/opt/kafka/bin/kafka-topics.sh --zookeeper 10.156.0.12:2181 --config cleanup.policy=compact --config delete.retention.ms=100 --config segment.ms=100 --config min.cleanable.dirty.ratio=0.01 --create --partitions 1 --replication-factor 3 --topic testcompaction

**config segment.ms=100 -- This configuration controls the period of time after which Kafka will force the log to roll even if the segment file isn't full to ensure that retention can delete or compact old data. Default is 7 days.**

**min.cleanable.dirty.ratio**

**This configuration controls how frequently the log compactor will attempt to clean the log (assuming log compaction is enabled). By default we will avoid cleaning a log where more than 50% of the log has been compacted.**

**delete.retention.ms**

**The amount of time to retain the tombstone markers for log compacted topics.**

/opt/kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --property parse.key=true --property key.separator=: --topic testcompaction

>p3:10

>p5:7

>p3:11

>p6:25

>p6:12

>p5:14

>p5:17

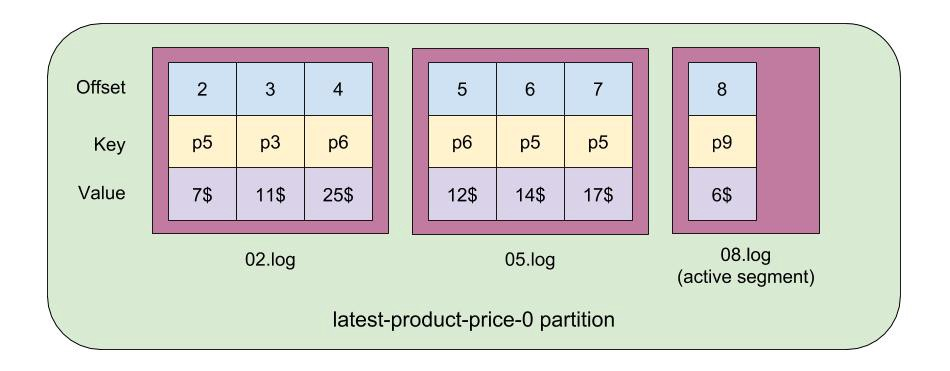
/opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --property print.key=true --property key.separator=: --from-beginning --topic testcompaction

p3:11

p6:12

p5:14

p5:17



As you can see from the above pic we have a partition log that holds 7 records residing in 3 separate segment files. The first offset of a segment is called the base offset of the segment. The segment file name is always equal to its base offset value. The last segment in the partition is called the active segment. Only the active segment of a log can receive the newly produced messages.

**/tmp/kafka-logs/testcompaction-0**$ ls -l

10485760 Oct 28 15:53 00000000000000000006.index

**00000000000000000006.log //active segment**

00000000000000000006.snapshot

00000000000000000006.timeindex

Leader-epoch-checkpoint

When does Kafka create a new segment? One option is by setting segment.bytes (default is 1GB) config during topic creation. When your segment size becomes bigger than this value, Kafka will create a new segment. Another option is by setting segment.ms. In our case at every 100ms. **Active segment will not be cleaned/compacted.**

**The reason why p5 has both 14 and 17 values - is that 17 is in the active segment and cannot be compacted. Let’s check what the active segment has inside.**

/opt/kafka/bin/kafka-run-class.sh kafka.tools.DumpLogSegments --deep-iteration --print-data-log --files /tmp/kafka-logs/testcompaction-0/00000000000000000006.log

**Output:**

Dumping /tmp/kafka-logs/testcompaction-0/00000000000000000006.log

Starting offset: 6

baseOffset: 6 lastOffset: 6 count: 1 baseSequence: -1 lastSequence: -1 producerId: -1 producerEpoch: -1 partitionLeaderEpoch: 0 isTransactional: false isControl: false position: 0 CreateTime: 1572278026568 size: 73 magic: 2 compresscodec: NONE crc: 1897573186 isvalid: true

| offset: 6 CreateTime: 1572278026568 keysize: 2 valuesize: 3 sequence: -1 headerKeys: [] **key: p5 payload: 17$**

Lets insert more data in the topic in order for the active segment to become inactive:

/opt/kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic testcompaction --property parse.key=true --property key.separator=:

p3:12

Lets see what happened on disk

**/tmp/kafka-logs/testcompaction-0**$ ls -l

A new log was created and become active. Now lets consume the topic data:

/opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic testcompaction --property print.key=true --property key.separator=: --from-beginning

p3:11$

p6:12$

p5:17$

p3:12$

P5 values were compacted.