Kafka Improvement Proposals

This page describes a proposed Kafka Improvement Proposal (KIP) process for proposing a major change to Kafka.

To create your own KIP, click on Create KIP. If you don't have permission, please send an email with your Wiki ID to `dev@kafka.apache.org` and request permission (http://kafka.apache.org/contact). Also add an entry to the table KIPs under discussion (for Streams API KIPs, please also add it to Kafka Streams sub page).

Purpose

We want to make Kafka a core architectural component for users. We also support a large number of integrations with other tools, systems, and clients. Keeping this kind of usage health requires a high level of compatibility between releases — core architectural elements can't break compatibility or shift functionality from release to release. As a result each new major feature or public api has to be done in a way that we can stick with it going forward.

This means when making this kind of change we need to think through what we are doing as best we can prior to release. And as we go forward we need to stick to our decisions as much as possible. All technical decisions have pros and cons so it is important we capture the thought process that lead to a decision or design to avoid flip-flopping needlessly.

Hopefully we can make these proportional in effort to their magnitude — small changes should just need a couple brief paragraphs, whereas large changes need detailed design discussions.

This process also isn't meant to discourage incompatible changes — proposing an incompatible change is totally legitimate. Sometimes we will have made a mistake and the best path forward is a clean break that cleans things up and gives us a good foundation going forward. Rather this is intended to avoid accidentally introducing half thought-out interfaces and protocols that cause needless heartburn when changed. Likewise the definition of "compatible" is itself squishy: small details like which errors are thrown when are clearly part of the contract but may need to change in some circumstances, likewise performance isn't part of the public contract but dramatic changes may break use cases. So we just need to use good judgement about how big the impact of an incompatibility will be and how big the payoff is.

What is considered a "major change" that needs a KIP?

Any of the following should be considered a major change:

- Any major new feature, subsystem, or piece of functionality
- Any change that impacts the public interfaces of the project

What are the "public interfaces" of the project?

All of the following are public interfaces that people build around:

- Binary log format
- The network protocol and api behavior
- Any class in the public packages under clients
    - org/apache/kafka/common/serialization
    - org/apache/kafka/common
    - org/apache/kafka/common/errors
    - org/apache/kafka/clients/producer
    - org/apache/kafka/clients/consumer (eventually, once stable)
- Configuration, especially client configuration
- Monitoring
- Command line tools and arguments

Not all compatibility commitments are the same. We need to spend significantly more time on log format and protocol as these break code in lots of clients, cause downtime releases, etc. Public apis are next as they cause people to rebuild code and lead to compatibility issues in large multi-dependency projects (which end up requiring multiple incompatible versions). Configuration, monitoring, and command line tools can be faster and looser — changes here will break monitoring dashboards and require a bit of care during upgrades but aren't a huge burden.

For the most part monitoring, command line tool changes, and configs are added with new features so these can be done with a single KIP.

What should be included in a KIP?

A KIP should contain the following sections:

- **Motivation**: describe the problem to be solved
- **Proposed Change**: describe the new thing you want to do. This may be fairly extensive and have large subsections of its own. Or it may be a few sentences, depending on the scope of the change.
- **New or Changed Public Interfaces**: impact to any of the "compatibility commitments" described above. We want to call these out in particular so everyone thinks about them.
- **Migration Plan and Compatibility**: if this feature requires additional support for a no-downtime upgrade describe how that will work
- **Rejected Alternatives**: What are the other alternatives you considered and why are they worse? The goal of this section is to help people understand why this is the best solution now, and also to prevent churn in the future when old alternatives are reconsidered.

Who should initiate the KIP?

Anyone can initiate a KIP but you shouldn't do it unless you have an intention of getting the work done to implement it (otherwise it is silly).

Process

Here is the process for making a KIP:

1. Click Create KIP. Take the next available KIP number and give your proposal a descriptive heading. e.g. "KIP 42: Allow Infinite Retention With Bounded Disk Usage".
2. Fill in the sections as described above
3. Start a [DISCUSS] thread on the Apache mailing list. Please ensure that the subject of the thread is of the format *[DISCUSS] KIP-{your KIP number} {your KIP heading}* and the body contains a link to your new KIP. The discussion should happen on the mailing list, not on the wiki, since the wiki comment system doesn't work well for larger discussions. In the process of the discussion you may update the proposal. You should let people know the changes you are making. When you feel you have a finalized proposal
4. Once the proposal is finalized call a [VOTE] to have the proposal adopted. These proposals are more serious than code changes and more serious even than release votes. The criteria for acceptance is lazy majority. The vote should remain open for at least 72 hours.
5. ***Please*** update the KIP wiki page, and the index below, to reflect the current stage of the KIP after a vote. This acts as the permanent record indicating the result of the KIP (e.g., Accepted or Rejected). Also report the result of the KIP vote to the voting thread on the mailing list so the conclusion is clear.

KIP round-up

Next KIP Number: 477

Use this number as the identifier for your KIP and increment this value.

