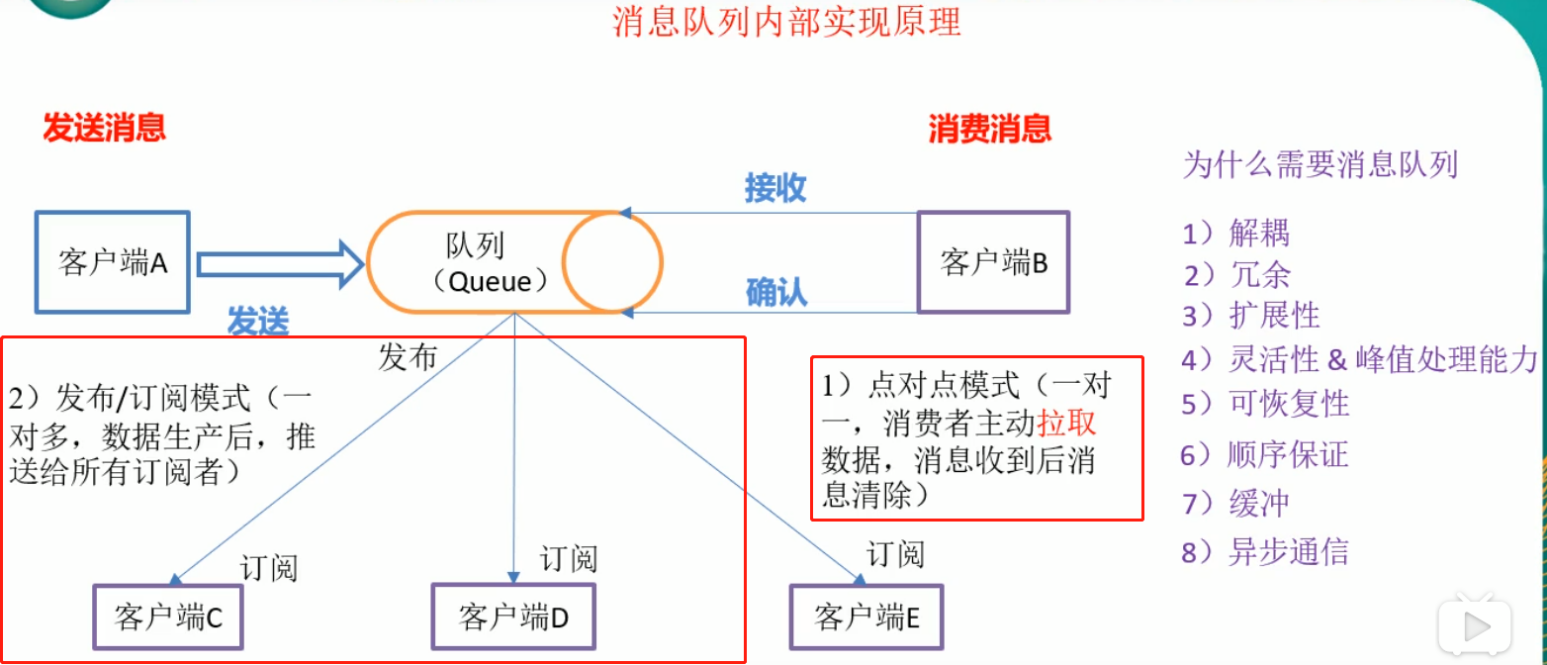
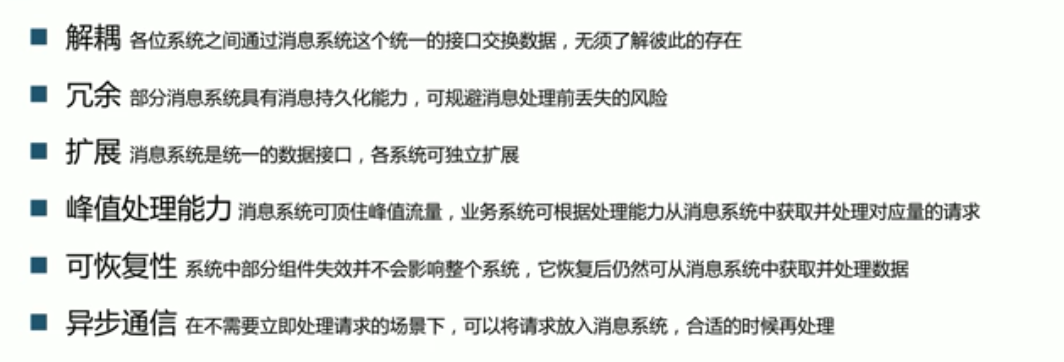
# 消息队列

