07 | 最最最重要的集群参数配置（上）

2019-06-18 胡夕

你好，我是胡夕。今天我想和你聊聊最最最重要的Kafka集群配置。我这里用了3个“最”字并非哗众取宠，而是因为有些配置的重要性并未体现在官方文档中，并且从实际表现看，很多参数对系统的影响要比从文档上看更加明显，因此很有必要集中讨论一下。

我希望通过两期内容把这些重要的配置讲清楚。严格来说这些配置并不单单指Kafka服务器端的配置，其中既有Broker端参数，也有主题（后面我用我们更熟悉的Topic表示）级别的参数、JVM端参数和操作系统级别的参数。下面我先从Broker端参数说起。

Broker端参数

目前Kafka Broker提供了近200个参数，这其中绝大部分参数都不用你亲自过问。当谈及这些参数的用法时，网上的文章多是罗列出一些常见的参数然后一个一个地给出它们的定义，事实上我以前写文章时也是这么做的。不过今天我打算换个方法，按照大的用途类别一组一组地介绍它们，希望可以更有针对性，也更方便你记忆。

首先Broker是需要配置存储信息的，即Broker使用哪些磁盘。那么针对存储信息的重要参数有以下这么几个：

- log.dirs：这是非常重要的参数，指定了Broker需要使用的若干个文件目录路径。要知道这个参数是没有默认值的，这说明什么？这说明它必须由你亲自指定。
- log.dir：注意这是dir，结尾没有s，说明它只能表示单个路径，它是补充上一个参数用的。

这两个参数应该怎么设置呢？很简单，你只要设置log.dirs，即第一个参数就好了，不要设置log.dir。而且更重要的是，在线上生产环境中一定要为log.dirs配置多个路径，具体格式是一个CSV格式，也就是用逗号分隔的多个路径，比如/home/kafka1,/home/kafka2,/home/kafka3这样。如果有条件的话你最好保证这些目录挂载到不同的物理磁盘上。这样做有两个好处：

- 提升读写性能：比起单块磁盘，多块物理磁盘同时读写数据有更高的吞吐量。
- 能够实现故障转移：即Failover。这是Kafka 1.1版本新引入的强大功能。要知道在以前，只要Kafka Broker使用的任何一块磁盘挂掉了，整个Broker进程都会关闭。但是自1.1开始，这种情况被修正了，坏掉的磁盘上的数据会自动地转移到其他正常的磁盘上，而且Broker还能正常工作。还记得上一期我们关于Kafka是否需要使用RAID的讨论吗？这个改进正是我们舍弃RAID方案的基础：没有这种Failover的话，我们只能依靠RAID来提供保障。

下面说说与ZooKeeper相关的设置。首先ZooKeeper是做什么的呢？它是一个分布式协调框架，负责协调管理并保存Kafka集群的所有元数据信息，比如集群都有哪些Broker在运行、创建了哪些Topic，每个Topic都有多少分区以及这些分区的Leader副本都在哪些机器上等信息。

Kafka与ZooKeeper相关的最重要的参数当属zookeeper.connect。这也是一个CSV格式的参数，比如我可以指定它的值为zk1:2181,zk2:2181,zk3:2181。2181是ZooKeeper的默认端口。

现在问题来了，如果我让多个Kafka集群使用同一套ZooKeeper集群，那么这个参数应该怎么设置呢？这时候chroot就派上用场了。这个chroot是ZooKeeper的概念，类似于别名。

如果你有两套Kafka集群，假设分别叫它们kafka1和kafka2，那么两套集群的zookeeper.connect参数可以这样指定：zk1:2181,zk2:2181,zk3:2181/kafka1和zk1:2181,zk2:2181,zk3:2181/kafka2。切记chroot只需要写一次，而且是加到最后的。我经常碰到有人这样指定：zk1:2181/kafka1,zk2:2181/kafka2,zk3:2181/kafka3，这样的格式是不对的。

第三组参数是与Broker连接相关的，即客户端程序或其他Broker如何与该Broker进行通信的设置。有以下三个参数：

- listeners：学名叫监听器，其实就是告诉外部连接者要通过什么协议访问指定主机名和端口开放的Kafka服务。
- advertised.listeners：和listeners相比多了个advertised。Advertised的含义表示宣称的、公布的，就是说这组监听器是Broker用于对外发布的。
- host.name/port：列出这两个参数就是想说你把它们忘掉吧，压根不要为它们指定值，毕竟都是过期的参数了。

