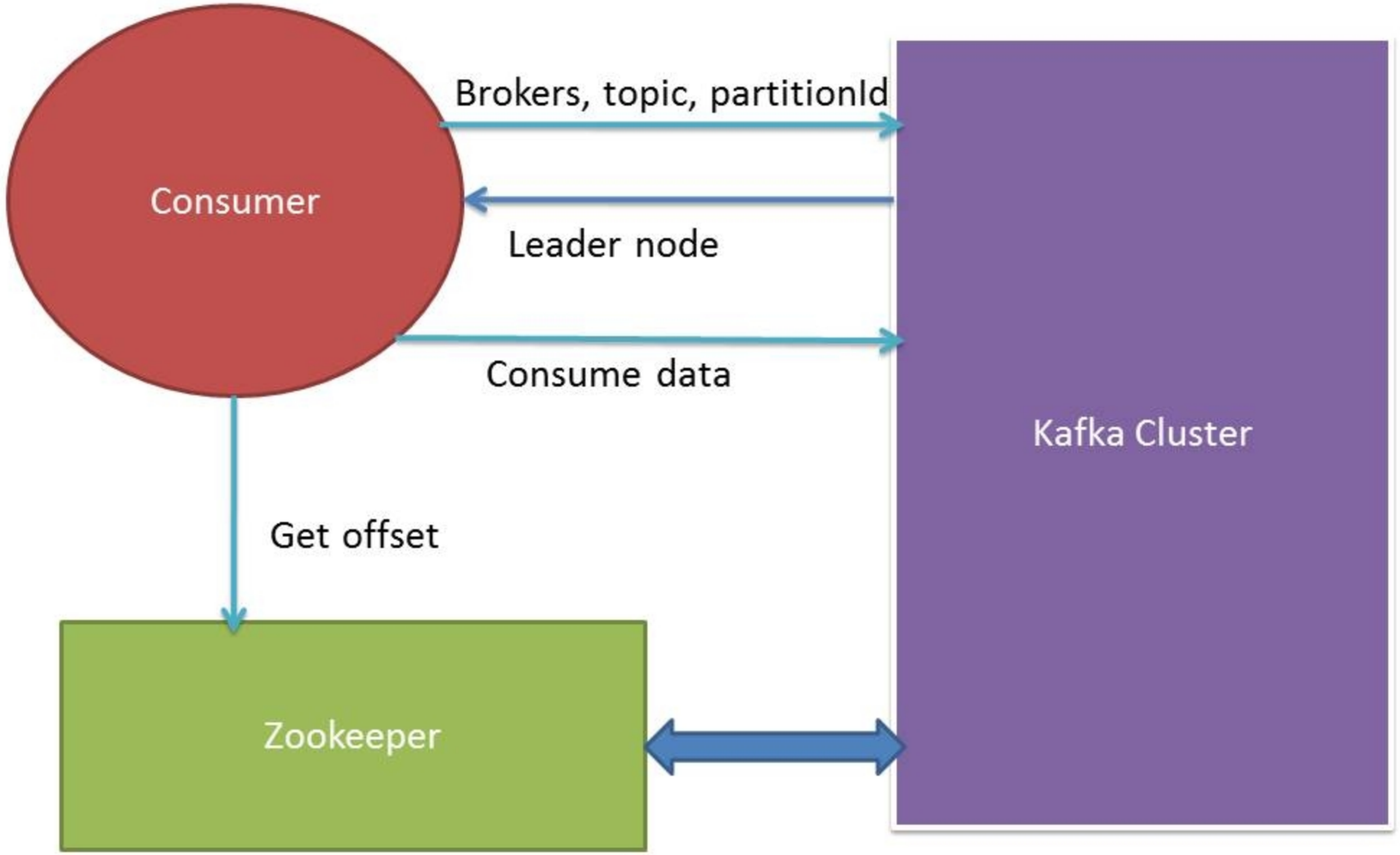
Welcome to the World of Distributed Messaging Queue

Low level consumer Hands-On



1. Consumer send the request to find the leader partition of a broker (request contains the broker list, topicName, partitionId)
2. Return the leader node
3. Get the offset from which we need to start the data read
4. Start data consuming
5. Re-elect the leader, if leader node goes down

Data Retention

- The Kafka cluster retains all published messages whether or not they have been consumed for a configurable period of time.
- For example if the log retention is set to two days, then for the two days after a message is published it is available for consumption, after which it will be discarded to free up space.
- Kafka's performance is effectively constant with respect to data size so retaining lots of data is not a problem.