Adopted KIPs

*Please insert new rows in **sorted order** (by KIP number).*

	KIP (please keep this sorted by KIP number)	Release
1	KIP-465: Add Consolidated Connector Endpoint to Connect REST API	2.3.0
2	KIP-464: Defaults for AdminClient#createTopic	2.4.0
3	KIP-462: Use local thread id for KStreams	2.3.0
4	KIP-461: Improve Replica Fetcher behavior at handling partition failure	2.3.0
5	KIP-460: Admin Leader Election RPC	2.4.0
6	KIP-458: Connector Client Config Override Policy	2.3.0
7	KIP-454: Expansion of the ConnectClusterState interface	2.3.0
8	KIP-453: Add close() method to RocksDBConfigSetter	2.3.0
9	KIP-449: Add connector contexts to log messages in Connect workers	2.3.0
10	KIP-446: Add changelog topic configuration to KTable suppress	2.4.0 (WIP)
11	KIP-445: In-memory Session Store	2.3.0
12	KIP-443: Return to default segment.ms and segment.index.bytes in Streams repartition topics	2.3.0
13	KIP-442: Return to default max poll interval in Streams	2.3.0
14	KIP-436: Add a metric indicating start time	2.3.0
15	KIP-430 - Return Authorized Operations in Describe Responses	2.3.0
16	KIP-429: Kafka Consumer Incremental Rebalance Protocol	2.4.0 (WIP)

17	KIP-428: Add in-memory window store	2.3.0
18	KIP-427: Add AtMinIsr topic partition category (new metric & TopicCommand option)	2.3.0
19	KIP-425: Add some Log4J Kafka Appender Properties for Producing to Secured Brokers	2.3.0
20	KIP-421: Support resolving externalized secrets in AbstractConfig	2.3.0
21	KIP-420: Add Single Value Fetch in Session Stores	2.2.0
22	KIP-417: Allow JmxTool to connect to a secured RMI port	2.3.0
23	KIP-415: Incremental Cooperative Rebalancing in Kafka Connect	2.3.0
24	KIP-414: Expose Embedded ClientIds in Kafka Streams	2.2.0
25	KIP-412: Extend Admin API to support dvynamic application log levels	2.4.0 (WIP)
26	KIP-411: Make default Kafka Connect worker task client IDs distinct	2.3.0
27	KIP-402: Improve fairness in SocketServer processors	2.2.0 (partially implemented) / 2.3.0
28	KIP-394: Require member.id for initial join group request	2.2.0
29	KIP-393: Time windowed serde to properly deserialize changelog input topic	2.2.0
30	KIP-386: Standardize on Min/Avg/Max metrics' default value	2.2.0
31	KIP-382: MirrorMaker 2.0	2.4.0 (WIP)
32	KIP-380: Detect outdated control requests and bounced brokers using broker generation	2.2.0
33	KIP-377: TopicCommand to use AdminClient	2.2.0
34	KIP-376: Implement AutoClosable on appropriate classes that want to be used in a try-with-resource statement	2.2.0
35	KIP-374: Add '--help' option to all available Kafka CLI commands	2.2.0
36	KIP-372: Naming Repartition Topics for Joins and Grouping	2.1.0
37	KIP-371: Add a configuration to build custom SSL principal name	2.2.0
38	KIP 368: Allow SASL Connections to Periodically Re-Authenticate	2.2.0
39	KIP-367 Introduce close(Duration) to Producer and AdminClient instead of close(long, TimeUnit)	2.2.0
40	KIP-366: Make FunctionConversions deprecated	2.1.0
41	KIP-365: Materialized, Serialized, Joined, Consumed and Produced with implicit Serde	2.1.0
42	KIP-361: Add Consumer Configuration to Disable Auto Topic Creation	2.3.0
43	KIP-359: Verify leader epoch in produce requests	2.4.0 (WIP)
44	KIP-358: Migrate Streams API to Duration instead of long ms times	2.1.0
45	KIP-357: Add support to list ACLs per principal	2.1.0
46	KIP-356: Add withCachingDisabled() to StoreBuilder	2.1.0
47	KIP-354: Add a Maximum Log Compaction Lag	2.3.0
48	KIP-353: Improve Kafka Streams Timestamp Synchronization	2.1.0
49	KIP-351: Add --under-min-isr option to describe topics command	2.3.0
50	KIP-346: Improve LogCleaner behavior on error	2.1
51	KIP-345: Introduce static membership protocol to reduce consumer rebalances	2.4.0 (WIP), partially available in 2.3.0
52	KIP-342 Add support for custom SASL extensions in OAuthBearer authentication	2.1.0
53	KIP-341: Update Sticky Assignor's User Data Protocol	2.3.0
54	KIP-340: Allow kafka-reassign-partitions.sh and kafka-log-dirs.sh to take admin client property file	2.1.0
55	KIP-339: Create a new IncrementalAlterConfigs API	2.3.0
56	KIP-338 Support to exclude the internal topics in kafka-topics.sh command	2.1.0