我们具体说说监听器的概念，从构成上来说，它是若干个逗号分隔的三元组，每个三元组的格式为<协议名称，主机名，端口号>。这里的协议名称可能是标准的名字，比如PLAINTEXT表示明

文传输、SSL表示使用SSL或TLS加密传输等；也可能是你自己定义的协议名字，比如CONTROLLER: //localhost:9092。

一旦你自己定义了协议名称，你必须还要指定listener.security.protocol.map参数告诉这个协议底层使用了哪种安全协议，比如指定listener.security.protocol.map=CONTROLLER:PLAINTEXT表示CONTROLLER这个自定义协议底层使用明文不加密传输数据。

至于三元组中的主机名和端口号则比较直观，不需要做过多解释。不过有个事情你还是要注意一下，经常有人会问主机名这个设置中我到底使用IP地址还是主机名。**这里我给出统一的建议：最好全部使用主机名，即Broker端和Client端应用配置中全部填写主机名。** Broker源代码中也使用的是主机名，如果你在某些地方使用了IP地址进行连接，可能会发生无法连接的问题。

第四组参数是关于Topic管理的。我来讲讲下面这三个参数：

- auto.create.topics.enable：是否允许自动创建Topic。
- unclean.leader.election.enable：是否允许Unclean Leader选举。
- auto.leader.rebalance.enable：是否允许定期进行Leader选举。

我还是一个个说。

auto.create.topics.enable参数我建议最好设置成false，即不允许自动创建Topic。在我们的线上环境里面有很多名字稀奇古怪的Topic，我想大概都是因为该参数被设置成了true的缘故。

你可能有这样的经历，要为名为test的Topic发送事件，但是不小心拼写错误了，把test写成了tst，之后启动了生产者程序。恭喜你，一个名为tst的Topic就被自动创建了。

所以我一直相信好的运维应该防止这种情形的发生，特别是对于那些大公司而言，每个部门被分配的Topic应该由运维严格把控，决不能允许自行创建任何Topic。

第二个参数unclean.leader.election.enable是关闭Unclean Leader选举的。何谓Unclean？还记得Kafka有多个副本这件事吗？每个分区都有多个副本来提供高可用。在这些副本中只能有一个副本对外提供服务，即所谓的Leader副本。

那么问题来了，这些副本都有资格竞争Leader吗？显然不是，只有保存数据比较多的那些副本才有资格竞选，那些落后进度太多的副本没资格做这件事。

好了，现在出现这种情况了：假设那些保存数据比较多的副本都挂了怎么办？我们还要不要进行Leader选举了？此时这个参数就派上用场了。

如果设置成false，那么就坚持之前的原则，坚决不能让那些落后太多的副本竞选Leader。这样做的后果是这个分区就不可用了，因为没有Leader了。反之如果是true，那么Kafka允许你从那些“跑得慢”的副本中选一个出来当Leader。这样做的后果是数据有可能就丢失了，因为这些副本

保存的数据本来就不全，当了Leader之后它本人就变得膨胀了，认为自己的数据才是权威的。

这个参数在最新版的Kafka中默认就是false，本来不需要我特意提的，但是比较搞笑的是社区对这个参数的默认值来来回回改了好几版了，鉴于我不知道你用的是哪个版本的Kafka，所以建议你还是显式地把它设置成false吧。

第三个参数auto.leader.rebalance.enable的影响貌似没什么人提，但其实对生产环境影响非常大。设置它的值为true表示允许Kafka定期地对一些Topic分区进行Leader重选举，当然这个重选举不是无脑进行的，它要满足一定的条件才会发生。严格来说它与上一个参数中Leader选举的最大不同在于，它不是选Leader，而是换Leader！比如Leader A一直表现得很好，但若auto.leader.rebalance.enable=true，那么有可能一段时间后Leader A就要被强行卸任换成Leader B。

你要知道换一次Leader代价很高的，原本向A发送请求的所有客户端都要切换成向B发送请求，而且这种换Leader本质上没有任何性能收益，因此我建议你在生产环境中把这个参数设置成false。

