KIP-673: Emit JSONs with new auto-generated schema

Status

Current state: Approved

Discussion thread: *here*

JIRA: *KAFKA-10525*

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

Motivation

Kafka's request and response traces currently output in a format that is JSON-like and are not easily parsable. These are currently emitted by RequestChannel when logging is turned on at DEBUG level. Structured logs will be easier to load and parse with other tools like jq, elasticsearch, druid or presto.

There is a new auto-generated schema for each request type that supports outputting JSON payloads for request and response payloads. These can be adapted to provide structured request tracing.

Public Interfaces

The new auto-generated schemas generate a converter class for each request/response type such as `FetchRequestDataJsonConverter` when `./gradlew processMessages` is run. The signature looks as follows:

Method Signature of the auto-generated XYZDataJsonConverter

```
public class XYZDataJsonConverter {

        /* Converts the request/response data into a JsonNode*/
        public static JsonNode write(XYZData data, short version);
```

A new Scala singleton will be added to handle the data conversion to a JsonNode

Method Signature of RequestConvertToJson

```
object RequestConvertToJson {

        /**
         * The data converter for request types which calls the appropriate
         * XYZRequestDataJsonConverter.write()
         * @return JsonNode
         */
        def request(request: AbstractRequest, verbose: Boolean): JsonNode


        /**
         * The data converter for response types which calls the appropriate
         * XYZResponseDataJsonConverter.write()
         * @return JsonNode
         */
        def response(response: AbstractResponse, version: Short): JsonNode
}
```

With these helper functions, logs will become easily parsable.

For example, a request log currently looks like this:

```
Completed request: RequestHeader(apiKey=LIST_OFFSETS, apiVersion=5, clientId=consumer-group-1,
correlationId=8799) -- {replicaId=-1,isolationLevel=0,topics=[{name=test_topic,partitions=[{partitionIndex=0,
timestamp=1599676632886,currentLeaderEpoch=0}]}]},response={throttleTimeMs=0,topics=[{name=test_topic,partitions=
[{partitionIndex=0,errorCode=0,timestamp=1599676632886,offset=8700,leaderEpoch=0}]}]} from connection 127.0.0.1:
62610-127.0.0.1:62622-3;totalTime:0.085,requestQueueTime:0.013,localTime:0.033,remoteTime:0.011,throttleTime:0,
responseQueueTime:0.011,sendTime:0.015,sendIoTime:0.01,securityProtocol:PLAINTEXT,principal:User:ANONYMOUS,
listener:PLAINTEXT,clientInformation:ClientInformation(softwareName=apache-kafka-java, softwareVersion=unknown)
```

But after switching to use the auto-generated schemas, request logs will look like this:

```
Completed request: {"requestHeader": {"apiKey": "LIST_OFFSETS", "apiVersion": 5, "clientId": "consumer-group-1",
"correlationId": 8799}, "request": {"replicaId":-1,"isolationLevel":0,"topics":[{"name":"test_topic","partitions":
[{"partitionIndex":0,"timestamp":1599676632886,"currentLeaderEpoch":0}]}]},"response":{"throttleTimeMs":0,"
topics":[{"name":"test_topic","partitions":[{"partitionIndex":0,"errorCode":0,"timestamp":1599676632886,"offset":
8700,"leaderEpoch":0}]}]}, "connection": "127.0.0.1:62610-127.0.0.1:62622-3", "totalTime": 0.085,
"requestQueueTime": 0.013, "localTime": 0.033, "remoteTime":0.011, "throttleTime":0,"responseQueueTime":0.011,"
sendTime":0.015,"sendIoTime":0.01,"securityProtocol":"PLAINTEXT","principal":"User:ANONYMOUS","listener":"
PLAINTEXT","clientInformation":"ClientInformation(softwareName=apache-kafka-java, softwareVersion=unknown)"}
```

The addition of a `serializeRecords` parameter was added to the auto-generated schemas so that ProduceRequest and FetchResponse logs can either output the record's bytes or the record's size in bytes.

Proposed trace fields

Current Trace Field	New Key
RequestHeader	"requestHeader"
--	"request"
response	"response"
from connection	"connection"
totalTime	"totalTime"
requestQueueTime	"requestQueueTime"
localTime	"localTime"
remoteTime	"remoteTime"
throttleTime	"throttleTime"

responseQueueTime	"responseQueueTime"
sendTime	"sendTime"
sendIoTime	"sendIoTime"
securityProtocol	"securityProtocol"
principal	"principal"
listener	"listener"
clientInformation	"clientInformation"

Proposed Changes

Make each request type's `data` accessible. Construct a helper class `RequestConvertToJson` to handle converting the request data to a parsable JSON format.

In order to a log request, the appropriate helper function in `RequestConvertToJson` is called on the request. From there, the respective `XYZDataJsonConverter` is called and returns the JsonNode of the data which when converted to a string is in a parsable JSON format.

Compatibility, Deprecation, and Migration Plan

This simply changes the format of request and response logs.

Rejected Alternatives

Adding toJson() to each of the request types that calls data.toJson()

- This would require importing the `XYZRequestDataJsonConverter` class which has a Jackson dependency, but we want to avoid putting the Jackson dependency on the clients.