57	KIP-336: Consolidate ExtendedSerializer/Serializer and ExtendedDeserializer/Deserializer	2.1.0
58	KIP-332: Update AclCommand to use AdminClient API	2.1.0
59	KIP-331 Add default implementation to close() and configure() for Serializer, Deserializer and Serde	2.3.0
60	KIP-330: Add retentionPeriod in SessionBytesStoreSupplier	2.1.0
61	KIP-328: Ability to suppress updates for KTables	2.1.0 (partially implemented) / 2.3.0 (WIP)
62	KIP-324: Add method to get metrics() in AdminClient	2.1.0
63	KIP-322: Return new error code for DeleteTopics API when topic deletion disabled.	2.1.0
64	KIP-321: Update TopologyDescription to better represent Source and Sink Nodes	2.1.0
65	KIP-320: Allow fetchers to detect and handle log truncation	2.1.0 (partially implemented) / 2.2.0
66	KIP-319: Replace segments with segmentInterval in WindowBytesStoreSupplier	2.1.0
67	KIP-313: Add KStream.flatTransform and KStream.flatTransformValues	2.2.0 (partially implemented) / 2.3.0
68	KIP-312 Add Overloaded StreamsBuilder Build Method to Accept java.util.Properties	2.1.0
69	KIP-308: Support dynamic update of max.connections.per.ip/max.connections.per.ip.overrides	2.1.0
70	KIP-307: Allow to define custom processor names with KStreams DSL	2.3.0 (partial)
71	KIP-306: Configuration for Delaying Response to Failed Authentication	2.1.0
72	KIP-305: Add Connect primitive number converters	2.0.0
73	KIP-303: Add Dynamic Routing in Streams Sink	2.0.0
74	KIP-302 - Enable Kafka clients to use all DNS resolved IP addresses	2.1.0
75	KIP-300: Add Windowed KTable API in StreamsBuilder	2.4.0
76	KIP-298: Error Handling in Connect	2.0.0
77	KIP-297: Externalizing Secrets for Connect Configurations	2.0.0
78	KIP-295 Add Streams Configuration Allowing for Optional Topology Optimization	2.0.0
79	KIP-294 - Enable TLS hostname verification by default	2.0.0
80	KIP-292: Add transformValues() method to KTable	2.0.0
81	KIP-290: Support for Prefixed ACLs	2.0.0
82	KIP-289: Improve the default group id behavior in KafkaConsumer	2.2.0
83	KIP-285: Connect Rest Extension Plugin	2.0.0
84	KIP-284: Set default retention ms for Streams repartition topics to Long.MAX_VALUE	2.0.0
85	KIP-283: Efficient Memory Usage for Down-Conversion	2.0.0
86	KIP-282: Add the listener name to the authentication context	2.0.0
87	KIP-281: ConsumerPerformance: Increase Polling Loop Timeout and Make It Reachable by the End User	2.0.0
88	KIP-279: Fix log divergence between leader and follower after fast leader fail over	2.0.0
89	KIP-278 - Add version option to Kafka's commands	2.0.0
90	KIP-277 - Fine Grained ACL for CreateTopics API	2.0.0
91	KIP-276 - Add StreamsConfig prefix for different consumers	2.0.0
92	KIP-274: Kafka Streams Skipped Records Metrics	2.0.0
93	KIP-272: Add API version tag to broker's RequestsPerSec metric	2.0.0
94	KIP-270 - A Scala Wrapper Library for Kafka Streams	2.0.0
95	KIP-268: Simplify Kafka Streams Rebalance Metadata Upgrade	2.0.0