最后一组参数是数据留存方面的，即：

- log.retention.{hour|minutes|ms}：这是个“三兄弟”，都是控制一条消息数据被保存多长时间。从优先级上来说ms设置最高、minutes次之、hour最低。
- log.retention.bytes：这是指定Broker为消息保存的总磁盘容量大小。
- message.max.bytes：控制Broker能够接收的最大消息大小。

先说这个“三兄弟”，虽然ms设置有最高的优先级，但是通常情况下我们还是设置hour级别的多一些，比如log.retention.hour=168表示默认保存7天的数据，自动删除7天前的数据。很多公司把Kafka当做存储来使用，那么这个值就要相应地调大。

其次是这个log.retention.bytes。这个值默认是-1，表明你想在这台Broker上保存多少数据都可以，至少在容量方面Broker绝对为你开绿灯，不会做任何阻拦。这个参数真正发挥作用的场景其实是在云上构建多租户的Kafka集群：设想你要做一个云上的Kafka服务，每个租户只能使用100GB的磁盘空间，为了避免有个“恶意”租户使用过多的磁盘空间，设置这个参数就显得至关重要了。

最后说说message.max.bytes。实际上今天我和你说的重要参数都是指那些不能使用默认值的参数，这个参数也是一样，默认的1000012太少了，还不到1MB。实际场景中突破1MB的消息都是屡见不鲜的，因此在线上环境中设置一个比较大的值还是比较保险的做法。毕竟它只是一个标尺而已，仅仅衡量Broker能够处理的最大消息大小，即使设置大一点也不会耗费什么磁盘空间的。

小结

再次强调一下，今天我和你分享的所有参数都是那些要修改默认值的参数，因为它们的默认值不

适合一般的生产环境。当然，我并不是说其他100多个参数就不重要。事实上，在专栏的后面我们还会陆续提到其他的一些参数，特别是那些和性能息息相关的参数。所以今天我提到的所有参数，我希望作为一个最佳实践给到你，可以有的放矢地帮助你规划和调整你的Kafka生产环境。

开放讨论

除了今天我分享的这些参数，还有哪些参数是你认为比较重要而文档中没有提及的？你曾踩过哪些关于参数配置的“坑”？欢迎提出来与我和大家一起讨论。

欢迎你写下自己的思考或疑问，我们一起讨论。如果你觉得有所收获，也欢迎把文章分享给你的朋友。

精选留言

草帽路飞 14

老师 advertised.listeners 这个配置能否再解释一下。感觉配置了 listeners之后就不用配置这个了呀？

2019-06-18

作者回复

advertised.listeners主要是为外网访问用的。如果clients在内网环境访问Kafka不需要配置这个参数。

常见的玩法是：你的Kafka Broker机器上配置了双网卡，一块网卡用于内网访问（即我们常说

的内网IP）；另一个块用于外网访问。那么你可以配置listeners为内网IP，advertised.listeners为外网IP。

2019-06-18

QQ怪 3

老师帮我们讲讲这个参数吧auto.offset.reset，我有时候删除一个topic时会导致offset异常，出现重复消费问题，不知道跟这个参数有没有关系？？

2019-06-18

作者回复

不太懂“删除topic后还出现重复消费”是什么意思？删完了还要继续消费它吗？

当consumer启动后它会从Kafka读取它上次消费的位移。情况1： 如果 Kafka broker端没有保存这个位移值，那么consumer会看auto.offset.reset的脸色

情况2：consumer拿到位移值开始消费，如果后面发现它要读取消息的位移在Kafka中不存在（可能对应的消息已经被删除了），那么它也会看auto.offset.reset的脸色

