Why do we have to balance topics in a Kafka cluster?

Whenever a Kafka node is down, the load of that server is distributed to the other nodes in the cluster and this distribution is not even, i.e the load is not distributed evenly across all nodes in the cluster. We need to do some steps to achieve this balancing (also called rebalancing).

There are two things that people usually mean when they talk about rebalancing. One is leader re-election, or preferred replica election and the other one is partition rebalancing. The first one is more of a case when a node is down and is brought back again within a certain period of time . The other one is when we want to either decrease or increase the number of nodes in the cluster. Let's see how to deal with these different scenarios to balance topics.

All the examples below assume a 5-broker Kafka cluster.

Scenario 1: When the broker is down because of maintenance or due to server failure and is brought back within a certain period of time.

There are two ways to handle this scenario. One is adding the following line to the broker configuration "*auto.leader.rebalance.enable*" to automatically rebalance, but this is reported to have issues. The other way is to manually use the "kafka-preferred-replica-election.sh" tool.

Edit *server.properties* of Kafka to add the following line,*auto.leader.rebalance.enable = false*

```
kafka@kafka1:~/kafka$ vim config/server.properties
kafka@kafka1:~/kafka$ nohup bin/kafka-server-start.sh config/server.properties > kafa.log
2>&1 &
[1] 1195
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2181,zoo4:2
181,zoo5:2181/kafka --replication-factor 3 --partitions 10 --create --topic test
Created topic "test".
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2181,zoo4:2
181,zoo5:2181/kafka --replication-factor 3 --partitions 10 --create --topic test1
Created topic "test1".
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2181,zoo4:2
181,zoo5:2181/kafka --describe
Topic:test      PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test     Partition: 0    Leader: 4       Replicas: 4,2,3 Isr: 4,2,3
        Topic: test     Partition: 1    Leader: 5       Replicas: 5,3,4 Isr: 5,3,4
        Topic: test     Partition: 2    Leader: 1       Replicas: 1,4,5 Isr: 1,4,5
        Topic: test     Partition: 3    Leader: 2       Replicas: 2,5,1 Isr: 2,5,1
        Topic: test     Partition: 4    Leader: 3       Replicas: 3,1,2 Isr: 3,1,2
        Topic: test     Partition: 5    Leader: 4       Replicas: 4,3,5 Isr: 4,3,5
        Topic: test     Partition: 6    Leader: 5       Replicas: 5,4,1 Isr: 5,4,1
        Topic: test     Partition: 7    Leader: 1       Replicas: 1,5,2 Isr: 1,5,2
        Topic: test     Partition: 8    Leader: 2       Replicas: 2,1,3 Isr: 2,1,3
        Topic: test     Partition: 9    Leader: 3       Replicas: 3,2,4 Isr: 3,2,4
Topic:test1     PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test1    Partition: 0    Leader: 5       Replicas: 5,1,2 Isr: 5,1,2
        Topic: test1    Partition: 1    Leader: 1       Replicas: 1,2,3 Isr: 1,2,3
        Topic: test1    Partition: 2    Leader: 2       Replicas: 2,3,4 Isr: 2,3,4
        Topic: test1    Partition: 3    Leader: 3       Replicas: 3,4,5 Isr: 3,4,5
        Topic: test1    Partition: 4    Leader: 4       Replicas: 4,5,1 Isr: 4,5,1
        Topic: test1    Partition: 5    Leader: 5       Replicas: 5,2,3 Isr: 5,2,3
        Topic: test1    Partition: 6    Leader: 1       Replicas: 1,3,4 Isr: 1,3,4
        Topic: test1    Partition: 7    Leader: 2       Replicas: 2,4,5 Isr: 2,4,5
        Topic: test1    Partition: 8    Leader: 3       Replicas: 3,5,1 Isr: 3,5,1
        Topic: test1    Partition: 9    Leader: 4       Replicas: 4,1,2 Isr: 4,1,2
```