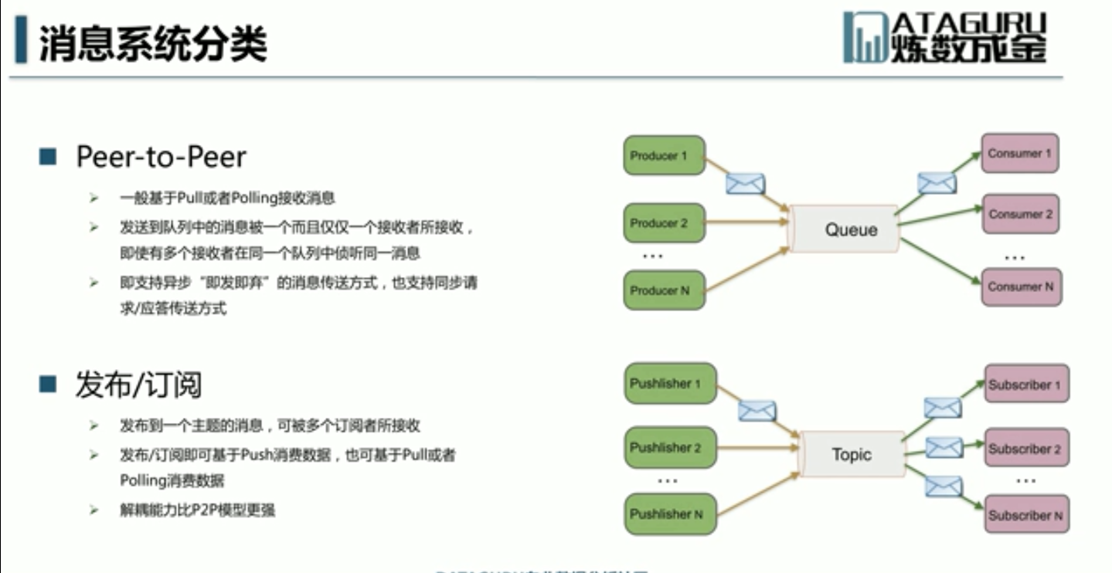


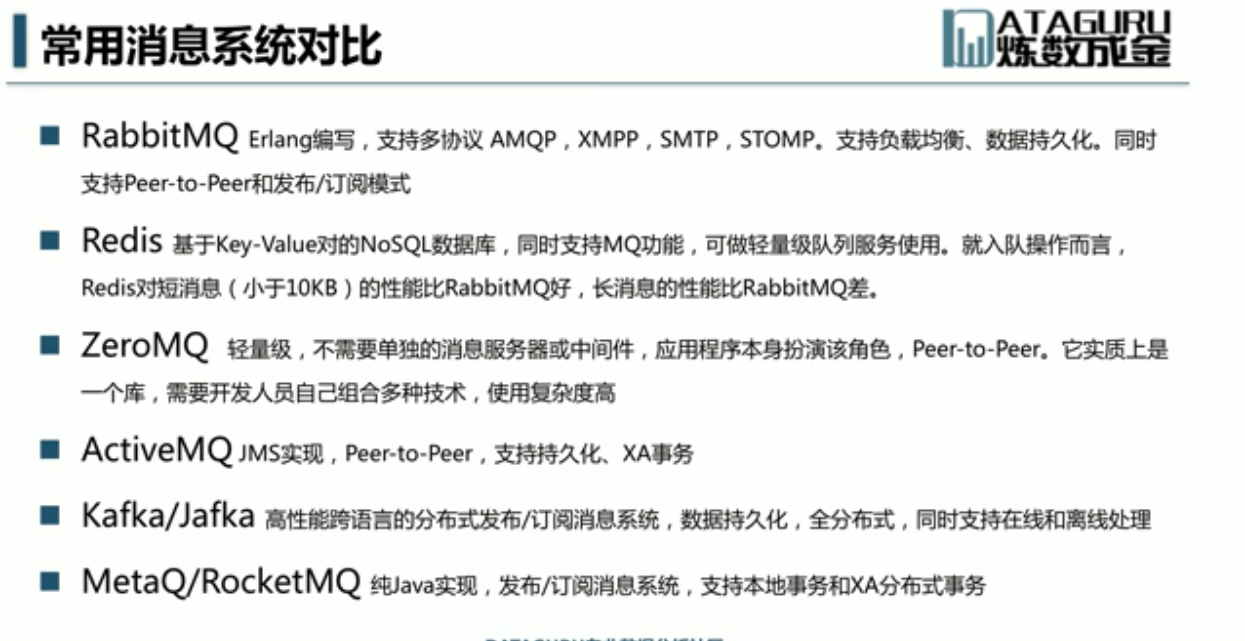


消息队列两种消费消息方式：

1点对点,kafka只支持这种

2发布/订阅





Kafka：jafka是kafka的java实现。

