KIP-994: Minor Enhancements to ListTransactions and DescribeTransactions APIs

Status

Current state: *Accepted*

Discussion thread: *here* *(Vote thread: here)*

JIRA: *here*

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

Motivation

KIP-664 introduced tooling to detect, analyze and resolve hanging Kafka transactions. We propose to enhance this tooling to serve some operationally useful scenarios like reducing the size of response for `ListTransactionsRequest` and allowing users to build some observability / automation around long running transactions.

⚠ Unable to render Jira issues macro, execution error. reported a bug where tooling reports incorrect run duration for completed transactions. This gives us an opportunity to add information about the time of last state change which can be useful to analyze stale transactions.

Public Interfaces

- **`ListTransactionsRequest`**

Add a new field, `DurationFilter` to the `ListTransactionsRequest` and bump the version to 1

```
{
  "apiKey": 66,
  "type": "request",
  "listeners": ["zkBroker", "broker"],
  "name": "ListTransactionsRequest",
  "validVersions": "0-1",
  "flexibleVersions": "0+",
  "fields": [
    { "name": "StateFilters", "type": "[]string", "versions": "0+",
      "about": "The transaction states to filter by: if empty, all transactions are returned; if non-empty,
then only transactions matching one of the filtered states will be returned"
    },
    { "name": "ProducerIdFilters", "type": "[]int64", "versions": "0+", "entityType": "producerId",
      "about": "The producerIds to filter by: if empty, all transactions will be returned; if non-empty, only
transactions which match one of the filtered producerIds will be returned"
    },
        // Add a DurationFilter field
        { "name": "DurationFilter", "type": "long", "versions": "1+",
          "about": "Return transactions running longer than this time duration, specified in milliseconds"
        }
  ]
}
```

Add `DurationFilter` to the `ListTransactionsOptions` class used by `AdminClient` to pass on filters to the Kafka broker.

```
@InterfaceStability.Evolving
public class ListTransactionsOptions extends AbstractOptions<ListTransactionsOptions> {
...
// return transactions open for more than this time duration specified in milliseconds
Duration durationFilter;

public ListTransactionsOptions durationFilter(Duration timeDuration) {
        this.durationFilter = timeDuration;
        return this;
}

public Duration durationFilter() {
        return this.durationFilter;
}
...
}
```

- **`ListTransactionsResponse`**

`Version will be bumped to 1 to match request. No changes otherwise.`

- **`DescribeTransactionsResponse`**

Add a new field, `TransactionLastUpdateTimeMs to DescribeTransactionsResponse and bump the version to 1`.

Broker will populate this field from `txnLastUpdateTimestamp` contained at `TransactionMetadata.` This field is updated at the broker every time the transaction's state changes.

```
{
  "apiKey": 65,
  "type": "response",
  "name": "DescribeTransactionsResponse",
  "validVersions": "0-1",
  "flexibleVersions": "0+",
  "fields": [
      { "name": "ThrottleTimeMs", "type": "int32", "versions": "0+",
        "about": "The duration in milliseconds for which the request was throttled due to a quota violation, or
zero if the request did not violate any quota." },
      { "name": "TransactionStates", "type": "[]TransactionState", "versions": "0+", "fields": [
        { "name": "ErrorCode", "type": "int16", "versions": "0+" },
        { "name": "TransactionalId", "type": "string", "versions": "0+", "entityType": "transactionalId" },
        { "name": "TransactionState", "type": "string", "versions": "0+" },
        { "name": "TransactionTimeoutMs", "type": "int32", "versions": "0+" },
        { "name": "TransactionStartTimeMs", "type": "int64", "versions": "0+" },
                // New field to indicate the timestamp when transaction state was last changed
        { "name": "TransactionLastUpdateTimeMs", "type": "int64", "versions": "1+" },
        { "name": "ProducerId", "type": "int64", "versions": "0+", "entityType": "producerId" },
        { "name": "ProducerEpoch", "type": "int16", "versions": "0+" },
        { "name": "Topics", "type": "[]TopicData", "versions": "0+",
          "about": "The set of partitions included in the current transaction (if active). When a transaction
is preparing to commit or abort, this will include only partitions which do not have markers.",
          "fields": [
            { "name": "Topic", "type": "string", "versions": "0+", "entityType": "topicName", "mapKey": true },
            { "name": "Partitions", "type": "[]int32", "versions": "0+" }
          ]
        }
      ]}
  ]
}
```

Add `transactionLastUpdateTimeMs` to `TransactionDescription` class. This object is used to build `APIResult` at `org.apache.kafka.clients.admin.internals.DescribeTransactionsHandler#handleResponse` from `DescribeTransactionsResponse` .