96	KIP-267: Add Processor Unit Test Support to Kafka Streams Test Utils	2.0.0
97	KIP-266: Fix consumer indefinite blocking behavior	2.0.0
98	KIP-265: Make Windowed Serde to public APIs	2.0.0
99	KIP-261: Add Single Value Fetch in Window Stores	2.0.0
100	KIP-258: Allow to Store Record Timestamps in RocksDB	2.3.0 (partially implemented)
101	KIP-257 - Configurable Quota Management	2.0.0
102	KIP-255: OAuth Authentication via SASL/OAUTHBEARER	2.0.0
103	KIP-251: Allow timestamp manipulation in Processor API	2.0.0
104	KIP-249: Add Delegation Token Operations to KafkaAdminClient	2.0.0
105	KIP-247: Add public test utils for Kafka Streams	1.1.0
106	KIP-245: Use Properties instead of StreamsConfig in KafkaStreams constructor	2.0.0
107	KIP-244: Add Record Header support to Kafka Streams Processor API	2.0.0
108	KIP-243: Make ProducerConfig and ConsumerConfig constructors public	1.1.0
109	KIP-239 Add queryableStoreName() to GlobalKTable	1.1.0
110	KIP-238: Expose Kafka cluster ID in Connect REST API	1.1.0
111	KIP-237: More Controller Health Metrics	2.0.0
112	KIP-235: Add DNS alias support for secured connection	2.1.0
113	KIP-233: Simplify StreamsBuilder#addGlobalStore	1.1.0
114	KIP-231: Improve the Required ACL of ListGroups API	2.1.0
115	KIP-229: DeleteGroups API	1.1.0
116	KIP-227 - Introduce Incremental FetchRequests to Increase Partition Scalability	1.1.0
117	KIP-226 - Dynamic Broker Configuration	1.1.0
118	KIP-225 - Use tags for consumer “records.lag” metrics	1.1.0
119	KIP-224: Add configuration parameter `retries` to Streams API	1.1.0
120	KIP-223 - Add per-topic min lead and per-partition lead metrics to KafkaConsumer	2.0.0
121	KIP-222 - Add Consumer Group operations to Admin API	2.0.0
122	KIP-220: Add AdminClient into Kafka Streams' ClientSupplier	1.1.0
123	KIP-219 - Improve quota communication	2.0.0
124	KIP-218: Make KafkaFuture.Function java 8 lambda compatible	1.1.0
125	KIP-215: Add topic regex support for Connect sinks	1.1.0
126	KIP-214: Add zookeeper.max.in.flight.requests config to the broker	1.1.0
127	KIP-213 Support non-key joining in KTable	2.4.0 (WIP)
128	KIP-212: Enforce set of legal characters for connector names	1.1.0
129	KIP-211: Revise Expiration Semantics of Consumer Group Offsets	2.1.0
130	KIP-210 - Provide for custom error handling when Kafka Streams fails to produce	1.1.0
131	KIP-208: Add SSL support to Kafka Connect REST interface	1.1.0
132	KIP-207: Offsets returned by ListOffsetsResponse should be monotonically increasing even during a partition leader change	2.2.0
133	KIP-206: Add support for UUID serialization and deserialization	2.1.0
134	KIP-205: Add all() and range() API to ReadOnlyWindowStore	1.1.0
135	KIP-204 : Adding records deletion operation to the new Admin Client API	1.1.0

136	KIP-203: Add toLowerCase support to sasl.kerberos.principal.to.local rule	1.1.0
137	KIP-202: Move merge() from StreamsBuilder to KStream	1.0.0
138	KIP-198: Remove ZK dependency from Streams Reset Tool	1.0.0
139	KIP-197 Connect REST API should include the connector type when describing a connector	1.0.0
140	KIP-196: Add metrics to Kafka Connect framework	1.0.0
141	KIP-195: AdminClient.createPartitions	1.0.0
142	KIP-192 : Provide cleaner semantics when idempotence is enabled	1.0.0
143	KIP-191: KafkaConsumer.subscribe() overload that takes just Pattern	1.0.0
144	KIP-190: Handle client-ids consistently between clients and brokers	1.0.0
145	KIP-189: Improve principal builder interface and add support for SASL	1.0.0
146	KIP-188 - Add new metrics to support health checks	1.0.0
147	KIP-187 - Add cumulative count metric for all Kafka rate metrics	1.0.0
148	KIP-186: Increase offsets retention default to 7 days	2.0.0
149	KIP-183 - Change PreferredReplicaLeaderElectionCommand to use AdminClient	2.2.0
150	KIP-182: Reduce Streams DSL overloads and allow easier use of custom storage engines	1.0.0
151	KIP-180: Add a broker metric specifying the number of consumer group rebalances in progress	1.1.0
152	KIP-177: Consumer perf tool should count rebalance time	1.0.0
153	KIP-176: Remove deprecated new-consumer option for tools	2.0.0
154	KIP-175: Additional '--describe' views for ConsumerGroupCommand	1.1.0
155	KIP-174 - Deprecate and remove internal converter configs in WorkerConfig	2.0.0
156	KIP-173: Add prefix to StreamsConfig to enable setting default internal topic configs	1.0.0
157	KIP-171 - Extend Consumer Group Reset Offset for Stream Application	1.1.0
158	KIP-168: Add GlobalTopicCount and GlobalPartitionCount metric per cluster	1.0.0
159	KIP-167: Add interface for the state store restoration process	1.0.0
160	KIP-164 - Add UnderMinIsrPartitionCount and per-partition UnderMinIsr metrics	1.0.0
161	KIP-163: Lower the Minimum Required ACL Permission of OffsetFetch	1.0.0
162	KIP-162: Enable topic deletion by default	1.0.0
163	KIP-161: streams deserialization exception handlers	1.0.0
164	KIP-160: Augment KStream.print(), KStream.writeAsText() to allow users pass in extra parameters in the printed string	1.0.0
165	KIP-157 - Add consumer config options to streams reset tool	1.0.0
166	KIP-156 Add option "dry run" to Streams application reset tool	0.11.0.0
167	KIP-155 - Add range scan for windowed state stores	0.11.0.0
168	KIP-154 Add Kafka Connect configuration properties for creating internal topics	0.11.0.0
169	KIP-153: Include only client traffic in BytesOutPerSec metric	0.11.0.0
170	KIP-152 - Improve diagnostics for SASL authentication failures	1.0.0
171	KIP-151 Expose Connector type in REST API	0.11.0.0
172	KIP-150 - Kafka-Streams Cogroup	2.4.0 (WIP)
173	KIP-149: Enabling key access in ValueTransformer, ValueMapper, and ValueJoiner	1.1.0 (partially implemented) (WIP for 2.4.0)
174	KIP-146 - Classloading Isolation in Connect	0.11.0.0
175	KIP-145 - Expose Record Headers in Kafka Connect	1.1.0