# Kafka简介：

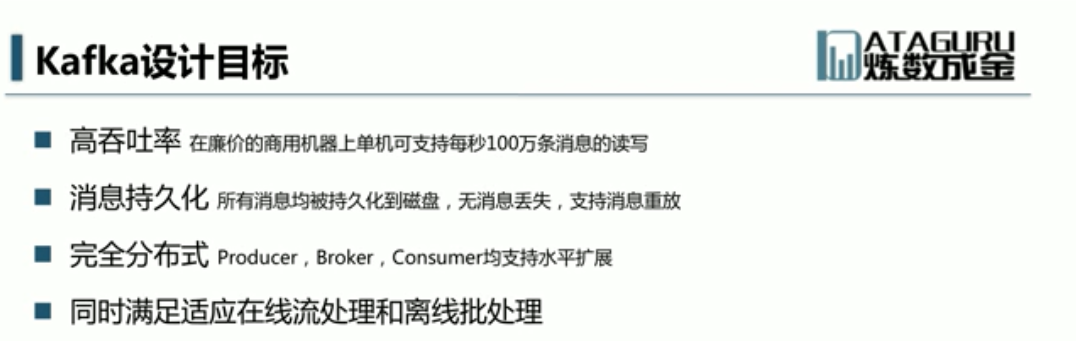
在流式计算中，Kafka一般用来缓存数据，storm通过消费Kafka的数据进行计算。

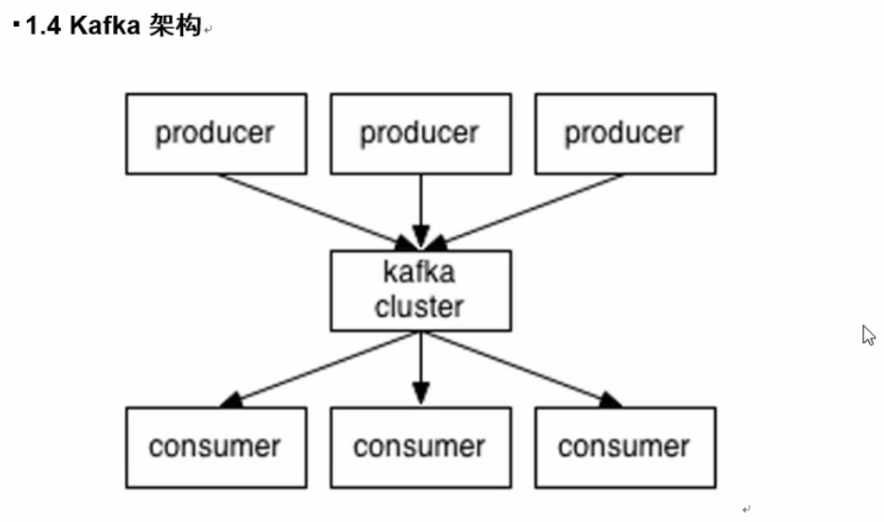
1、分布式消息队列，集群，由Scala写成，由apache软件基金会开发的开源消息系统，

