07 | 最最最重要的集群参数配置（上）

胡夕 · Kafka核心技术与实战

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。

需要你注意的是，这里所说的 Broker 端参数也被称为静态参数（Static Configs）。我会在专栏后面介绍与静态参数相对应的动态参数。所谓静态参数，是指你必须在 Kafka 的配置文件 server.properties 中进行设置的参数，不管你是新增、修改还是删除。同时，你必须重启 Broker 进程才能令它们生效。而主题级别参数的设置则有所不同，Kafka 提供了专门的 kafka-configs 命令来修改它们。至于 JVM 和操作系统级别参数，它们的设置方法比较通用化，我介绍的也都是标准的配置参数，因此，你应该很容易就能够对它们进行设置。

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，我分别介绍一下。

- `log.retention.{hours|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、hours 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 hours 级别的多一些，比如`log.retention.hours=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 当作存储来使用，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。

最最最重要的集群参数配置（上）

Broker端参数

- 与存储信息相关的参数：`log.dirs`和`log.dir`。
- 与ZooKeeper相关的参数：`zookeeper.connect`。
- 与Broker连接相关的参数：`listeners`、`advertised.listeners`和`host.name/port`。
- 关于Topic管理的参数：`auto.create.topics.enable`、`unclean.leader.election.enable`和`auto.leader.rebalance.enable`。
- 关于数据留存的参数：`log.retention.{hours|minutes|ms}`、`log.retention.bytes`和`message.max.bytes`。

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。

AI智能总结

Kafka集群参数配置对于构建高效消息系统至关重要。本文详细介绍了Kafka集群配置的重要性，包括Broker端参数、主题级别参数、JVM端参数和操作系统级别参数。强调了静态参数的重要性，需要在Kafka的配置文件中设置，并重启Broker进程生效。另外，介绍了使用kafka-configs命令修改主题级别参数，以及JVM和操作系统级别参数的设置方法。读者可通过本文快速了解Kafka集群配置的重要性和相关参数的设置方法，为构建高效的消息系统提供了重要参考。

文章内容主要围绕Kafka集群参数配置展开，包括Broker端参数、与ZooKeeper相关的设置、Broker连接相关的参数、Topic管理和数据留存方面的参数。对于每组参数，文章提供了详细的解释和建议，如存储信息的重要参数、与ZooKeeper相关的参数设置、Broker连接相关的设置、Topic管理参数和数据留存方面的参数。这些内容涵盖了构建高效消息系统所需的关键配置，对于Kafka用户具有实际指导意义。

在具体参数设置方面，文章提到了`log.retention.hours`和`log.retention.bytes`的重要性，以及`message.max.bytes`的设置建议。这些参数的合理配置对于Kafka集群的性能和稳定性至关重要，尤其在云上构建多租户的Kafka集群时更显重要。通过本文的总结，读者能够快速了解Kafka集群参数配置的关键性，并掌握相关参数的设置方法，为构建高效的消息系统提供了重要参考。

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

全部留言 (111)

最新　精选

大坏狐狸
2020-03-16

auto.create.topics.enable： 不能自立为王

unclean.leader.election.enable： 宁缺毋滥

auto.leader.rebalance.enable：江山不易改

log.retention.{hours|minutes|ms} ： 数据寿命 hours=168
log.rentention.bytes: 祖宅大小 -1 表示没限制
message.max.bytes: 祖宅大门宽度，默认 1000012=976KB

作者回复: 我去，这个过于形象了~

共 18 条评论>

266

草帽路飞
2019-06-18

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?