176	KIP-144: Exponential backoff for broker reconnect attempts	0.11.0.0
177	KIP-143: Controller Health Metrics	0.11.0.0
178	KIP-140: Add administrative RPCs for adding, deleting, and listing ACLs	0.11.0.0
179	KIP-138: Change punctuate semantics	1.0.0
180	KIP-137: Enhance TopicCommand --describe to show topics marked for deletion	0.11.0.0
181	KIP-136: Add Listener name to SelectorMetrics tags	0.11.0.0
182	KIP-134: Delay initial consumer group rebalance	0.11.0.0
183	KIP-133: Describe and Alter Configs Admin APIs	0.11.0.0
184	KIP-130: Expose states of active tasks to KafkaStreams public API	1.0.0
185	KIP-129: Streams Exactly-Once Semantics	0.11.0.0
186	KIP-128: Add ByteArrayConverter for Kafka Connect	0.11.0.0
187	KIP-126 - Allow KafkaProducer to split and resend oversized batches.	0.11.0.0
188	KIP-124 - Request rate quotas	0.11.0.0
189	KIP-123: Allow per stream/table timestamp extractor	0.11.0.0
190	KIP-122: Add Reset Consumer Group Offsets tooling	0.11.0.0
191	KIP-121: Add KStream peek method	0.11.0.0
192	KIP-120: Cleanup Kafka Streams builder API	1.0.0
193	KIP-119: Drop Support for Scala 2.10 in Kafka 0.11	0.11.0.0
194	KIP-118: Drop Support for Java 7	2.0.0
195	KIP-117: Add a public AdminClient API for Kafka admin operations	0.11.0.0
196	KIP-115: Enforce offsets.topic.replication.factor upon __consumer_offsets auto topic creation	0.11.0.0
197	KIP-114: KTable state stores and improved semantics	0.11.0.0
198	KIP-113: Support replicas movement between log directories	1.1.0
199	KIP-112: Handle disk failure for JBOD	1.0.0
200	KIP-110: Add Codec for ZStandard Compression	2.1.0
201	KIP-109: Old Consumer Deprecation	0.11.0.0
202	KIP-108: Create Topic Policy	0.10.2.0
203	KIP-107: Add deleteRecordsBefore() API in AdminClient	0.11.0.0
204	KIP-106 - Change Default unclean.leader.election.enabled from True to False	0.11.0.0
205	KIP-105: Addition of Record Level for Sensors	0.10.2.0
206	KIP-104: Granular Sensors for Streams	0.10.2.0
207	KIP-103: Separation of Internal and External traffic	0.10.2.0
208	KIP-102 - Add close with timeout for consumers	0.10.2.0
209	KIP-101 - Alter Replication Protocol to use Leader Epoch rather than High Watermark for Truncation	0.11.0.0
210	KIP-100 - Relax Type constraints in Kafka Streams API	0.10.2.0
211	KIP-99: Add Global Tables to Kafka Streams	0.10.2.0
212	KIP-98 - Exactly Once Delivery and Transactional Messaging	0.11.0.0
213	KIP-97: Improved Kafka Client RPC Compatibility Policy	0.10.2.0
214	KIP-96 - Add per partition metrics for in-sync and assigned replica count	0.10.2.0
215	KIP-94 Session Windows	0.10.2.0
216	KIP-93: Improve invalid timestamp handling in Kafka Streams	0.10.2.0