Let us bring broker4 down and see how the topic load is distributed.

```
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2181,zoo4:
181,zoo5:2181/kafka --describe
Topic:test      PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test     Partition: 0    Leader: 2       Replicas: 4,2,3 Isr: 2,3
        Topic: test     Partition: 1    Leader: 5       Replicas: 5,3,4 Isr: 5,3
        Topic: test     Partition: 2    Leader: 1       Replicas: 1,4,5 Isr: 1,5
        Topic: test     Partition: 3    Leader: 2       Replicas: 2,5,1 Isr: 2,5,1
        Topic: test     Partition: 4    Leader: 3       Replicas: 3,1,2 Isr: 3,1,2
        Topic: test     Partition: 5    Leader: 3       Replicas: 4,3,5 Isr: 3,5
        Topic: test     Partition: 6    Leader: 5       Replicas: 5,4,1 Isr: 5,1
        Topic: test     Partition: 7    Leader: 1       Replicas: 1,5,2 Isr: 1,5,2
        Topic: test     Partition: 8    Leader: 2       Replicas: 2,1,3 Isr: 2,1,3
        Topic: test     Partition: 9    Leader: 3       Replicas: 3,2,4 Isr: 3,2
Topic:test1     PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test1    Partition: 0    Leader: 5       Replicas: 5,1,2 Isr: 5,1,2
        Topic: test1    Partition: 1    Leader: 1       Replicas: 1,2,3 Isr: 1,2,3
        Topic: test1    Partition: 2    Leader: 2       Replicas: 2,3,4 Isr: 2,3
        Topic: test1    Partition: 3    Leader: 3       Replicas: 3,4,5 Isr: 3,5
        Topic: test1    Partition: 4    Leader: 5       Replicas: 4,5,1 Isr: 5,1
        Topic: test1    Partition: 5    Leader: 5       Replicas: 5,2,3 Isr: 5,2,3
        Topic: test1    Partition: 6    Leader: 1       Replicas: 1,3,4 Isr: 1,3
        Topic: test1    Partition: 7    Leader: 2       Replicas: 2,4,5 Isr: 2,5
        Topic: test1    Partition: 8    Leader: 3       Replicas: 3,5,1 Isr: 3,5,1
        Topic: test1    Partition: 9    Leader: 1       Replicas: 4,1,2 Isr: 1,2
kafka@kafka1:~/kafka$
```

Here we could see that the load is not evenly distributed. Let's bring back broker 4 online.

```
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2
181,zoo5:2181/kafka --describe
Topic:test      PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test     Partition: 0    Leader: 2       Replicas: 4,2,3 Isr: 2,3
        Topic: test     Partition: 1    Leader: 5       Replicas: 5,3,4 Isr: 5,3
        Topic: test     Partition: 2    Leader: 1       Replicas: 1,4,5 Isr: 1,5
        Topic: test     Partition: 3    Leader: 2       Replicas: 2,5,1 Isr: 2,5
        Topic: test     Partition: 4    Leader: 3       Replicas: 3,1,2 Isr: 3,1
        Topic: test     Partition: 5    Leader: 3       Replicas: 4,3,5 Isr: 3,5
        Topic: test     Partition: 6    Leader: 5       Replicas: 5,4,1 Isr: 5,1
        Topic: test     Partition: 7    Leader: 1       Replicas: 1,5,2 Isr: 1,5
        Topic: test     Partition: 8    Leader: 2       Replicas: 2,1,3 Isr: 2,1
        Topic: test     Partition: 9    Leader: 3       Replicas: 3,2,4 Isr: 3,2
Topic:test1     PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test1    Partition: 0    Leader: 5       Replicas: 5,1,2 Isr: 5,1
        Topic: test1    Partition: 1    Leader: 1       Replicas: 1,2,3 Isr: 1,2
        Topic: test1    Partition: 2    Leader: 2       Replicas: 2,3,4 Isr: 2,3
        Topic: test1    Partition: 3    Leader: 3       Replicas: 3,4,5 Isr: 3,5
        Topic: test1    Partition: 4    Leader: 5       Replicas: 4,5,1 Isr: 5,1
        Topic: test1    Partition: 5    Leader: 5       Replicas: 5,2,3 Isr: 5,2
        Topic: test1    Partition: 6    Leader: 1       Replicas: 1,3,4 Isr: 1,3
        Topic: test1    Partition: 7    Leader: 2       Replicas: 2,4,5 Isr: 2,5
        Topic: test1    Partition: 8    Leader: 3       Replicas: 3,5,1 Isr: 3,5
        Topic: test1    Partition: 9    Leader: 1       Replicas: 4,1,2 Isr: 1,2
kafka@kafka1:~/kafka$
```