2、Kafka对消息保存时根据topic进行归类，

3、发送消息者成为producer（生产者）,消息接受者称为Consumer（消费者），此外kafka有多个kafka实例组成，每个实例server称为broker

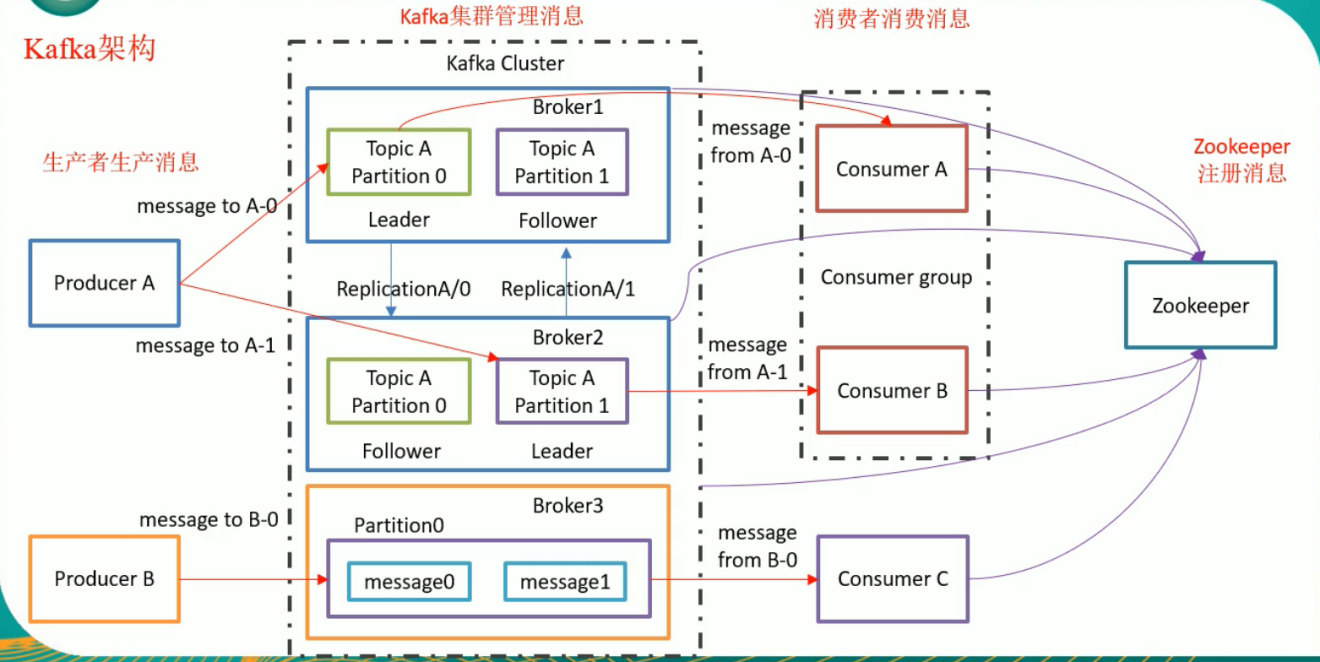
4、无论是kafka集群，还是consumer都依赖于zookeeper集群保存一些meta信息，来保证系统可用性。