217	KIP-92 - Add per partition lag metrics to KafkaConsumer	0.10.2.0
218	KIP-91 Provide Intuitive User Timeouts in The Producer	2.1.0
219	KIP-90 - Remove zkClient dependency from Streams	0.10.2.0
220	KIP-89: Allow sink connectors to decouple flush and offset commit	0.10.2.0
221	KIP-88: OffsetFetch Protocol Update	0.10.2.0
222	KIP-86: Configurable SASL callback handlers	2.0.0
223	KIP-85: Dynamic JAAS configuration for Kafka clients	0.10.2.0
224	KIP-84: Support SASL SCRAM mechanisms	0.10.2.0
225	KIP-82 - Add Record Headers	0.11.0.0
226	KIP-81: Bound Fetch memory usage in the consumer	2.4.0 (WIP)
227	KIP-79 - ListOffsetRequest/ListOffsetResponse v1 and add timestamp search methods to the new consumer	0.10.1.0
228	KIP-78: Cluster Id	0.10.1.0
229	KIP-77: Improve Kafka Streams Join Semantics	0.10.2.0
230	KIP-75 - Add per-connector Converters	0.10.1.0
231	KIP-74: Add Fetch Response Size Limit in Bytes	0.10.1.0
232	KIP-73: Replication Quotas	0.10.1.0
233	KIP-72: Allow putting a bound on memory consumed by Incoming request	1.0.0
234	KIP-71: Enable log compaction and deletion to co-exist	0.10.1.0
235	KIP-70: Revise Partition Assignment Semantics on New Consumer's Subscription Change	0.10.1.0
236	KIP-67: Queryable state for Kafka Streams	0.10.1.0
237	KIP-66: Single Message Transforms for Kafka Connect	0.10.2.0 / 0.11.0.0
238	KIP-65: Expose timestamps to Connect	0.10.1.0
239	KIP-63: Unify store and downstream caching in streams	0.10.1.0
240	KIP-62: Allow consumer to send heartbeats from a background thread	0.10.1.0
241	KIP-60 - Make Java client classloading more flexible	0.10.1.0
242	KIP-58 - Make Log Compaction Point Configurable	0.10.1.0
243	KIP-57 - Interoperable LZ4 Framing	0.10.0.0
244	KIP-56: Allow cross origin HTTP requests on all HTTP methods	0.10.0.0
245	KIP-55: Secure Quotas for Authenticated Users	0.10.1.0
246	KIP-54: Sticky Partition Assignment Strategy	0.11.0.0
247	KIP-52: Connector Control APIs	0.10.0.0
248	KIP-51 - List Connectors REST API	0.10.0.0
249	KIP-50 - Move Authorizer to o.a.k.common package	0.10.1.0
250	KIP-48 Delegation token support for Kafka	1.1.0
251	KIP-45 - Standardize all client sequence interaction on j.u.Collection.	0.10.0.0
252	KIP-43: Kafka SASL enhancements	0.10.0.0
253	KIP-42: Add Producer and Consumer Interceptors	0.10.0.0
254	KIP-41: Consumer Max Records	0.10.0.0
255	KIP-40: ListGroups and DescribeGroup	0.9.0.0
256	KIP-38: ZooKeeper Authentication	0.9.0.0

257	KIP-36 - Rack aware replica assignment	0.10.0.0
258	KIP-35 - Retrieving protocol version	0.10.0.0
259	KIP-33 - Add a time based log index	0.10.1.0
260	KIP-32 - Add timestamps to Kafka message	0.10.0.0
261	KIP-31 - Move to relative offsets in compressed message sets	0.10.0.0
262	KIP-28 - Add a processor client	0.10.0.0
263	KIP-26 - Add Kafka Connect framework for data import/export	0.9.0.0
264	KIP-25 - System test improvements	0.9.0.0
265	KIP-22 - Expose a Partitioner interface in the new producer	0.9.0.0
266	KIP-21 - Dynamic Configuration	0.9.0.0 (WIP)
267	KIP-20 Enable log preallocate to improve consume performance under windows and some old Linux file system	0.9.0.0
268	KIP-19 - Add a request timeout to NetworkClient	0.9.0.0
269	KIP-16 - Automated Replica Lag Tuning	0.9.0.0
270	KIP-15 - Add a close method with a timeout in the producer	0.9.0.0
271	KIP-13 - Quota Design	0.9.0.0
272	KIP-12 - Kafka Sasl/Kerberos and SSL implementation	0.9.0.0
273	KIP-11 - Kafka Authorizer design	0.9.0.0
274	KIP-8 - Add a flush method to the producer API	0.9.0.0
275	KIP-4 - Metadata Protocol Changes	0.10.0.0
276	KIP-4 - Command line and centralized administrative operations	0.9.0.0, 0.10.0.0, 0.10.1.0
277	KIP-3 - Mirror Maker Enhancement	0.9.0.0
278	KIP-2 - Refactor brokers to allow listening on multiple ports and IPs	0.9.0.0
279	KIP-1 - Remove support of request.required.acks	0.9.0.0

KIPs under discussion

Please insert new rows in sorted order (by KIP number).