Now we can see that even though the node 4 is back online, it is not serving as a leader for any of the partitions. Let's run the kafka-preferred-replica-election.sh tool to balance the load

```
kafka@kafka1:~/kafka$ bin/kafka-preferred-replica-election.sh --zookeeper zoo1:2
181,zoo3:2181,zoo4:2181,zoo5:2181/kafka
Successfully started preferred replica election for partitions Set([test,4], [te
st1,9], [test1,6], [test,3], [test1,3], [test,9], [test,2], [test1,1], [test1,7]
], [test1,5], [test,1], [test,7], [test1,2], [test,8], [test,6], [test,5], [test
t1,4])
kafka@kafka1:~/kafka$
```

```
kafka@kafka1:~/kafka$ bin/kafka-topics.sh --zookeeper zoo1:2181,zoo2:2181,zoo3:2
181,zoo5:2181/kafka --describe
Topic:test      PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test     Partition: 0    Leader: 4       Replicas: 4,2,3 Isr: 2,3
        Topic: test     Partition: 1    Leader: 5       Replicas: 5,3,4 Isr: 5,3
        Topic: test     Partition: 2    Leader: 1       Replicas: 1,4,5 Isr: 1,5
        Topic: test     Partition: 3    Leader: 2       Replicas: 2,5,1 Isr: 2,5
        Topic: test     Partition: 4    Leader: 3       Replicas: 3,1,2 Isr: 3,1
        Topic: test     Partition: 5    Leader: 4       Replicas: 4,3,5 Isr: 3,5
        Topic: test     Partition: 6    Leader: 5       Replicas: 5,4,1 Isr: 5,1
        Topic: test     Partition: 7    Leader: 1       Replicas: 1,5,2 Isr: 1,5
        Topic: test     Partition: 8    Leader: 2       Replicas: 2,1,3 Isr: 2,1
        Topic: test     Partition: 9    Leader: 3       Replicas: 3,2,4 Isr: 3,2
Topic:test1     PartitionCount:10       ReplicationFactor:3     Configs:
        Topic: test1    Partition: 0    Leader: 5       Replicas: 5,1,2 Isr: 5,1
        Topic: test1    Partition: 1    Leader: 1       Replicas: 1,2,3 Isr: 1,2
        Topic: test1    Partition: 2    Leader: 2       Replicas: 2,3,4 Isr: 2,3
        Topic: test1    Partition: 3    Leader: 3       Replicas: 3,4,5 Isr: 3,5
        Topic: test1    Partition: 4    Leader: 4       Replicas: 4,5,1 Isr: 5,1
        Topic: test1    Partition: 5    Leader: 5       Replicas: 5,2,3 Isr: 5,2
        Topic: test1    Partition: 6    Leader: 1       Replicas: 1,3,4 Isr: 1,3
        Topic: test1    Partition: 7    Leader: 2       Replicas: 2,4,5 Isr: 2,5
        Topic: test1    Partition: 8    Leader: 3       Replicas: 3,5,1 Isr: 3,5
        Topic: test1    Partition: 9    Leader: 4       Replicas: 4,1,2 Isr: 1,2
kafka@kafka1:~/kafka$
```

Now we can see that topics are evenly balanced.

Scenario 2: When a node is down and not recoverable.

We create a new broker and update the broker.id with the previous one's id which was not recoverable and manually run "kafka-preferred-replica-election.sh" for topic balancing.

Scenario 3: To increase or decrease the number of nodes in a Kafka cluster.

Following are the steps to balance topics when increase or decreasing number of nodes.

1. Using partition reassignment tool (kafka-reassign-partition.sh), generate (with the –generate option) the candidate assignment configuration. This shows the current and the proposed replica assignments.
2. Copy the proposed assignment to a JSON file.
3. Execute the partition reassignment tool (kafka-reassign-partition.sh –execute) to update the metadata for balancing. Make sure run this at a time when there is not much load on the cluster as this moves the data between different available nodes.
4. Wait for a while (based on the amount of data that has to move around) and verify that the balancing is completed successfully with –verify option.

5. Once the partition reassignment is completed, run “kafka-preferred-replica-election.sh” tool to complete the balancing.