情况3：除以上这两种情况之外consumer不会再顾忌auto.offset.reset的值

怎么看auto.offset.reset的脸色呢？简单说就是earliest从头消息；latest从当前新位移处消费。

2019-06-19

mickle 2

老师，对于磁盘坏掉以后转移到其他磁盘的机制，我有点疑问，如果已经坏掉，则不可读了，那么是不是只能从副本去转移了，如果从副本转移那就有可能会丢失部分最新的数据吧？

2019-06-18

作者回复

不会啊，broker会重建副本，然后走正常的同步机制：从leader处拉取数据。

2019-06-19

小头针 1

胡老师，我在kafka升级过程中遇到过这样的问题，就是升级后的Kafka与之前的Kafka 的配置完全一样，就是版本不一样了。但是5个Broker后，Kafka Manager工具中，只有1个Broker有数据进入进出。后来同时添加了以下4个参数：

rebalance.max.retries=4

auto.leader.rebalance.enable=true

leader.imbalance.check.interval.seconds=300

leader.imbalance.per.broker.percentage=10

再重启Kafka，5个Broker都有数据进入进出，但是我不清楚这到底是哪个参数起到了决定性的作用。其中就有老师讲的auto.leader.rebalance.enable这个参数，但是我这里设置的是true？

2019-06-24

作者回复

只有一个broker有数据进出，我猜是因为这样的原因：1. 首先你的主题分区副本数是1；2. 在你升级的过程中所有分区的Leader副本都变更到了同一台broker上。

后面开启了auto.leader.rebalance.enable=true之后它定期将Leader副本分散到不同broker上了。

2019-06-25

kaiux

1

，相当于把Kafka里面的一些坑预先告诉了我们。

2019-06-24

Geek_edc612

1

胡老师您好，我对这两个参数有些疑问：
（1）auto.leader.rebalance.enable 这个值设置为true，那么您说的定期重新选举，应该有个触发的条件吧？我刚才跟同事沟通过，他说是跟每台broker的leader数量有关，如果leader分布不均衡就会触发重新选举leader，但是感觉说的还是不够具体，您能给解答下吗，感谢
（2）log.retention.bytes这个参数，您说的对于总磁盘容量来说，那我这样理解您看正确不（极端情况）---这个值我设置为100G，我机器有3个磁盘，每个磁盘大小1T，每个磁盘有不同topic的partition且，如果一个租户恶意写数据到自己的topic，造成某块磁盘的partition大小为100G，那么这台broker是不是所有topic都无法继续写入数据了？劳烦您解答下，感谢

2019-06-19

作者回复

1. 的确是有个比例，要超过这个比例才会触发preferred leader选举。这个比例由broker端参数leader.imbalance.per.broker.percentage控制，默认是10%。举个例子，如果一个broker上有10个分区，有2个分区的leader不是preferred leader，那么就会触发

2. 没太明白为什么写到100GB，broker就不能继续写入了？

2019-06-19

李 P

1

和本节无关，消息队列重复消费问题有什么太好的办法吗？我们现在的做法是把offset和消费后的计算结果一并保存在业务系统中，有没有更好的做法

2019-06-19

作者回复

可以试试Kafka 0.11引入的事务

2019-06-19

Liam

1

请问老师，坏掉的数据是怎么自动转移到其他磁盘上的呢？

2019-06-19

作者回复

可能有点没说清楚。

1. Broker自动在好的路径上重建副本，然后从leader同步；

2. Kafka支持工具能够将某个路径上的数据拷贝到其他路径上

2019-06-19

不了峰

1

请教老师

gg.handler.kafkahandler.Mode = tx

gg.handler.kafkahandler.Mode = op

这两个的差别。我们遇到时 dml 数据会丢失的情况。用的是 op 。

谢谢

2019-06-18

作者回复

搜了一下，像是Oracle GoldenGate Kafka Adapter的参数。我没有用过，从文档中看这两者的区别是：当设置成op单个数据库表的变更（插入、更新、删除）会被当成一条Kafka消息发送；当设置成tx时，数据库事务所做的所有变更统一被封装进一条Kafka消息，并在事务提交后被发送。

显然，后者有事务性的保障，至少有原子性方面的保证，不会丢失部分CDC数据。

2019-06-19

你看起来很好吃

1

'如果设置成 false，那么就坚持之前的原则，坚决不能让那些落后太多的副本竞选 Leader。'想问一下老师，每个partition的副本保存的数据不是应该和leader是一模一样的吗？为什么会有丢失的？

