KIP-758: Remove list of latest producer append information for broker

Status

Current state: Under Discussion

Discussion thread: here[TODO]

JIRA: here

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

Each partition in broker maintains a list of the 5 most recent appended batches as producer state for each producerId (a.k.a PID). If a producer fails to receive a successful response and retries the produce request, broker can still return the offset of the successful append from the maintained cache via `RecordMetadata`. The maintained cache per producerId per partition in broker instance consumes lots of memory which causes OOM in production and it is not necessary to always provide client the offset of already appended batch in response. Therefore, this KIP proposes to only maintain one latest appended batch write so that both memory usage for producer state and the complexity to manage it are reduced on the broker side.

Public Interfaces

Existing producer configuration `max.in.flight.requests.per.connection` with default value of 5 will be obsolete eventually. We will define a duplicate sequence range threshold in broker side as `max.in.flight.sequence.number.per.connection`. Broker will treat all sequence number smaller than (accounting overflow) "latest appended sequence" but within `max.in.flight.sequence.number.per.connection` as already appended and duplicate from producer request. All sequence number outside of this range will be treated as future sequence number which haven't been appended or allocated from producer request. We will define this configuration default value large enough so that producer won't be able to have more than this amount of inflight sequence numbers.

Producer needs keep a same value of `max.in.flight.sequence.number.per.connection` decided by broker. When producer starts up, it sends API request to broker. Broker will piggyback the value of `max.in.flight.sequence.number.per.connection` on API response to producer.

Proposed Changes

The broker will only maintain the latest appended batch's sequence/offset for each producerId instead of 5 recent appended batches. The broker will only accept incoming batch request in sequence order so that it can guarantee any sequence number equal or less than the latest sequence will be durable.

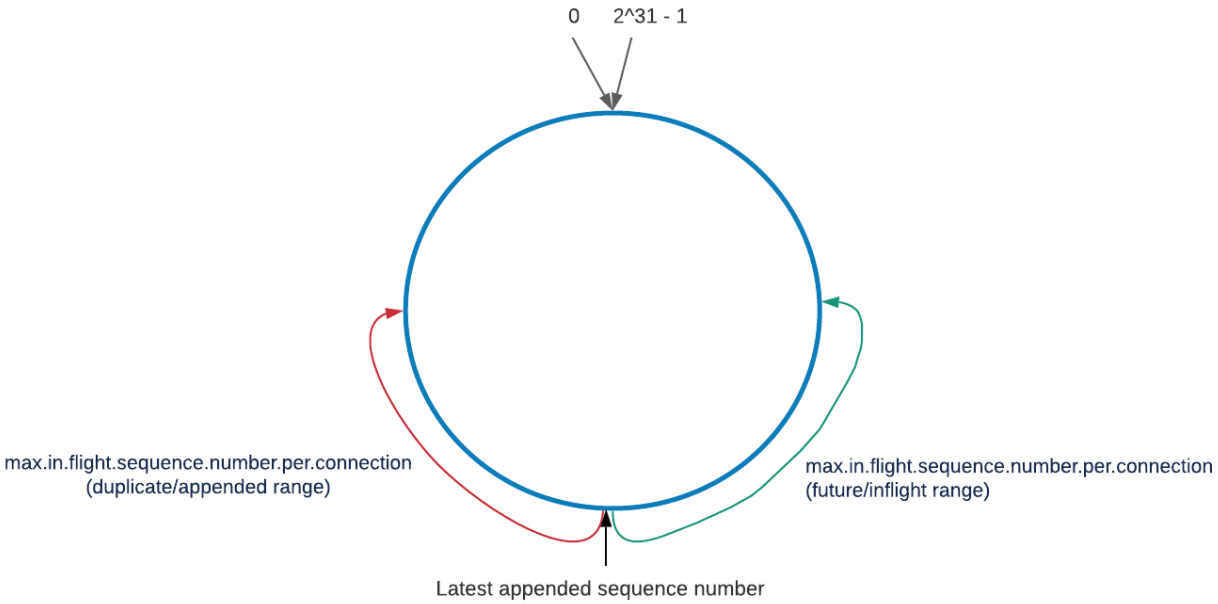
When the sequence number reaches Int.MaxValue, client can wraparound starting from 0 again.