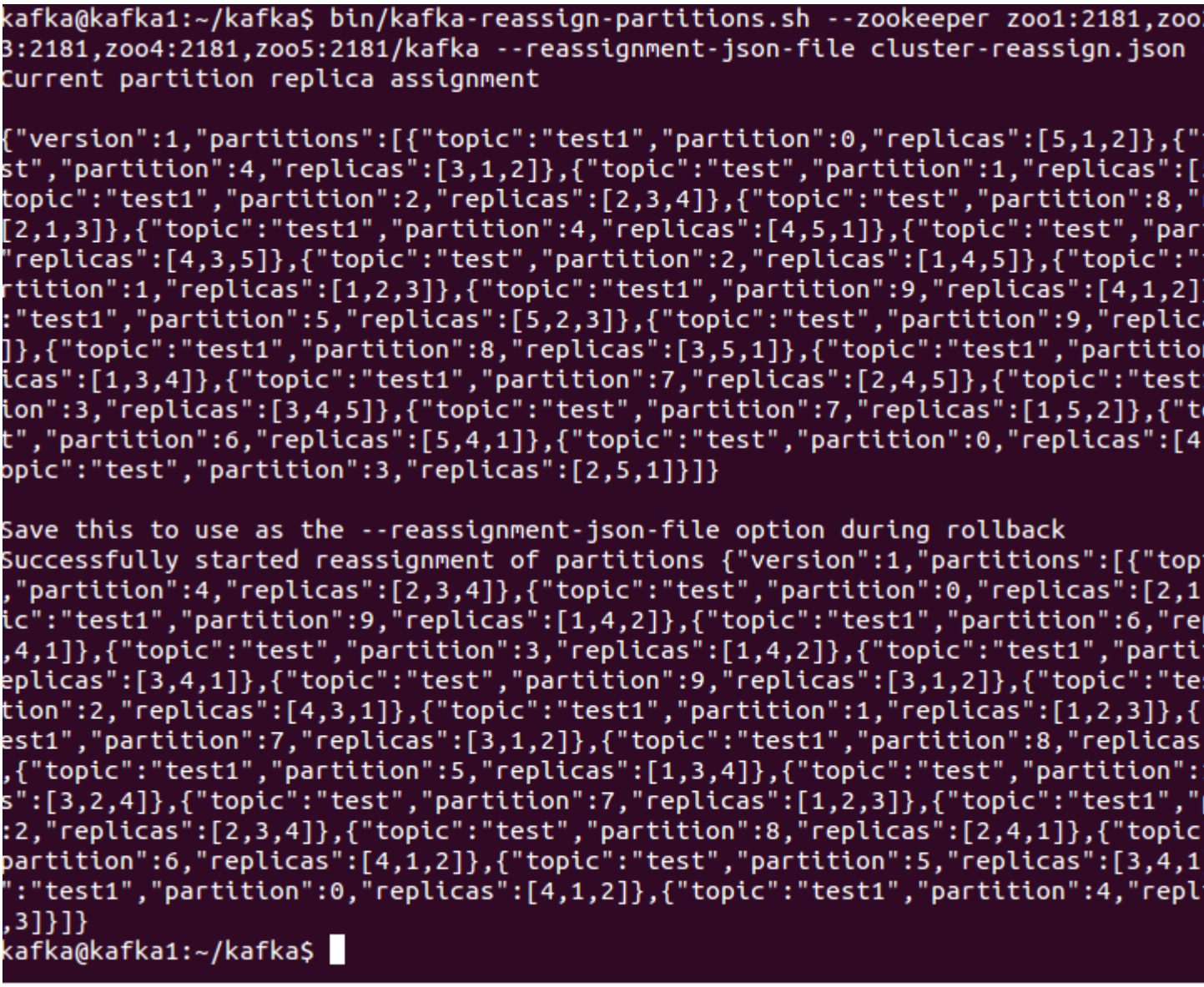
Let’s us assume that we want to decrease the number of nodes in the cluster and bring down one of the nodes (broker 5). Once the broker is terminated, we generate a candidate assignment config using the partition reassignment tool which shows current and proposed replica assignments as JSONs.

```
kafka@kafka1:~/kafka$ bin/kafka-reassign-partitions.sh --zookeeper zoo1:2181/kafka --topic
s-to-move topics-to-move.json --broker-list "1,2,3,4" --generate
Current partition replica assignment

{"version":1,"partitions":[{"topic":"test1","partition":0,"replicas":[5,1,2]},{"topic":"te
st","partition":4,"replicas":[3,1,2]},{"topic":"test","partition":1,"replicas":[5,3,4]},{"
topic":"test1","partition":2,"replicas":[2,3,4]},{"topic":"test","partition":8,"replicas":
[2,1,3]},{"topic":"test1","partition":4,"replicas":[4,5,1]},{"topic":"test","partition":5,
"replicas":[4,3,5]},{"topic":"test","partition":2,"replicas":[1,4,5]},{"topic":"test1","pa
rtition":1,"replicas":[1,2,3]},{"topic":"test1","partition":9,"replicas":[4,1,2]},{"topic"
:"test1","partition":5,"replicas":[5,2,3]},{"topic":"test","partition":9,"replicas":[3,2,4
]},{"topic":"test1","partition":8,"replicas":[3,5,1]},{"topic":"test1","partition":6,"repl
icas":[1,3,4]},{"topic":"test1","partition":7,"replicas":[2,4,5]},{"topic":"test1","partit
ion":3,"replicas":[3,4,5]},{"topic":"test","partition":7,"replicas":[1,5,2]},{"topic":"tes
t","partition":6,"replicas":[5,4,1]},{"topic":"test","partition":0,"replicas":[4,2,3]},{"t
opic":"test","partition":3,"replicas":[2,5,1]}]}
Proposed partition reassignment configuration

{"version":1,"partitions":[{"topic":"test1","partition":0,"replicas":[4,1,2]},{"topic":"te
st","partition":4,"replicas":[2,3,4]},{"topic":"test","partition":1,"replicas":[3,2,4]},{"
topic":"test1","partition":2,"replicas":[2,3,4]},{"topic":"test","partition":8,"replicas":
[2,4,1]},{"topic":"test1","partition":4,"replicas":[4,2,3]},{"topic":"test","partition":5,
"replicas":[3,4,1]},{"topic":"test","partition":2,"replicas":[4,3,1]},{"topic":"test1","pa
rtition":1,"replicas":[1,2,3]},{"topic":"test1","partition":9,"replicas":[1,4,2]},{"topic"
:"test","partition":9,"replicas":[3,1,2]},{"topic":"test1","partition":5,"replicas":[1,3,4
]},{"topic":"test1","partition":8,"replicas":[4,3,1]},{"topic":"test1","partition":6,"repl
icas":[2,4,1]},{"topic":"test1","partition":7,"replicas":[3,1,2]},{"topic":"test1","partit
ion":3,"replicas":[3,4,1]},{"topic":"test","partition":7,"replicas":[1,2,3]},{"topic":"tes
t","partition":6,"replicas":[4,1,2]},{"topic":"test","partition":0,"replicas":[2,1,3]},{"t
opic":"test","partition":3,"replicas":[1,4,2]}]}
```