2019-06-18

作者回复

它们是异步拉取消息的，必然有一个时间窗口导致它和leader中的数据是不一致的，或者说它是落后于leader的。

2019-06-19

南辕北辙

1

刚用的时候大一点的消息就有问题，后来知道是message.max.bytes,不过老师是不是打错单位了，记得是900多KB。今天干货很多

2019-06-18

作者回复

感谢反馈，sorry，笔误了:(

2019-06-18

Geek_jacky

1

老师好，如果磁盘坏掉了，这些数据是什么机制读取到其他磁盘上的呢？不是都坏了吗？不应该读取其他副本中的数据了吗？这个磁盘上的数据就算是丢失了吗？

2019-06-18

作者回复

Broker会在好的目录上重建副本。另外Kafka也提供了工具将某块磁盘上的数据直接搬移到另一

个磁盘上，毕竟磁盘坏了也不是不能修好：）

2019-06-18

bunny 1

后面会单独讲解producer，consumer相关配置参数吧？还有这个参数delete.topic.enable，胡老师有什么建议么？

2019-06-18

作者回复

后面讲到producer和consumer会有涉及，但不会专门来讲，毕竟很多人反映单纯讲配置参数太枯燥了，还是结合具体的使用场景来讲比较好。另外建议delete.topic.enable保持默认值true就好，毕竟不能删除topic总显得不太方便。只是要注意权限设置即可，不可能 让任何人都能有删除topic的权限。

2019-06-18

与狼共舞 0

老师，请教一个关于内外网设置方面的问题。如果每个broker节点的 advertised.listeners都设置为外网IP，这样一来，假设集群有多个 broker节点，是不是得要有多个外网ip？ 而外网ip这个资源比较有限，有没有更好的方法呢？

2019-06-29

作者回复

如果你的client只能通过外网IP访问具体的broker，那么broker只能设置外网IP

2019-07-01

王纪娟 0

你好，我目前在用0.10.2版本，实际用下来log.retention.bytes限定的是每个topic的每个分区的最大存储，但其实还可能超，因为还跟reflush 的间隔有关，同时我观察到kafka在写的时候会单独再开一个segment。所以某个时间点topic的实际存储可能大过配置值

2019-06-27

作者回复

它的算法比较复杂，不是简单的不超过就行了。有兴趣的话可以看下我写的这篇文章：https://www.cnblogs.com/huxi2b/p/8042099.html

2019-06-28

非礼勿言-非礼勿听-非礼勿视 0

竟然不能回复。老师，之前用的是0.11版本的，就是消费者处理消息如果时间过长的话，会长期阻塞在poll方法上，无法继续消费新的消息

2019-06-27

作者回复

“消费者处理消息如果时间过长” 为什么会阻塞在poll方法上呢？应该是阻塞在处理方法中吧？我觉得一个可能的原因是处理时间过长导致的频繁rebalance。看看日志是否存在频繁rebalance的情况。

2019-06-27

非礼勿言-非礼勿听-非礼勿视 0

老师你好，之前曾碰到过消费者处理时间过长，可能导致会话过期，然后消费者就再也拿不到数据了，无法消费，不晓得是什么原因

2019-06-26

作者回复

会话过期？是老版本consumer吗？有什么具体的日志打出来吗

2019-06-26

jacke 0

胡老师，2个问题：

1.zookeeper.connect的chroot这个点没有看懂，两套kafka集群不应该是每套都单独有自己的配置文件吗？kafka1和kafka2这2个别名在哪里设置的呢

2.unclean.leader.election.enable 设置为false表示落后太多进度的副本无法没有资格做完leader，落后的进度多少算多呢？这个也是可以配置的吗？

2019-06-25

作者回复

1. 是有独立的配置文件。这里的chroot设置是指在ZooKeeper中使用独立的znode父节点。

2. 由replica.lag.time.max.ms控制。

2019-06-26

ban 0

老师，我其实是想问下 租户 是什么意思，口语化来讲的话

2019-06-24

作者回复

不论是公有云还是私有云，云上面有很多个用户，他们以租赁的方式来订购云上的系统资源（CPU、RAM、DISK、Bandwidth）。一般管这些用户成为租户。如果构建和运维支持多租户的Kafka集群是我们需要研究的课题。

2019-06-25

Geek_986289 0

老师，这里的log.retention.hour 和log.retention.bytes指的是单个log吧，感觉如果是所有log 的大小似乎还能理解，但是所有log 的hour 似乎有点说不通

2019-06-23

作者回复

这些都是全局参数，针对broker上所有log而言的

2019-06-24