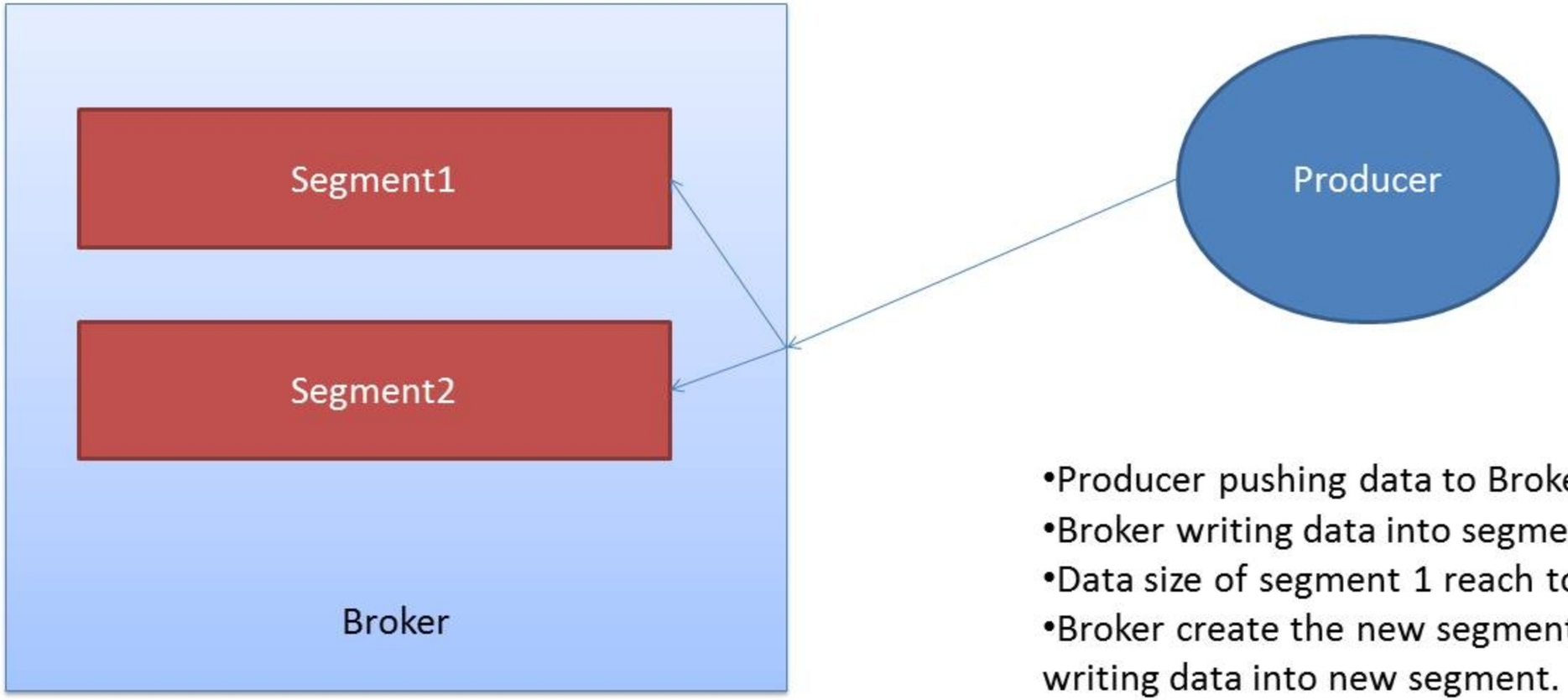
Data Retention

- The property **log.retention.{minutes,hours}** define the amount of time to keep a log segment before it is deleted, i.e. the default data retention window for all topics. The default value of this property is 7 days.
- The property **log.retention.bytes** define the amount of data to retain in the log for each topic-partitions. Note that this is the limit per-partition so multiply by the number of partitions to get the total data retained for the topic. The default value of this property is -1.
- Also note that if both **log.retention.hours** and **log.retention.bytes** are both set we delete a gsegment when either limit is exceeded.
- We can overwrite this property by setting **retention.bytes** and **retention.ms** properties at the time of topic creation.
- The propety **log.retention.check.interval.ms** define the period with which we check whether any log segment is eligible for deletion to meet the retention policies. The default value of this property is 5 minutes.

Log Segment

- The property **log.segment.bytes** define the log for a topic partition is stored as a directory of segment files. This setting controls the size to which a segment file will grow before a new segment is rolled over in the log. The default value is 1GB.
- The property **log.roll.hours** define the setting will force Kafka to roll a new log segment even if the log.segment.bytes size has not been reached. The defaulr value is 168 hours.

Log Segment



- Producer pushing data to Broker.
- Broker writing data into segment 1.
- Data size of segment 1 reach to 1 GB.
- Broker create the new segment and start writing data into new segment.

