# Title: Central Consumer

Central Kafka Consumer Implementation-

Every topic in Kafka has multiple partitions to enable parallel processing. We have created a consumer group that has more than one consumer process, and each consumer process traverses from one partition only. the number of consumers in a consumer group <= number of partitions in a topic. Every consumer group must have a unique name because it will commit it's metadata i.e the last read offset in the kafka-store with that unique identifier. The consumer will periodically commit (offset\_commit\_interval: 10 seconds) the last processed offset in the kafka-store to start processing from in case of failures and consumer crashes or restart.

All configurations are kept constants: topics, group\_id and the same can be extended to consume any topic from central. Generic class CentralConsumer::Runner which can run any consumer client from any Kafka cluster. By Default, all consumers mentioned in the constants will be started. If no options are provided configurations will be taken from the default. The only param that can be overridden is the number of consumers to spawn, from the command line.

Command -

rails runner lib/central\_consumer/runner.rb  
  
  
rails runner lib/central\_consumer/runner.rb --consumer "freshmarketer-activities"

Options -

--consumer = Name of consumer

--n = Number of consumer processes to spawn

Ex**- Number of Partitions** in the topic central-freshmarketer-fsales-activities-topic is **3**, so spawning that many numbers of consumers.

**How to handle errors while processing?**

The consumer is loosely coupled with the processors, such that errors while processing or other issues can be mitigated gracefully. After n number of retries (threshold = 3) we will push the failed messages as it is to a separate central topic - "freshsales-failed-messages", which will hold all the errored messages so that we don't lose any data. Messages from this topic can be consumed after we fix the issues on our side. We will be using the central producer API rather than the central-publisher gem, to post messages using an API call.

Important Links-

More on kafka - <https://kafka.apache.org/intro>

Complete info on central kafka - [https://confluence.freshworks.com/display/PLAT/Central+Details](https://confluence.freshworks.com/display/PLAT/Central+Details#CentralDetails-Burrow-Consumerlagmonitoringtool)

Monitor consumers and topic - <https://central-tools-staging.freshworksapi.com/v3/kafka/central/consumer>

Another framework that can also be used- <https://github.com/zendesk/racecar>

Central Kafka has moved from apache kafka cluster to confluent-kafka which is built on top of the former. But confluent doesn't provide any library for ruby. Since confluent-kafka doesn't provide any library for ruby, ruby-Kafka is used. So there are options and configurations that confluent-kafka has, but ruby-kafka doesn't support it.