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。

共 8 条评论>

107

第一片心意
2019-12-13

auto.leader.rebalance.enable

关于这个参数的设置，我有一点不同的意见，官网说的是如果某个broker挂了，那分布在他上的leader副本就会自动切换到其他活着的broker上，但是挂掉的broker重启之后，集群并不会将他之前的leader副本再切换回来，这样就会使其他broker上leader副本数较多，而该broker上无leader副本（无新主题创建），从而造成负载不均衡的情况。

这时我们可以通过 kafka-preferred-replica-election.sh 脚本来重新平衡集群中的leader副本。但是我们配置这个参数为true的话，controller角色就会每五分钟（默认）检查一下集群不平衡的状态，进而重新平衡leader副本。

作者回复: 同意。不过实际上，线上环境贸然大面积迁移副本leader是非常有风险的事情：）

共 2 条评论>

45

QQ怪
2019-06-18

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？？

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。

39

H O ...
2020-04-09

老师 我把message.max.bytes设置地挺大，但是java生产者发送1M以上数据就失败，集群也重启过，版本0.10左右 是否有其他参数需要调?

作者回复: 需要。producer、broker、consumer三端都需要调整

broker: message.max.bytes和replica.fetch.max.bytes
consumer：fetch.message.max.bytes

共 3 条评论>

32

杨陆伟
2019-12-30

你好，log.retention.bytes这个参数是针对主题的吧？比如设置为100M，Kafka定期会把每个主题的日志数据留存到100M以下?

作者回复: 这个参数既有broker端也有topic端，不过最终都是作用于topic的。另外算法上也不是简单的比较大小。举个例子吧：假设日志段大小是700MB，当前分区共有4个日志段文件，大小分别是700MB，700MB，700MB和1234B——显然1234B那个文件就是active日志段。此时该分区总的日志大小是3*700MB+1234B=2100MB+1234B，如果阈值设置为2000MB，那么超出阈值的部分就是100MB+1234B，小于日志段大小700MB，故Kafka不会执行任何删除操作，即使总大小已经超过了阈值；反之如果阈值设置为1400MB，那么超过阈值的部分就是700MB+1234B > 700MB，此时Kafka会删除最老的那个日志段文件

共 5 条评论>

28

小头针
2019-06-24

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?

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。

共 3 条评论>

27

不了峰
2019-06-18

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

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。

15

你好旅行者

2019-06-18

老师好！关于Unclean这个参数，将其设置为false之后，就意味着如果ISR内的所有broker都宕机，那么这个分区就不可用了。
刚好我前几天看到饶军在2013年的一次报告上讲到Kafka在CAP问题上的取舍，他说，因为Kafka是部署在一个DataCenter中的，而一个DataCenter很少会出现Partitioning的情况，所以Kafka放弃了分区容忍性。
我想问的是，Kafka舍弃了分区容忍性这一点是否可以体现在社区默认将Unclean设置为false上呢?
附上报告的地址：https://www.youtube.com/watch?v=XcvHmqmh16g
关于CAP的取舍出现在21:50左右的地方。谢谢老师！

作者回复: 首先，CAP理论有很多有歧义的地方，我很好奇为什么国内很多人追捧CAP，其实对于分布式系统而言，很多一致性问题都是CAP覆盖不了的。
其次，我个人觉得饶大神并不是说Kafka放弃了P，其实Kafka是依托于ZooKeeper以及合理配置minlsr等参数来规避脑裂的。
第三，我翻看了社区对此提案的讨论，变更为false就是很朴素的思想：用户在默认情况下可能更加关心数据一致性，不想数据丢失。如果用户想要更高的可用性，手动调整即可。你可以看看社区对此问题的讨论：https://www.mail-archive.com/dev@kafka.apache.org/msg63086.html

15

咸淡一首诗

2020-02-14

老师，对于failover机制，kafka会新建副本，从leader处同步最新的数据给新建副本。如果坏掉的盘是leader持久化的盘并且其他副本没有来的及从坏掉的leader分区同步最新数据，重新选举leader后岂不是也会丢失数据？？？

作者回复: 是的，这种情况会丢失数据。其实Kafka并没有承诺不丢失数据，而是在满足某些条件下才做持久化保证。

11