* **Kafka cluster:** kafka由多个kafka实例组成，
* **Broker：**每个实例server称为broker
* **Producer：**消息生产者， Producer将消息发布到指定的Topic中,同时Producer也能决定将此消息归属于哪个partition;比如基于"round-robin"(循环，轮循)方式或者通过其他的一些算法等.
* **Consumer:**消息消费者、订阅者，本质上kafka只支持Topic.每个consumer属于一个consumer group;反过来说,每个group中可以有多个consumer.发送到Topic的消息,只会被订阅此Topic的每个group中的一个consumer消费. 任何发布到此partition的消息都会被直接追加到log文件的尾部，每条消息在文件中的位置称为offset（偏移量），offset为一个long型数字，它是唯一标记一条消息。它唯一的标记一条消息。kafka并没有提供其他额外的索引机制来存储offset，因为在kafka中几乎不允许对消息进行“随机读写”。
* **Topic（话题）:** 一个Topic可以认为是一类消息，每个topic将被分成多个partition(区),每个partition在存储层面是append log文件。
* Partition(分区:做负载均衡)：partitions的设计目的有多个.最根本原因是kafka基于文件存储.通过分区,可以将日志内容分散到多个server上,来避免文件尺寸达到单机磁盘的上限,每个partiton都会被当前server(kafka实例)保存;可以将一个topic切分多任意多个partitions,来消息保存/消费的效率.此外越多的partitions意味着可以容纳更多的consumer,有效提升并发消费的能力.(具体原理参见下文).
* Leader(领导者)：存取数据，消费数据都是找Leader，Leader挂了，follower可以变为leader
* Follower(追随者)：是leader的备份
* Guarantees
  + 发送到partitions中的消息将会按照它接收的顺序追加到日志中
  + 对于消费者而言,它们消费消息的顺序和日志中消息顺序一致.
  + 如果Topic的"replicationfactor"为N,那么允许N-1个kafka实例失效.