KIP	Comment
KIP-59: Proposal for a kafka broker command	Sent emails to Dev discussion group. Work tracked under KAFKA-3663.
KIP-125: ZookeeperConsumerConnector to KafkaConsumer Migration and Rollback	
KIP-131 - Add access to OffsetStorageReader from SourceConnector	
KIP-135 : Send of null key to a compacted topic should throw non-retriable error back to user	
KIP 141 - ProducerRecord & SourceRecord: Add timestamp constructors	
KIP-142: Add ListTopicsRequest to efficiently list all the topics in a cluster	
KIP-148: Add a connect timeout for client	
KIP-158: Kafka Connect should allow source connectors to set topic-specific settings for new topics	
KIP-159: Introducing Rich functions to Streams	
KIP-166 - Add a tool to make amounts of replicas and leaders on brokers balanced	

KIP-169 - Lag-Aware Partition Assignment Strategy	
KIP-178: Size-based log directory selection strategy	
KIP-185: Make exactly once in order delivery the default producer setting	
KIP-193: Add SchemaBuilder.from(Schema)	
KIP-199: Add Kafka Connect offset tool	
KIP-201: Rationalising Policy interfaces	
KIP-209: Connection String Support	
KIP-216: IQ should throw different exceptions for different errors	
KIP-217: Expose a timeout to allow an expired ZK session to be re-created	
KIP-221: Enhance KStream with Connecting Topic Creation and Repartition Hint	Discussion
KIP-228 Negative record timestamp support	Voting in progress
KIP-234: add support for getting topic defaults from AdminClient	
KIP-236: Interruptible Partition Reassignment	Discussion
KIP-240: AdminClient.listReassignments() AdminClient.describeReassignments()	
KIP-242: Mask password in Kafka Connect Rest API response	
KIP-250 Add Support for Quorum-based Producer Acknowledgment	
KIP-252 - Extend ACLs to allow filtering based on ip ranges and subnets	
KIP-253: Support in-order message delivery with partition expansion	Discussion
KIP-254: JsonConverter Exception Handeling	
KIP-259: Improve Streams DSL Timestamp Propagation Semantics	
KIP-260: add primary join operation for Stream-Stream join (WIP)	Draft
KIP-264: Add a consumer metric to record raw fetch size	Voting in progress
KIP-271: Add NetworkClient redirector	Discussion
KIP-273: Kafka to support using ETCD beside Zookeeper	Discussion
KIP-275 - Indicate "isClosing" in the SinkTaskContext	Voting in progress
KIP-280: Enhanced log compaction	Discussion
KIP-291: Separating controller connections and requests from the data plane	Accepted
KIP-293 Add new metrics for consumer/replication fetch requests	Voting in progress
KIP-296: Add connector level configurability for producer/consumer client configs	Discussion
KIP-301: Schema Inferencing for JsonConverter	Discussion
KIP-304: Connect runtime mode improvements for container platforms	Discussion
KIP-314: KTable to GlobalKTable Bi-directional Join	Discussion
KIP-315: Stream Join Sticky Assignor	Discussion
KIP-316: Command-line overrides for ConnectDistributed worker properties	Discussion
KIP-317: Add end-to-end data encryption functionality to Apache Kafka	Discussion
KIP-325: Extend Consumer Group Command to Show Beginning Offsets	Voting in Progress345
KIP-326: Schedulable KTable as Graph source	Discussion
KIP-333: Add faster mode of rebalancing	Discussion
KIP-334 - Include partitions in exceptions raised during consumer record deserialization /validation	
KIP-335: Consider configurations for KafkaStreams	Discussion

KIP-347: Enable batching in FindCoordinatorRequest	Discussion
KIP-348 Eliminate null from SourceTask#poll()	
KIP-350: Allow kafka-topics.sh to take brokerid as parameter to show partitions associated with it	
KIP-356: Add KafkaConsumer fetch-error-rate and fetch-error-total metrics	Discussion
KIP-360: Improve handling of unknown producer	Discussion
KIP-362: Support Dynamic Session Window	Discussion
KIP-363: Allow performance tools to print final results to output file	Discussion
KIP-369: Alternative Partitioner to Support "Always Round-Robin" Selection	Accepted
KIP-370: Remove Orphan Partitions	Discussion
KIP-373: Allow users to create delegation tokens for other users	Discussion
KIP-375: Kafka Clients - make Metadata#TOPIC_EXPIRY_MS configurable	Discussion
KIP-378: Enable Dependency Injection for Kafka Streams handlers	Discussion
KIP-379: Multiple Consumer Group Management	Accepted
KIP-381: Connect: Tell about records that had their offsets flushed in callback	Voting in progress (restarted 18th January 2019, due to no votes in first attempt)
KIP-383: Pluggable interface for SSL Factory	Voting in progress
KIP-384: Add config for incompatible changes to persistent metadata	Discussion
KIP-385: Avoid throwing away prefetched data	Discussion
KIP-387: Fair Message Consumption Across Partitions in KafkaConsumer	Discussion
KIP-388: Add observer interface to record request and response	Discussion
KIP-389: Introduce a configurable consumer group size limit	Accepted
KIP-390: Allow fine-grained configuration for compression	Discussion, JIRA exists with pull-request
KIP-391: Allow Producing with Offsets for Cluster Replication	Discussion, JIRA exists with pull-request
KIP-392: Allow consumers to fetch from closest replica	Discussion
KIP-395: Encypt-then-MAC Delegation token metadata	
KIP-396: Add Commit/List Offsets Operations to AdminClient	Discussion
KIP-397: Add methods to override fetch offsets based on timestamp	Discussion
KIP-398: Support reading trust store from classpath	Discussion
KIP-399: Extend ProductionExceptionHandler to cover serialization exceptions	Discussion
KIP-400: Improve exit status in case of errors in ConsoleProducer	Discussion
KIP-401: TransformerSupplier/ProcessorSupplier StateStore connecting	Voting in progress
KIP-403: Increase ProducerPerformance precision by using nanoTime	Draft
KIP-405: Kafka Tiered Storage	Discussion
KIP-406: GlobalStreamThread should honor custom offset policy.	Discussion
KIP-407: Kafka Connect support override worker kafka api configuration with connector configuration that post by rest api	
KIP-408: Add asynchronous processing to Kafka Streams	Discussion
KIP-409: Allow creating under-replicated topics and partitions	
KIP-410: Add metric for request handler thread pool utilization by request type	Discussion
KIP-416: Notify SourceTask of ACK'd offsets, metadata	Discussion
KIP-418: A method-chaining way to branch KStream	Discussion, JIRA exists with pull-request