Copy the proposed reassignment configuration to a JSON file and execute the partition reassignment tool. This updates the metadata and then starts to move data around to balance the

load.

```
kafka@kafka1:~/kafka$ cat cluster-reassign.json
{"version":1,"partitions":[{"topic":"test1","partition":0,"replicas":[4,1,2]},{"topic":"te
st","partition":4,"replicas":[2,3,4]},{"topic":"test","partition":1,"replicas":[3,2,4]},{"
topic":"test1","partition":2,"replicas":[2,3,4]},{"topic":"test","partition":8,"replicas":
[2,4,1]},{"topic":"test1","partition":4,"replicas":[4,2,3]},{"topic":"test","partition":5,
"replicas":[3,4,1]},{"topic":"test","partition":2,"replicas":[4,3,1]},{"topic":"test1","pa
rtition":1,"replicas":[1,2,3]},{"topic":"test1","partition":9,"replicas":[1,4,2]},{"topic"
:"test","partition":9,"replicas":[3,1,2]},{"topic":"test1","partition":5,"replicas":[1,3,4
]},{"topic":"test1","partition":8,"replicas":[4,3,1]},{"topic":"test1","partition":6,"repl
icas":[2,4,1]},{"topic":"test1","partition":7,"replicas":[3,1,2]},{"topic":"test1","partit
ion":3,"replicas":[3,4,1]},{"topic":"test","partition":7,"replicas":[1,2,3]},{"topic":"tes
t","partition":6,"replicas":[4,1,2]},{"topic":"test","partition":0,"replicas":[2,1,3]},{"t
opic":"test","partition":3,"replicas":[1,4,2]}]}
kafka@kafka1:~/kafka$
```