Log Segment and Data Retention

- Kafka doesn’t delete single message but delete all the records belong to one segment in one go.
- It only mark the data deleted (soft delete).
  - Data inserted before this offset is marked as deleted.
- Why it does not delete the single record?
  - Deleting a single record from a file is very performance incentive task

Assignment

- Create a three nodes kafka cluster.
- Let's assume the file contains data of four countries india, usa, uk and chine.
  - Log file must contains following four fields
  - Tweet text, country, time, username
- Read the data from multiple files, convert the record into Map<String,Object> and pushed into Kafka using Sync producer.
- Write a Map encoder/decoder to convert the Map to bytes and bytes to Map.
- Run the producer on machine other then brokers.
- Create a topic having four partitions and replication factor 3.
- Create a partition class to push data of India on partition 0, data of USA on partition 1 and so on.
- Consume the data from Kafka and store all the data of India on 1 file, data of USA on other file and so on.
- Run the consumer on machine other then brokers

Apache Storm

- ✓ High Distributed real time **computation** system
- ✓ Horizontally scalable
- ✓ Fault Tolerance
- ✓ Can easily be used with any programming language
- ✓ Guaranteed message processing

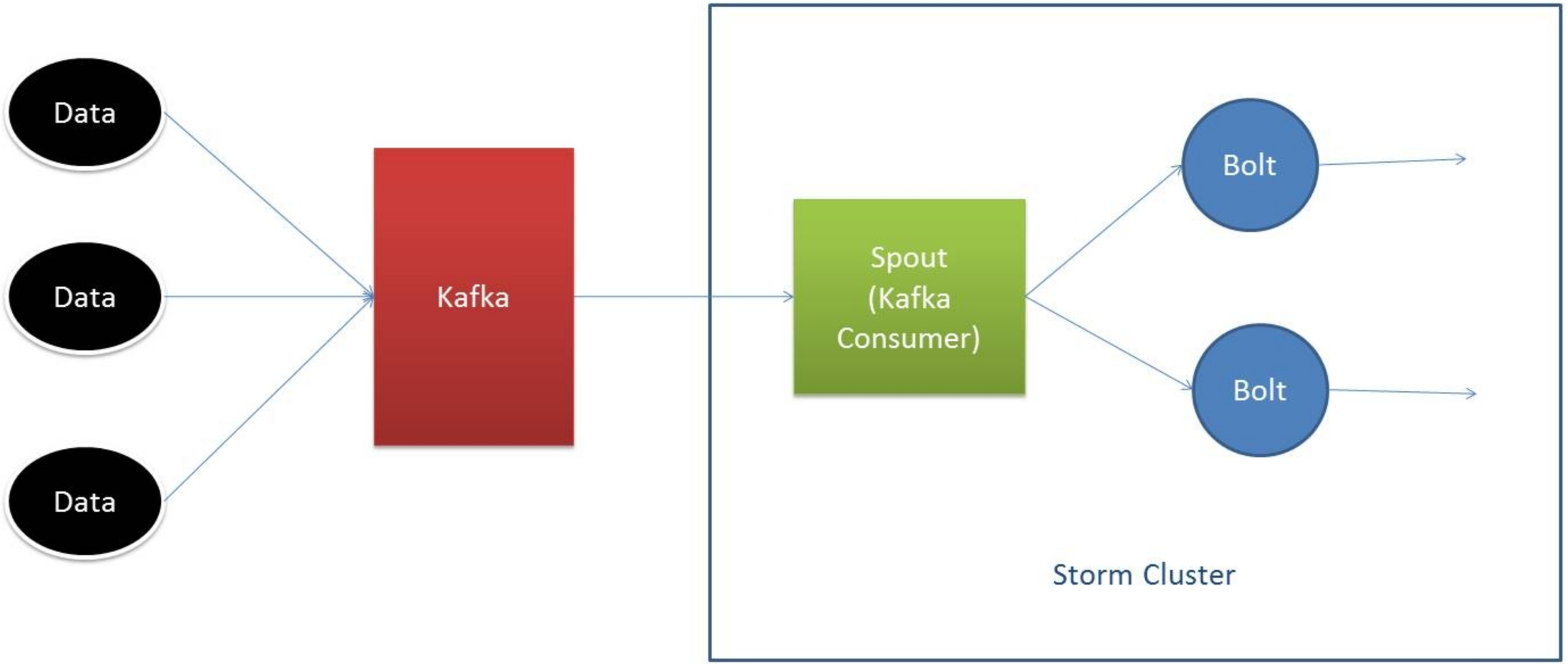
Apache Storm

- Apache 2.0 license
- Written in closure and API are exposed in Java
- Master/Slave architecture
- Rich community
- Easy to operate:
  - Storm is much easy to deploy and manage.
- Fast:
  - Storm Cluster can process **billion of records** per second

Kafka with Storm

- Consider, we have a real time app handling high volume data.
- Storm Spout doesn’t buffer/Queue the data.
- We would require external buffer/Queue for storing that data.
- Kafka is best choice for Queuing high volume data.
- Storm will read the data from Kafka and applies some required manipulation.

Kafka with Storm



DataFlair Web Services Pvt Ltd

+91-8451097879
info@data-flair.com
http://data-flair.com