Add/Remove nodes

Add new node

To add a node: install kafka software, run kafka-start script with Zookeeper-connect: IP (this will be enough to make the new broker node visible in the cluster)

But the new node has no data, only new topics will be distributed as well on the new broker, the existing data will not be rebalanced automatically. In order to assign partitions as well to the new node:

- Use **kafka-reassign-partitions.sh** tool to generate partition assignments. **It takes the topic list and the broker list as input, and produces the assignment plan in JSON format**. The output indicates as well the new partition leader for each partition.
- To run this plan, we need to use the **kafka-reassign-partitions.sh tool with the --execute command**. It takes the generated reassignment.json file as input. Once the reassignment is finished, your partitions have been redistributed over the cluster.
- To check the partition reassignment, you can either use:
    - The kafka-reassign-partitions.sh tool with the --verify command.
    - The kafka-topic.sh tool with the --describe command.

Kafka Reassign Partitions tool

This tool provides substantial control over partitions in a Kafka cluster. **It is mainly used to balance storage loads across brokers through the following reassignment actions**:

- Change the ordering of the partition assignment list. Used to control leader imbalances between brokers.
- Reassign partitions from one broker to another. Used to expand existing clusters.
- Reassign partitions between log directories across multiple brokers. Used to resolve storage load imbalance across multiple brokers.

The tool uses two JSON files for input. Both of these are created by the user. The two files are the following: Topics-to-Move JSON, Reassignment Configuration JSON (optional)

When the **kafka-reassign-partitions tool is executed with the --generate option**, it generates a proposed configuration which can be fine-tuned and saved as a JSON file. The file created this way is the reassignment configuration JSON. To generate a proposal, the tool requires a topics-to-move file as input.

Remove a broker node

When you plan to remove a broker from the cluster, first you should rearrange the partitions according to the new cluster layout.

kafka-topic.sh --list to get the topic list and write a topics.json

kafka-reassign-partitions.sh --generate to generate an assignment plan assignment.json excluding the node to remove

kafka-reassign-partitions.sh --execute to run the assignment plan

kafka-reassign-partitions.sh --verify to check whether the assignment plan is applied

Stop the broker and remove it

Replace a broker with another one

To replace a node by another one, you don’t need to use the above scenarios because you can keep the same partition assignment. All you have to do is:

Stop the old node

Give the new node the same Id as the old one

Start the new node

Broker Outage

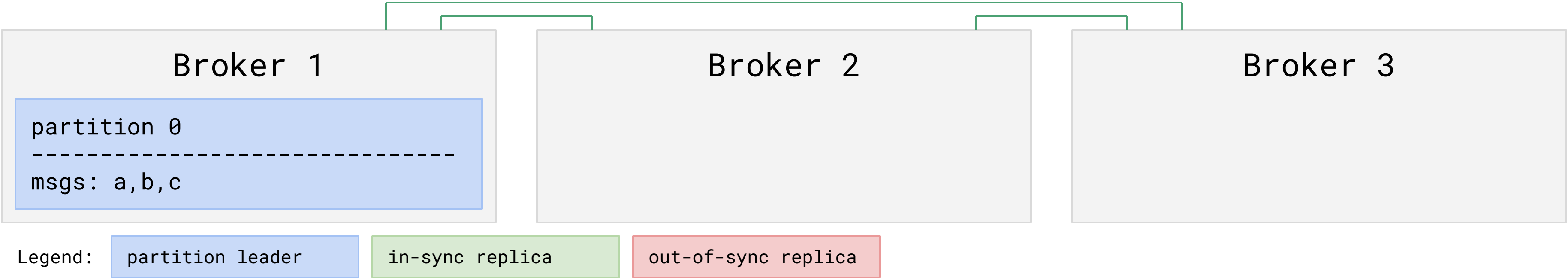
During a broker outage, all partition replicas on the broker become unavailable, so the affected partitions' availability is determined by the existence and status of their other replicas.

Important:

- **When a broker fails only its partition leaders will be reassigned to other brokers**
- The broker info is not immediately removed from the cluster metadata, if the broker had a temporary outage and comes back using the same broker-id a check will be done against its data and the cluster will try to re-assign the same partitions - but only after the broker is insync

Configuring to tolerate broker outages

- When a broker fails, all its replicas will become unavailable
  - If a partition loses its replica leader, a new leader will be chosen from the ISR
  - If the ISR is empty, we can either be unavailable or elect out-of-sync replica to be leader
- When the broker is back, its partitions will:
  - Catch up on missing data
  - If the node becomes insync then it becomes leader (if it was leader to begin with)



Broker 1

partition 0

----

msgs: a,b,c

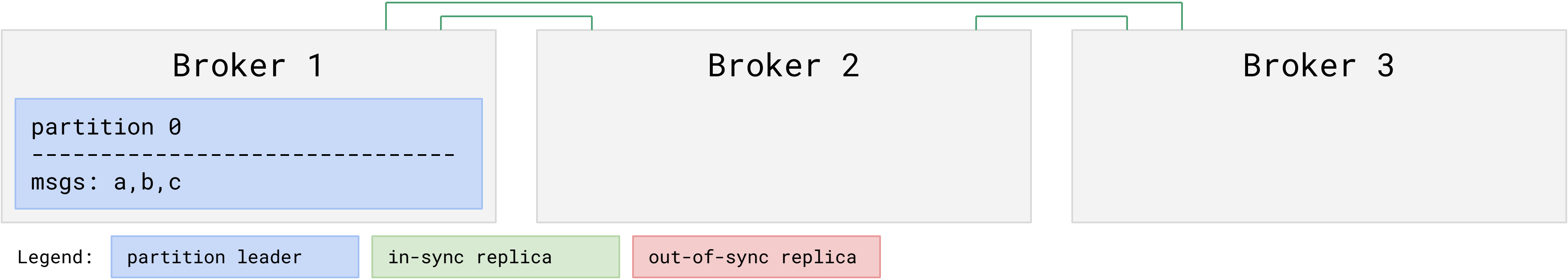
Broker 2

Broker 3

Legend: partition leader | in-sync replica | out-of-sync replica

Configuring to tolerate broker outages

- In most cases, we want to
  - Ensure consistency in our messages while
  - Maximizing availability (not solving CAP theorem)
- Three important settings to consider
  - **replication-factor**
  - **minimum size of ISR**
  - **producer acknowledgements requirement**



Backup

Tools for scaling the cluster

https://github.com/yahoo/CMAK - Kafka Manager - open source

Confluent Enterprise - license based