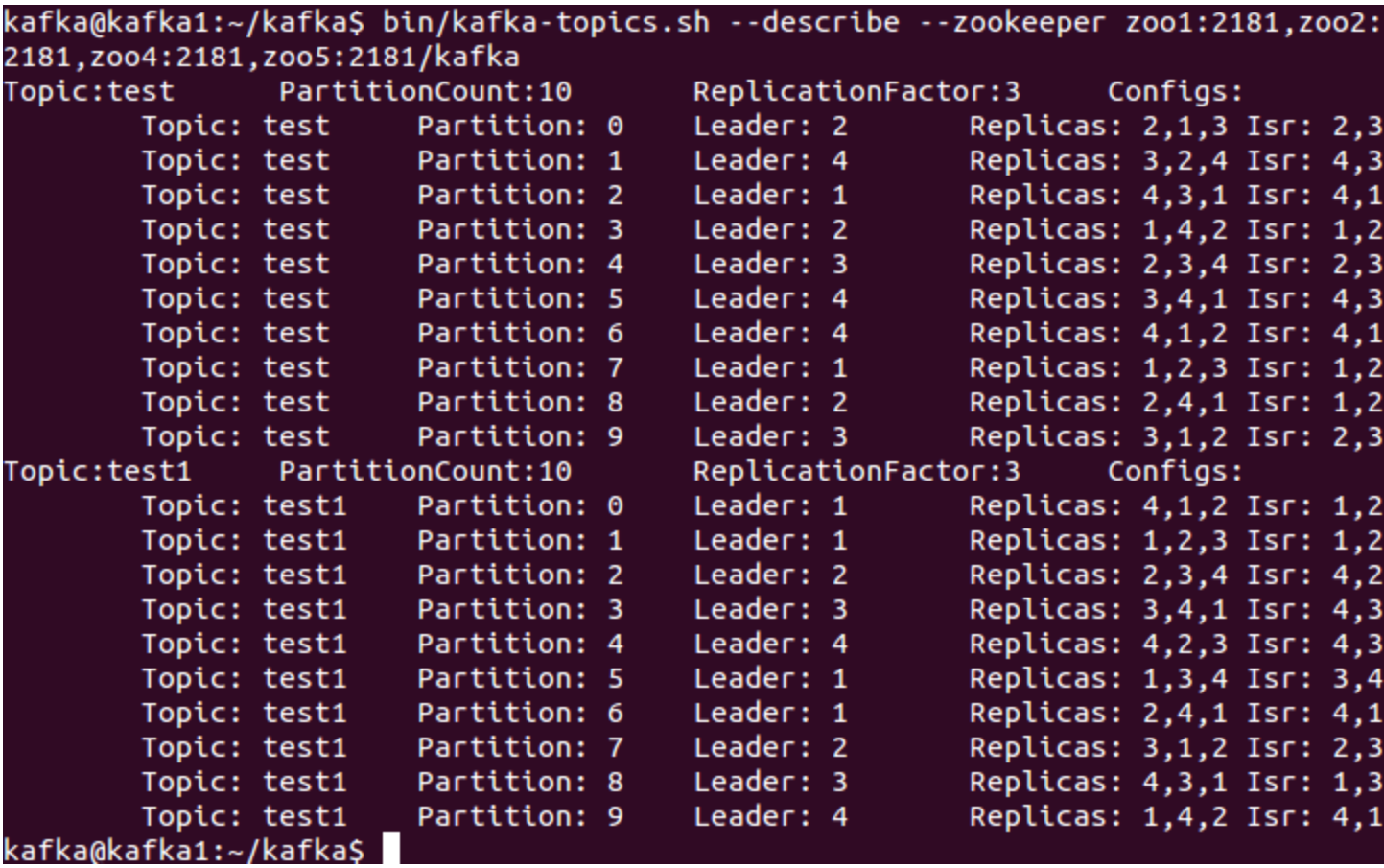

kafka@kafka1:~/kafka$ bin/kafka-reassign-partitions.sh --zookeeper zoo1:2181,zoo
3:2181,zoo4:2181,zoo5:2181/kafka --reassignment-json-file cluster-reassign.json
Current partition replica assignment
{"version":1,"partitions":[{"topic":"test1","partition":0,"replicas":[5,1,2]},{"
st","partition":4,"replicas":[3,1,2]},{"topic":"test","partition":1,"replicas":[
topic":"test1","partition":2,"replicas":[2,3,4]},{"topic":"test","partition":8,"
[2,1,3]},{"topic":"test1","partition":4,"replicas":[4,5,1]},{"topic":"test","par
"replicas":[4,3,5]},{"topic":"test","partition":2,"replicas":[1,4,5]},{"topic":"
rtition":1,"replicas":[1,2,3]},{"topic":"test1","partition":9,"replicas":[4,1,2]
:"test1","partition":5,"replicas":[5,2,3]},{"topic":"test","partition":9,"replic
]},{"topic":"test1","partition":8,"replicas":[3,5,1]},{"topic":"test1","partitio
icas":[1,3,4]},{"topic":"test1","partition":7,"replicas":[2,4,5]},{"topic":"test
ion":3,"replicas":[3,4,5]},{"topic":"test","partition":7,"replicas":[1,5,2]},{"t
t","partition":6,"replicas":[5,4,1]},{"topic":"test","partition":0,"replicas":[4
opic":"test","partition":3,"replicas":[2,5,1]}]}
Save this to use as the --reassignment-json-file option during rollback
Successfully started reassignment of partitions {"version":1,"partitions":[{"top
,"partition":4,"replicas":[2,3,4]},{"topic":"test","partition":0,"replicas":[2,1
ic":"test1","partition":9,"replicas":[1,4,2]},{"topic":"test1","partition":6,"re
,4,1]},{"topic":"test","partition":3,"replicas":[1,4,2]},{"topic":"test1","parti
eplicas":[3,4,1]},{"topic":"test","partition":9,"replicas":[3,1,2]},{"topic":"te
tion":2,"replicas":[4,3,1]},{"topic":"test1","partition":1,"replicas":[1,2,3]},{
est1","partition":7,"replicas":[3,1,2]},{"topic":"test1","partition":8,"replicas
,{"topic":"test1","partition":5,"replicas":[1,3,4]},{"topic":"test","partition":
s":[3,2,4]},{"topic":"test","partition":7,"replicas":[1,2,3]},{"topic":"test1","
:2,"replicas":[2,3,4]},{"topic":"test","partition":8,"replicas":[2,4,1]},{"topic
partition":6,"replicas":[4,1,2]},{"topic":"test","partition":5,"replicas":[3,4,1