On broker side,

- Brokers will accept batch with first sequence number one strictly higher than (accounting for overflow) last sequence number of latest appended batch.
- Brokers will return DUPLICATE_SEQUENCE_NUMBER for any sequence that is within `max.in.flight.sequence.number.per.connection` range of the latest sequence number (accounting for overflow). In this case, we won't return offset of the appended batch except for duplicate batch with latest appended one.
- Brokers will return OUT_OF_ORDER_SEQUENCE for any sequence that is outside `max.in.flight.sequence.number.per.connection` range of the latest sequence number.

On Producer side,

- When clients receive DUPLICATE_SEQUENCE_NUMBER, it will discard it and move on as before.
- When clients receive OUT_OF_ORDER_SEQUENCE, it will handle as before - retry the send request.
- Clients will throttles write when the number of inflight sequences reaches `max.in.flight.sequence.number.per.connection`



When producer and broker agree on the max.in.flight.sequence.number.per.connection value and producer throttles write accordingly, broker can make sure within the red range below "latest appended sequence number" are duplicate and within the green range above "latest appended sequence number" are future sequences not appended yet. We also need to make sure the red range and green range don't overlap each other, so the maximum value for max.in.flight.sequence.number.per.connection cannot exceed 2^30. On the other hand, the threshold of max.in.flight.sequence.number.per.connection should never throttle the customer workload at all in real world. Based on different factors like max payload size, max socket buffer size, compaction rate and etc. we come up with a default value of 10 million as max possible sequence number inflight that should not hit in extreme cases.

Compatibility, Deprecation, and Migration Plan

With this proposal, restriction on max.in.flight.requests.per.connection can be removed and we don't depend on number of inflight request batches to throttle write. Broker won't provide offset along with a DUPLICATE_SEQUENCE_NUMBER error response except for duplicate with latest appended batch. This feature needs document that the user won't expect the offset is always reported in `RecordMetadata` response. We will obsolete max.in.flight.requests.per.connection and introduce new configuration max.in.flight.sequence.number.per.connection in both broker & producer to control the inflight writes.

Mechanism may be needed to tell whether a broker supports the new duplicate detection logic, like bumping the Produce API version so that a client can tell whether the broker it talks to is an old version (a.k.a. limit to 5 inflight sequence number) or new version (has max.in.flight.sequence.number.per.connection to set back in producer) upon communication.

We will also deliver/migrate this feature in phases:

1. Deliver only broker side change: replace 5 appended batches cache with only one latest appended batch cache. Introduce broker side configuration max.in.flight.sequence.number.per.connection with default value of 10 million as duplicate check threshold. No producer side changes at all: max.in.flight.requests.per.connection as default 5 is still enforced. This makes broker in a safer place that producer won't generate more than max.in.flight.sequence.number.per.connection sequence numbers due to number of inflight batches limitation.
2. Deliver producer side changes: introduce producer side configuration max.in.flight.sequence.number.per.connection and bump producer API version for passing configuration value from broker to producer. Enforce producer workload throttling based on max.in.flight.sequence.number.per.connection. Obsolete producer side configuration max.in.flight.requests.per.connection.

Rejected Alternatives

There is no rejected alternatives. However, we thought about bumping producer epoch to solve the sequence ID wraparound issue so that we don't need to take an arbitrary large enough threshold number like 10 million. There are couple of concerns with this approach:

1. transactions with sequence number just across the Int.MaxValue will be aborted for new epoch bump up. This should be fine since we only abort at most one transaction per epoch life time.
2. Sequence number is a concept within topic partition but epoch is a concept within a producer. If a producer serves tons of topic partitions, the chance of sequence number wraparound for any topic partition will increase. In this case we may see relatively higher frequency of epoch bump and transaction abort. We need some estimation before jumping into this approach.
3. Old clients (non-idempotent producer which cannot bump epoch OR transactional producer used on brokers which don't support epoch bump) cannot use epoch bump mechanism to solve sequence number wraparound, so the proposal in this KIP is still needed anyway.

Based on above concerns, we decided to start with the default wraparound solution.