Kafka

1 Leader 均衡

1.1 自查

Hdh121 broker 0

Hdh122 broker 1

Hdh123 broker 2

[root@hdh121 bin]# ./kafka-topics.sh --topic BAYONET\_VEHICLEPASS --describe --zookeeper hdh121:2181,hdh122:2181,hdh123:2181

Topic:BAYONET\_VEHICLEPASS PartitionCount:10 ReplicationFactor:2 Configs:

Topic: BAYONET\_VEHICLEPASS Partition: 0 Leader: 2 Replicas: 2,1 Isr: 2,1

Topic: BAYONET\_VEHICLEPASS Partition: 1 Leader: 0 Replicas: 0,2 Isr: 0,2

Topic: BAYONET\_VEHICLEPASS Partition: 2 Leader: 1 Replicas: 1,0 Isr: 1,0

Topic: BAYONET\_VEHICLEPASS Partition: 3 Leader: 2 Replicas: 2,0 Isr: 2,0

Topic: BAYONET\_VEHICLEPASS Partition: 4 Leader: 0 Replicas: 0,1 Isr: 0,1

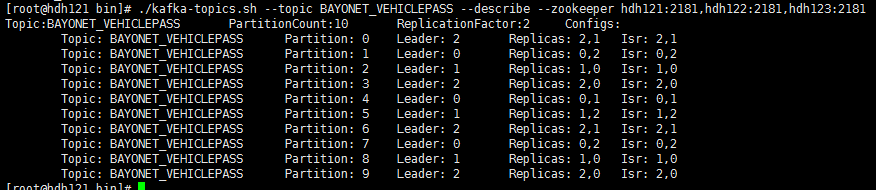
Topic: BAYONET\_VEHICLEPASS Partition: 5 Leader: 1 Replicas: 1,2 Isr: 1,2

Topic: BAYONET\_VEHICLEPASS Partition: 6 Leader: 2 Replicas: 2,1 Isr: 2,1

Topic: BAYONET\_VEHICLEPASS Partition: 7 Leader: 0 Replicas: 0,2 Isr: 0,2

Topic: BAYONET\_VEHICLEPASS Partition: 8 Leader: 1 Replicas: 1,0 Isr: 1,0

Topic: BAYONET\_VEHICLEPASS Partition: 9 Leader: 2 Replicas: 2,0 Isr: 2,0



此时重启hdh122,可以看到原来Leader是hdh122也就是broker1的全部转成了broker0(hdh121)和broker2(hdh122)上

[root@hdh121 bin]# ./kafka-topics.sh --topic BAYONET\_VEHICLEPASS --describe --zookeeper hdh121:2181,hdh122:2181,hdh123:2181

Topic:BAYONET\_VEHICLEPASS PartitionCount:10 ReplicationFactor:2 Configs:

Topic: BAYONET\_VEHICLEPASS Partition: 0 Leader: 2 Replicas: 2,1 Isr: 2

Topic: BAYONET\_VEHICLEPASS Partition: 1 Leader: 0 Replicas: 0,2 Isr: 0,2

**Topic: BAYONET\_VEHICLEPASS Partition: 2 Leader: 0 Replicas: 1,0 Isr: 0**

Topic: BAYONET\_VEHICLEPASS Partition: 3 Leader: 2 Replicas: 2,0 Isr: 2,0

Topic: BAYONET\_VEHICLEPASS Partition: 4 Leader: 0 Replicas: 0,1 Isr: 0

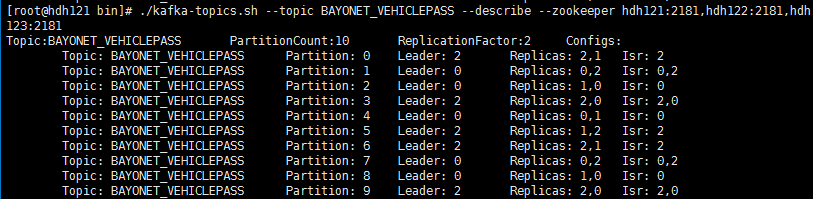
**Topic: BAYONET\_VEHICLEPASS Partition: 5 Leader: 2 Replicas: 1,2 Isr: 2**

Topic: BAYONET\_VEHICLEPASS Partition: 6 Leader: 2 Replicas: 2,1 Isr: 2

Topic: BAYONET\_VEHICLEPASS Partition: 7 Leader: 0 Replicas: 0,2 Isr: 0,2

**Topic: BAYONET\_VEHICLEPASS Partition: 8 Leader: 0 Replicas: 1,0 Isr: 0**

Topic: BAYONET\_VEHICLEPASS Partition: 9 Leader: 2 Replicas: 2,0 Isr: 2,0



当hdh122重启好之后, 又恢复了

[root@hdh121 bin]# ./kafka-topics.sh --topic BAYONET\_VEHICLEPASS --describe --zookeeper hdh121:2181,hdh122:2181,hdh123:2181

Topic:BAYONET\_VEHICLEPASS PartitionCount:10 ReplicationFactor:2 Configs:

Topic: BAYONET\_VEHICLEPASS Partition: 0 Leader: 2 Replicas: 2,1 Isr: 2,1

Topic: BAYONET\_VEHICLEPASS Partition: 1 Leader: 0 Replicas: 0,2 Isr: 0,2

**Topic: BAYONET\_VEHICLEPASS Partition: 2 Leader: 1 Replicas: 1,0 Isr: 1,0**

Topic: BAYONET\_VEHICLEPASS Partition: 3 Leader: 2 Replicas: 2,0 Isr: 2,0

Topic: BAYONET\_VEHICLEPASS Partition: 4 Leader: 0 Replicas: 0,1 Isr: 0,1

**Topic: BAYONET\_VEHICLEPASS Partition: 5 Leader: 1 Replicas: 1,2 Isr: 1,2**

Topic: BAYONET\_VEHICLEPASS Partition: 6 Leader: 2 Replicas: 2,1 Isr: 2,1

Topic: BAYONET\_VEHICLEPASS Partition: 7 Leader: 0 Replicas: 0,2 Isr: 0,2

**Topic: BAYONET\_VEHICLEPASS Partition: 8 Leader: 1 Replicas: 1,0 Isr: 1,0**

Topic: BAYONET\_VEHICLEPASS Partition: 9 Leader: 2 Replicas: 2,0 Isr: 2,0

恢复原因:

Kafka的server.properties有一个默认参数: auto.leader.rebalance.enable=true 作用就是Leader平衡.

也可以使用手动恢复方式kafka-preferred-replica-election.sh   \

--zookeeper hdh:2181,hdh122:2181,hdh123:2181

1.2 网络解释

当一个broker停止或者crashes时，所有本来将它作为leader的分区将会把leader转移到其它broker上去。这意味着当这个broker重启时，它将不再担任何分区的leader，kafka的client也不会从这个broker来读取消息，从而导致资源的浪费。比如下面的broker 7是挂掉重启的，我们可以发现Partition 1虽然在broker 7上有数据，但是由于它挂了，所以[Kafka](https://www.iteblog.com/archives/tag/kafka/" \o "" \t "_blank)重新将broker 3当作该分区的Leader，然而broker 3已经是Partition 6的Leader了。

|  |
| --- |
| [iteblog@www.iteblog.com ~]$ kafka-topics.sh --topic iteblog   \  --describe --zookeeper www.iteblog.com:2181    Topic:iteblog PartitionCount:7  ReplicationFactor:2 Configs:    Topic: iteblog  Partition: 0  Leader: 1 Replicas: 1,4 Isr: 1,4    Topic: iteblog  Partition: 1  Leader: 3 Replicas: 7,3 Isr: 3,7    Topic: iteblog  Partition: 2  Leader: 5 Replicas: 5,7 Isr: 5,7    Topic: iteblog  Partition: 3  Leader: 6 Replicas: 6,1 Isr: 1,6    Topic: iteblog  Partition: 4  Leader: 4 Replicas: 4,2 Isr: 4,2    Topic: iteblog  Partition: 5  Leader: 2 Replicas: 2,5 Isr: 5,2    Topic: iteblog  Partition: 6  Leader: 3 Replicas: 3,6 Isr: 3,6 |

　　幸运的是，[Kafka](https://www.iteblog.com/archives/tag/kafka/" \o "" \t "_blank)中有一个被称为优先副本(preferred replicas)的概念。如果一个分区有3个副本，且这3个副本的优先级别分别为1，5，9，根据优先副本的概念，1会作为leader。为了使kafka集群恢复默认的leader，需要运行以下命令：

|  |
| --- |
| [iteblog@www.iteblog.com ~]$ kafka-preferred-replica-election.sh   \  --zookeeper www.iteblog.com:2181    Successfully started preferred replica election for partitions Set([iteblog,1],   [iteblog,5], [iteblog,4], [iteblog,6], [iteblog,2], [iteblog,0], [iteblog,3]) |

　　每次运行上面的命令是比较烦躁的，不过[Kafka](https://www.iteblog.com/archives/tag/kafka/" \o "" \t "_blank)为我们提供了一个参数，可以使得Kafka集群自动平衡Leader，我们只需要在server.properties文件中配置如下设置：

|  |
| --- |
| auto.leader.rebalance.enable=true |

这个值默认就是打开的。下面是Leader平衡的结果：

|  |
| --- |
| [iteblog@www.iteblog.com ~]$ kafka-topics.sh --topic iteblog --describe \  --zookeeper www.iteblog.com:2181    Topic:iteblog PartitionCount:7  ReplicationFactor:2 Configs:    Topic: iteblog  Partition: 0  Leader: 1 Replicas: 1,4 Isr: 1,4    Topic: iteblog  Partition: 1  Leader: 7 Replicas: 7,3 Isr: 3,7    Topic: iteblog  Partition: 2  Leader: 5 Replicas: 5,7 Isr: 5,7    Topic: iteblog  Partition: 3  Leader: 6 Replicas: 6,1 Isr: 1,6    Topic: iteblog  Partition: 4  Leader: 4 Replicas: 4,2 Isr: 4,2    Topic: iteblog  Partition: 5  Leader: 2 Replicas: 2,5 Isr: 5,2    Topic: iteblog  Partition: 6  Leader: 3 Replicas: 3,6 Isr: 3,6 |

可以看出broker 7重新变成Partition 1的Leader了。