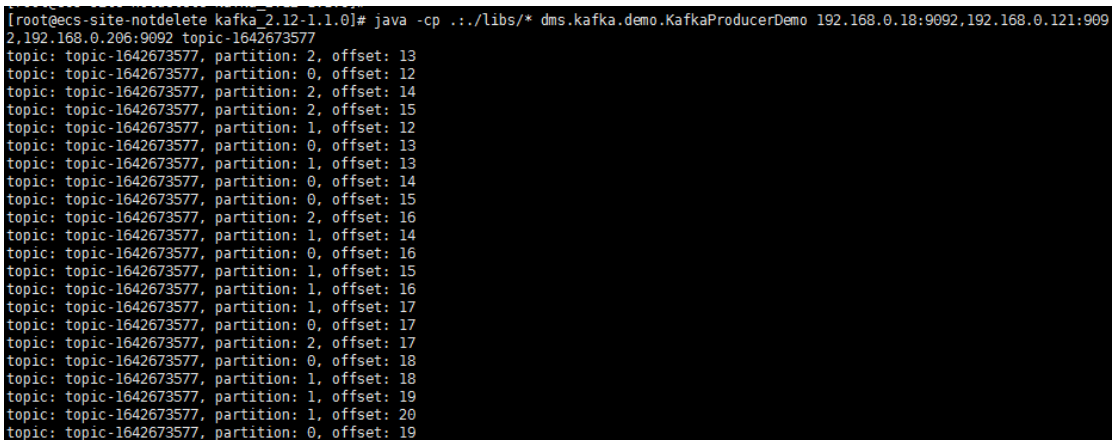
```
[root@ecs-site-notdelete libs]# pwd
/root/kafka_2.12-1.1.0/libs
[root@ecs-site-notdelete libs]# ll
total 48972
-rw-r--r-- 1 root root    14768 Aug 14  2017 aopalliance-repackaged-2.5.0-b32.jar
-rw-r--r-- 1 root root    90347 Aug 14  2017 argparse4j-0.7.0.jar
-rw-r--r-- 1 root root   479881 Aug 14  2017 commons-lang3-3.5.jar
-rw-r--r-- 1 root root    89496 Mar 24  2018 connect-api-1.1.0.jar
-rw-r--r-- 1 root root    19542 Mar 24  2018 connect-file-1.1.0.jar
-rw-r--r-- 1 root root    44928 Mar 24  2018 connect-json-1.1.0.jar
-rw-r--r-- 1 root root   408225 Mar 24  2018 connect-runtime-1.1.0.jar
-rw-r--r-- 1 root root    88980 Mar 24  2018 connect-transforms-1.1.0.jar
-rw------- 1 root root     8144 Jul 23 14:39 dms.kafka.demo.jar
-rw-r--r-- 1 root root  2442625 Aug 14  2017 guava-20.0.jar
-rw-r--r-- 1 root root   185793 Aug 14  2017 hk2-api-2.5.0-b32.jar
-rw-r--r-- 1 root root   187274 Aug 14  2017 hk2-locator-2.5.0-b32.jar
-rw-r--r-- 1 root root   134908 Aug 14  2017 hk2-utils-2.5.0-b32.jar
```

（1）生产消息：

java -cp .:./libs/* dms.kafka.demo.KafkaProducerDemo

192.168.0.18:9092,192.168.0.121:9092,192.168.0.206:9092 topic-1642673577

（需要保存此截图，并上传至打卡程序）

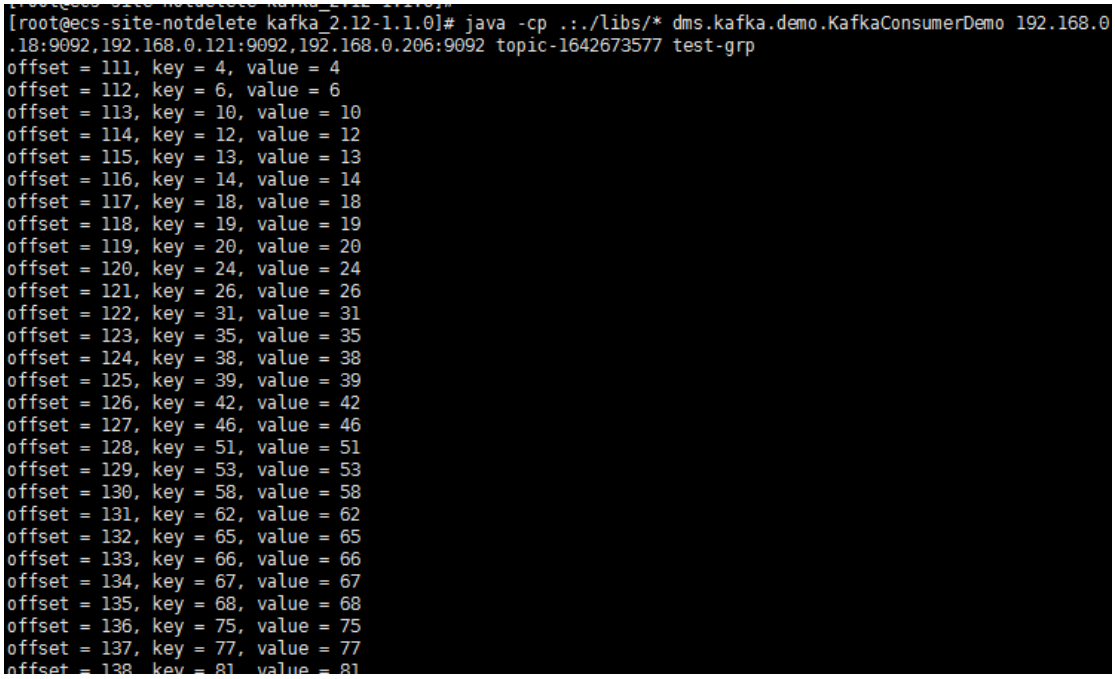


（2）消费消息：

java -cp .:./libs/* dms.kafka.demo.KafkaConsumerDemo

192.168.0.18:9092,192.168.0.121:9092,192.168.0.206:9092 topic-1642673577 test-grp

（需要保存此截图，并上传至打卡程序）



5. 消费代码 demo

```
package dms.kafka.demo;

import java.util.Arrays;
import java.util.Properties;

import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
```

```
public class KafkaConsumerDemo
{
    public static void main(String[] args)
    {
        if (args.length != 3)
        {
            throw new IllegalArgumentException("usage:
dms.kafka.demo.KafkaProducerDemo bootstrap-servers topic-name group-
name.");
        }

        Properties props = new Properties();
        props.put("bootstrap.servers", args[0]);
        props.put("group.id", args[2]);
        props.put("enable.auto.commit", "true");
        props.put("auto.offset.reset", "earliest");
        props.put("auto.commit.interval.ms", "1000");
        props.put("key.deserializer",
"org.apache.kafka.common.serialization.StringDeserializer");
        props.put("value.deserializer",
"org.apache.kafka.common.serialization.StringDeserializer");
        KafkaConsumer<String, String> consumer = new
KafkaConsumer<>(props);
        consumer.subscribe(Arrays.asList(args[1]));
        while (true)
        {
            ConsumerRecords<String, String> records = consumer.poll(200);
            for (ConsumerRecord<String, String> record : records)
                System.out.printf("offset = %d, key = %s, value = %s%n",
record.offset(), record.key(), record.value());
        }
    }
}
```