同一消费组不能同时消费同一分区，

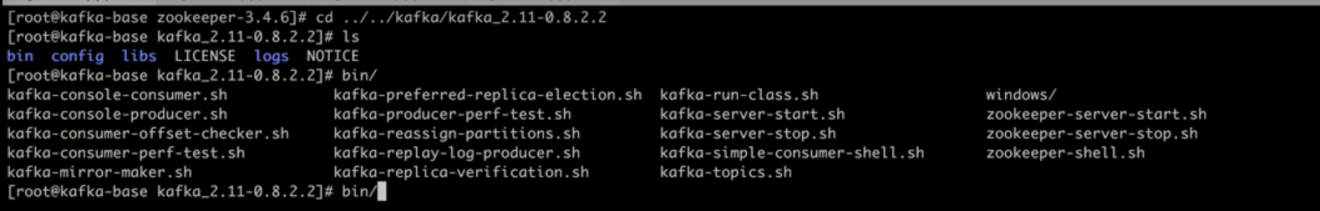


## Kafka安装----使用kafka自带的zookeeper

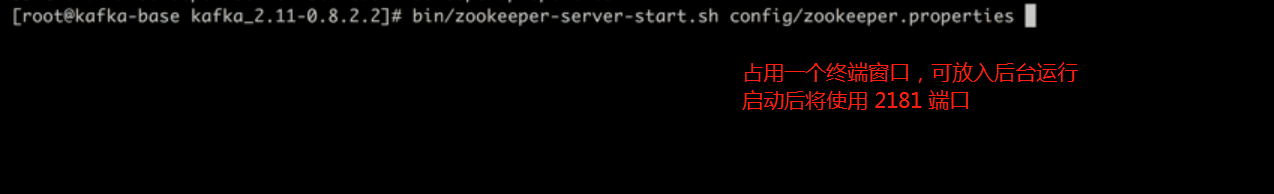
解压安装包如下目录：

bin目录中有许多脚本，启动zookeeper、kafka等

config目录中有许多配置文件，如zookeeper和kafka的配置文件，启动时需要指定相关配置文件



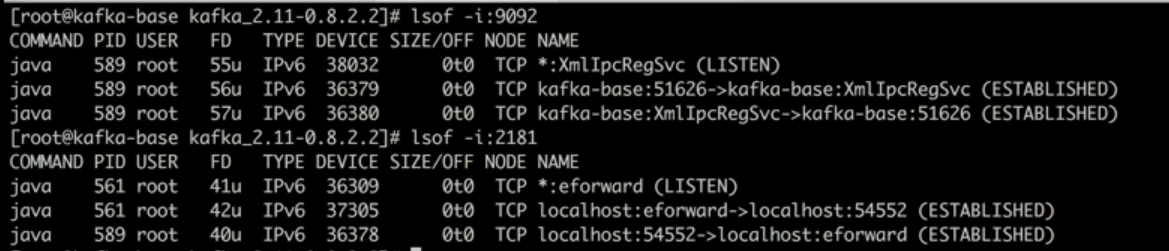
启动自带的zookeeper



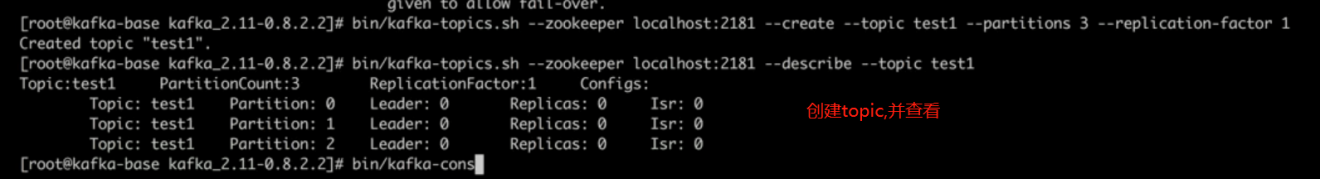
启动kafka的broker

端口默认9092

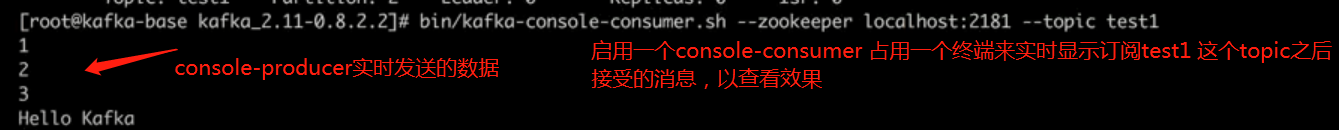




