KIP-729: Custom validation of records on the broker prior to log append

Status

Current state: *Under Discussion*

Discussion thread: https://lists.apache.org/thread.html/r41f7c11d449a4a809030ec35de48f0f79b2dc94d68cb6143d01a150e%40%3Cdev.kafka.apache.org%3E

JIRA:

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

Motivation

The motivation here is to gain some grounds on data-quality in Kafka. The broker does validation of the records from Kafka's message format perspective, but there is no way to validate them from an application perspective. This makes it very hard to guarantee any data-quality on a Kafka topic, and the consumers have to use mitigation strategies like either a dead-letter-queue or block/crash on malformed data.

One example use case for illustration - Ensure that a topic ingests records only of a certain (or an allowed list of) schema(s). That is, a way to enforce Topic T to only append records belonging to Schema S, making Kafka topics schema aware (a bit like a traditional database).

An interface to allow custom validation logic to validate records before they are appended to local log could enable the above scenario and many more. A chain of validations could be used to perform multiple validations in series.

Public Interfaces

The broker configuration would have a new config *record.validator.classes* which would take a comma separated list of implementation classes of the *BrokerRecordValidator* interface.
By default, the value would be an empty string and that means that there is no extra validation configured.

Configuration Name	Valid Values	Default Value
record.validator.classes	class name of BrokerRecordValidator implementation	empty string

The *BrokerRecordValidator* interface is provided below:

```
interface BrokerRecordValidator {
    /**
     * Validate the record for a given topic-partition.
         */
        Optional<InvalidRecordException> validateRecord(TopicPartition topicPartition, ByteBuffer key,
ByteBuffer value, Header[] headers);
}
```

Proposed Changes

The chain of validations would be called in the LogValidator.scala class' validateRecord(), right after the calls to validateKey() and validateTimestamp(). The return type Optional<ApiRecordError> is the same as the other internal validate functions are returning as of today.

Here is the proposed place in the existing code: https://github.com/apache/kafka/blob/744d05b12897267803f46549e8bca3d31d57be4c/core/src/main/scala/kafka/log/LogValidator.scala#L211

Compatibility, Deprecation, and Migration Plan

The introduction of the new config would be backward compatible. Not using it (default value) would let the broker not perform any extra validations on records.
Users who want to use this feature would need to provide an implementation of the proposed interface and add the new configuration to the broker.

Rejected Alternatives

An alternative to this would be to have the producers validate the records before sending them to the brokers, but guaranteeing that is difficult as producers could have bugs.