We will also add an overloaded public constructor to this class to incorporate the value for `transactionLastUpdateTimeMs` field.

```
package org.apache.kafka.clients.admin;

public class TransactionDescription {
    ...
    private final OptionalLong transactionLastUpdateTimeMs;
        ...

    public TransactionDescription(
        int coordinatorId,
        TransactionState state,
        long producerId,
        int producerEpoch,
        long transactionTimeoutMs,
        OptionalLong transactionStartTimeMs,
        Set<TopicPartition> topicPartitions
    ) {
                new TransactionDescription(coordinatorId, state, producerId, producerEpoch,
transactionTimeoutMs, transactionStartTimeMs, OptionalLong.empty(), topicPartitions);
    }

    // new overloaded public constructor
    public TransactionDescription(
        int coordinatorId,
        TransactionState state,
        long producerId,
        int producerEpoch,
        long transactionTimeoutMs,
        OptionalLong transactionStartTimeMs,
                OptionalLong transactionLastUpdateTimeMs,
        Set<TopicPartition> topicPartitions
    ) {
                ...
        this.transactionLastUpdateTimeMs = transactionLastUpdateTimeMs;
        ...
    }
        ...
         public OptionalLong transactionLastUpdateTimeMs() {
        return transactionLastUpdateTimeMs;
    }
        ...
}
```

Fixing the TransactionsCommand tool output

`TransactionDescription` is further utilized at `org.apache.kafka.tools.TransactionsCommand.DescribeTransactionsCommand#execute` to build a printable description of the transaction. This method will be changed to calculate `transactionDuration` as a difference between `transactionStartTime` and `transactionLastUpdateTimeMs`, `if state == COMPLETE_COMMIT || state == COMPLETE_ABORT`

- **DescribeTransactionsRequest**

`Version will be bumped to 1 to match response. No changes otherwise.`

Command Line Tool Changes

`kafka-transactions.sh --list` command will have a new option, `--durationFilter` to return transactions running longer than this time duration, specified in milli seconds.

`kafka-transactions.sh --describe` command will change the way it prints the value for transaction duration for completed transactions.

If TransactionLastUpdateTimeMs field is present in DescribeTransactionsResponse, transaction duration (for completed transactions) will be printed as the difference of TransactionLastUpdateTimeMs and transactionStartTime. On the other hand, if this field is not present, transaction duration value cannot be determined correctly (for completed transactions) and we will print -1.

Proposed Changes

We propose to add a `Duration` filter to `ListTransactionsOptions` in order to list only the transactions older than a certain time duration.

We also propose to add `lastUpdateTimestamp` to the `DescribeTransactionsResponse.` This bit can be used to calculate the correct run duration for a completed transaction and can be helpful in analyzing stale transactions.

Compatibility, Deprecation, and Migration Plan

We will bump API version for `ListTransactionsRequest and DescribeTransactionsResponse from 0 to 1.`

`When using a new AdminClient to send durationFilter (value greater than 0) to an older broker, AdminClient will fail to build the ListTransactionsRequest and throw an UnsupportedVersionException. This check will be made at ListTransactionsRequest.Builder.build(short version) method. A new AdminClient can still generate older version of ListTransactionsRequest when user sets durationFilter to 0 (or does not set a durationFilter).`

Rejected Alternatives

An alternative to enhancing these tools is to enable debug logging and parse through coordinator logs to get information like completion time and run duration for a transaction. Its better to enhance the tools (fix in case of `DescribeTransactions`) to provide a unified and convenient user experience.

We considered adding `DurationFilter` as a tagged field to `ListTransactionsRequest` and not bump the API version. This approach had a down side in terms of usability when a new client is talking to an older broker. New client can send `durationFilter` and assume that the returned transactions are running longer than the specified duration. It can further try to build follow up actions on these long running transactions like aborting them. This is dangerous if the broker supports older version of the API and does not recognize the new field. Broker will simply return all transactions. Therefore client can not build automated follow ups on the transactions returned by such `ListTransactionsRequest`