KIP-579: new exception on min.insync.replicas > replication.factor

Status

Current state: Under Discussion

Discussion thread: here

JIRA: *KAFKA-4680*

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

Motivation

Currently it is possible to specify min.insync.replicas > than the replication.factor for a topic. By doing this, produce with acks=all fails with NotEnoughReplicaException after 3 attempts:

```
[...] WARN [Producer clientId=console-producer] Got error produce response with correlation id 4 on topic-
partition [...],
retrying (2 attempts left). Error: NOT_ENOUGH_REPLICAS (org.apache.kafka.clients.producer.internals.Sender)
[...] WARN [Producer clientId=console-producer] Got error produce response with correlation id 5 on topic-
partition [...],
retrying (1 attempts left). Error: NOT_ENOUGH_REPLICAS (org.apache.kafka.clients.producer.internals.Sender)
[...] WARN [Producer clientId=console-producer] Got error produce response with correlation id 6 on topic-
partition [...],
retrying (0 attempts left). Error: NOT_ENOUGH_REPLICAS (org.apache.kafka.clients.producer.internals.Sender)
[...] ERROR Error when sending message to topic [...] with key: [...], value: 3 bytes with error:
(org.apache.kafka.clients.producer.internals.ErrorLoggingCallback) org.apache.kafka.common.errors.
NotEnoughReplicasException:
Messages are rejected since there are fewer in-sync replicas than required.
```

Each produce request (including redos) emits an error message on the broker, which, if this is not noticed quickly enough, can cause the logs to balloon.

At the moment, this exception is thrown because the insync broker count is lower than the specified minIsrReplicas configuration. When min.insync.replicas > replication.factor, however, the insync replicas mismatch is due to an invalid configuration setup instead of a real availability issue. Furthermore, NotEnoughReplicaException is declared as retriable, hence if fails only after 3 attempts. However, in case, it does not make sense to retry as all the requests will fail.

On top of that, the error message which appears on the client is not indicative of the configuration mismatch as it is related to insync replicas requirements.

Public Interfaces

The proposal is to:

- add a new Exception, InconsistentReplicaConfigurationException, to be thrown when a client produce with acks=all on a topic where min.insync.replicas > replication.factor. This exception extends ApiException and is not retriable.

InconsistentReplicationFactorException.java

```
public class InconsistentReplicationFactorException extends ApiException {
        ...
}
```

This exception will replace NotEnoughReplicaException for the configuration mismatch scenario.

Note: NotEnoughReplicaException will still be thrown on in-sync brokers count < minIsr.

- add a new client-server Error, INCONSISTENT_REPLICA_CONFIGURATION(89)

Errors.java

```
INCONSISTENT_REPLICATION_FACTOR(89, "Replication factor is set lower than min.isr. Acks requirements
cannot be satisfied.", InconsistentReplicaConfigurationException::new);
```

- update public doc for min.insync.replicas:

```
val MinInSyncReplicasDoc = "When a producer sets acks to \"all\" (or \"-1\"), " +
"min.insync.replicas specifies the minimum number of replicas that must acknowledge " +
"a write for the write to be considered successful. If this minimum cannot be met, " +
"then the producer will raise an exception (either NotEnoughReplicas or " +
"NotEnoughReplicasAfterAppend).<br>When used together, min.insync.replicas and acks " +
"then the producer will raise an exception (InconsistentReplicationFactorException)." +
"<br>When used together, min.insync.replicas and acks " +
"allow you to enforce greater durability guarantees. A typical scenario would be to " +
"create a topic with a replication factor of 3, set min.insync.replicas to 2, and " +
"produce with acks of \"all\". This will ensure that the producer raises an exception " +
"if a majority of replicas do not receive a write.
```

Proposed Changes

Proposal is to throw InconsistentReplicaConfigurationException when producing on a topic with replicationFactor < minIsr and requiring acks == all.

This change involves updating appendRecordToLeader in Partition.scala to verify the mismatch above. The verification is performed before checking the number of insync brokers to allow maintaining the same logic for NotEnoughReplicas.

Partition.scala

```
...
val replicationFactor = assignmentState.replicationFactor

// Avoid writing to leader if replication factor < min insync replicas
if (replicationFactor < minIsr && requiredAcks == -1) {
    throw new InconsistentReplicaConfigurationException(s"Replication factor [$replicationFactor] is < min.isr
[$minIsr]" +
    s"for topic ${topicPartition.topic()}. Acks requirements cannot be satisfied when
requiredAcks=$requiredAcks")
}

...
```

Compatibility, Deprecation, and Migration Plan

Clients expecting an error when producing on a topic with acks=all will receive INCONSISTENT_REPLICATION_FACTOR(89) instead of NOT_ENOUGH_REPLICAS (19) in the Java client (new Exception is not retriable) in case of the aforementioned configuration mismatch.

Rejected Alternatives

1) Anticipate the validation (min.insync.replicas <= replication.factor) at topic creation / configuration change and reject the change. This requires to validate the configuration in more than one place: at topic creation, at configuration setup/update (both for min.insync.replicas and the default.replication.factor), at partition reassignment (when reducing replication factor). This solution was rejected because it is still possible to create a topic with min.insync.replicas > replication.factor and produce/consume on it with acks != all, we don't want to disable this behaviour. In addition this might introduce new failure paths in the creation of internal topics.