":"test1","partition":0,"replicas":[4,1,2]},{"topic":"test1","partition":4,"repl
,3]}]}
kafka@kafka1:~/kafka$

Next we verify that the rebalance/reassignment is successful.

```
kafka@kafka1:~/kafka$ bin/kafka-reassign-partitions.sh --zookeeper zoo1:2181,zoo
3:2181,zoo4:2181,zoo5:2181/kafka --reassignment-json-file cluster-reassign.json
Status of partition reassignment:
Reassignment of partition [test,4] completed successfully
Reassignment of partition [test,0] completed successfully
Reassignment of partition [test1,9] completed successfully
Reassignment of partition [test1,6] completed successfully
Reassignment of partition [test,3] completed successfully
Reassignment of partition [test1,3] completed successfully
Reassignment of partition [test,9] completed successfully
Reassignment of partition [test,2] completed successfully
Reassignment of partition [test1,1] completed successfully
Reassignment of partition [test1,7] completed successfully
Reassignment of partition [test1,8] completed successfully
Reassignment of partition [test1,5] completed successfully
Reassignment of partition [test,1] completed successfully
Reassignment of partition [test,7] completed successfully
Reassignment of partition [test1,2] completed successfully
Reassignment of partition [test,8] completed successfully
Reassignment of partition [test,6] completed successfully
Reassignment of partition [test,5] completed successfully
Reassignment of partition [test1,0] completed successfully
Reassignment of partition [test1,4] completed successfully
kafka@kafka1:~/kafka$
```

Once the reassignment is successful for all partitions, we run the preferred replica election tool to balance the topics and then run ‘describe topic’ to check balancing of topics.

```
kafka@kafka1:~/kafka$ bin/kafka-preferred-replica-election.sh --zookeeper zoo1:2
Successfully started preferred replica election for partitions Set([test,4], [te
st1,9], [test1,6], [test,3], [test1,3], [test,9], [test,2], [test1,1], [test1,7]
], [test1,5], [test,1], [test,7], [test1,2], [test,8], [test,6], [test,5], [test
t1,4])
```



Now we can see that the topics (along with leaders and replicas) are all balanced.