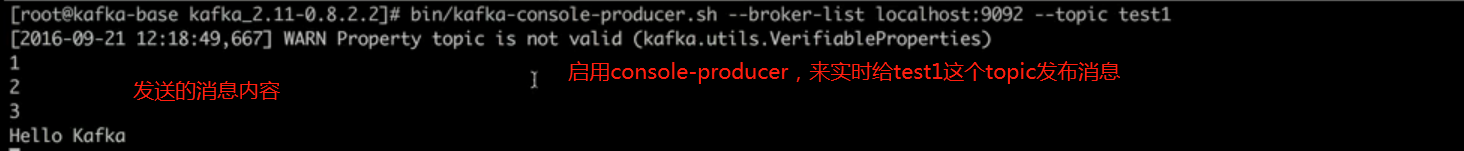
创建topic(话题)：



启用console-consumer订阅这个topic消息，并查看效果

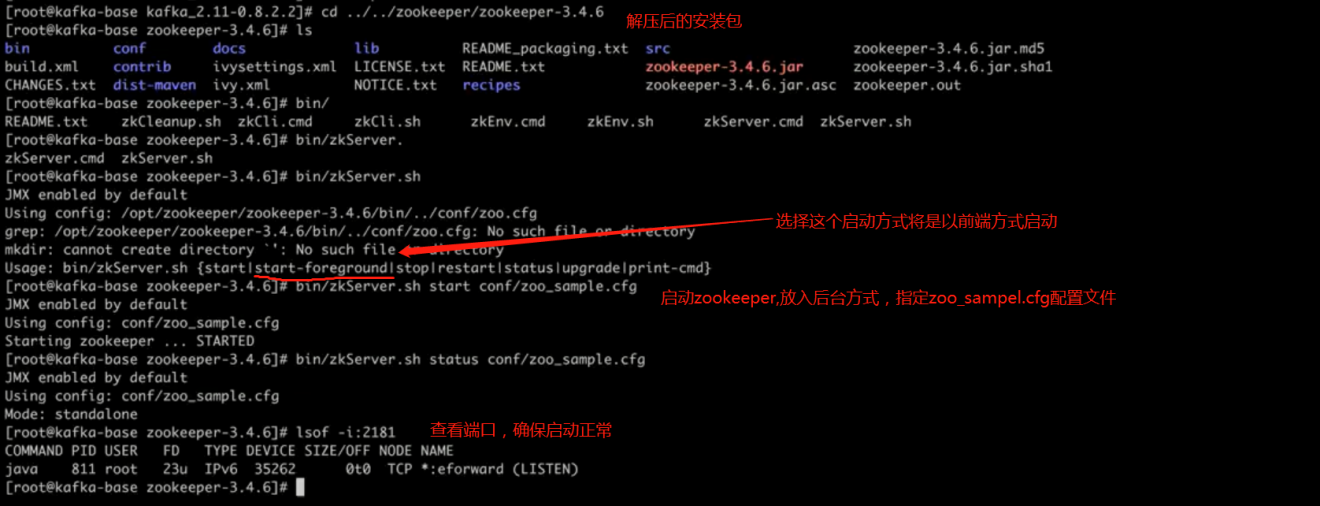


启用console-producer 使用发布消息

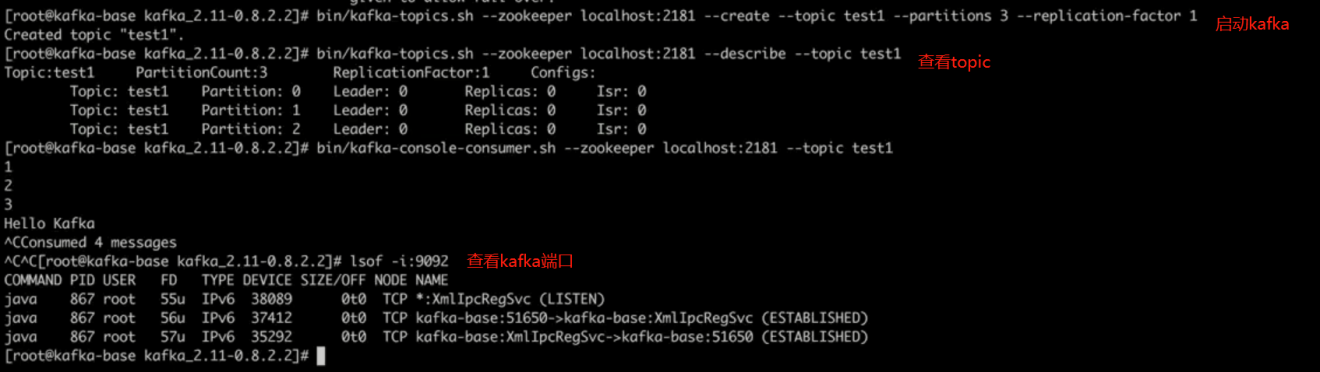


## Kafka安装----使用kafka单独的zookeeper

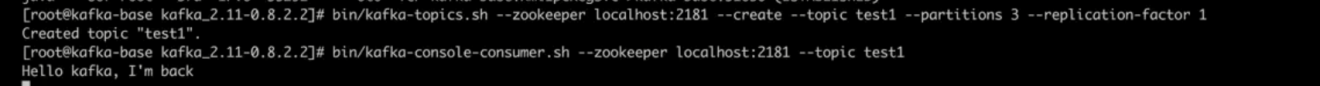
启动zookeeper



启动kafka



启用console-consumer订阅这个topic消息，并查看效果



启用console-producer 使用发布消息