KIP-419: Safely notify Kafka Connect SourceTask is stopped	Voting in progress, JIRA exists with pull request
KIP-421: Support resolving externalized secrets in AbstractConfig	Accepted
KIP-422: Add support for client quota configuration in the Kafka Admin Client	Discussion, JIRA exists with pull-request
KIP-423: Add JoinReason to Consumer Join Group Protocol	Discussion
KIP-424: Allow suppression of intermediate events based on wall clock time	Discussion
KIP-426: Persist Broker Id to Zookeeper	Discussion
KIP-431: Support of printing additional ConsumerRecord fields in DefaultMessageFormatter	Voting in progress, JIRA exists with pull-request
KIP-434: Add Replica Fetcher and Log Cleaner Count Metrics	Discussion
KIP-435: Incremental Partition Reassignment	Discussion
KIP-437: Custom replacement for MaskField SMT	Voting in progress, JIRA exists with pull-request
KIP-438: Expose task, connector IDs in Connect API	Discussion
KIP-439: Cleanup built-in Store interfaces	Discussion
KIP-440: Extend Connect Converter to support headers	Voting in progress
KIP-444: Augment metrics for Kafka Streams	Discussion
KIP-448: Add State Stores Unit Test Support to Kafka Streams Test Utils	Discussion
KIP-450: Sliding Window Aggregations in the DSL	Discussion
KIP-452: Tool to view cluster status	Discussion, JIRA exists
KIP-455: Create an Administrative API for Replica Reassignment	Discussion
KIP-456: Helper classes to make it simpler to write test logic with TopologyTestDriver	Discussion
KIP-457: Add DISCONNECTED status to Kafka Streams	Discussion
KIP-459: Improve KafkaStreams#close	Discussion
KIP-463: Auto-configure non-default Serdes passed alongside the TopologyBuilder	Under discussion
KIP-468: Avoid decompression of record when validate record at server in the scene of inPlaceAssignment .	Under discussion
KIP-470: TopologyTestDriver test input and output usability improvements	Under discussion
KIP-471: Expose RocksDB Metrics in Kafka Streams	Under discussion
KIP-466: Add support for List<T> serialization and deserialization	Under Discussion
KIP-473: Enable KafkaLog4JAppender to use SASL Authentication Callback Handlers	Under Discussion
KIP-474: To deprecate WindowStore#put(key, value)	Under Discussion
KIP-475: New Metric to Measure Number of Tasks on a Connector	Voting
KIP-476: Add Java AdminClient Interface	Under Discussion
KIP-477: Add PATCH method for connector config in Connect REST API	Under Discussion

Dormant/inactive KIPs

Please insert new rows in sorted order (by KIP number).

KIP	Comment
KIP-6 - New reassignment partition logic for rebalancing	Needs more detail
KIP-10 - Running Producer, Consumers and Brokers on Mesos	

KIP-14 - Tools standardization	
KIP-17 - Add HighwaterMarkOffset to OffsetFetchResponse	
KIP-23 - Add JSON/CSV output and looping options to ConsumerGroupCommand	
KIP-27 - Conditional Publish	
KIP-30 - Allow for brokers to have plug-able consensus and meta data storage sub systems	
KIP-37 - Add Namespaces to Kafka	
KIP-39: Pinning controller to broker	
KIP-44 - Allow Kafka to have a customized security protocol	
KIP-46 - Self Healing	
KIP-47 - Add timestamp-based log deletion policy	
KIP-49 - Fair Partition Assignment Strategy	
KIP-53 - Add custom policies for reconnect attempts to NetworkdClient	
KIP-59: Proposal for a kafka broker command	
KIP-64 -Allow underlying distributed filesystem to take over replication depending on configuration	
KIP-68 Add a consumed log retention before log retention	
KIP-69 - Kafka Schema Registry	Draft
KIP-76 Enable getting password from executable rather than passing as plaintext in config files	
KIP-87 - Add Compaction Tombstone Flag	Possibly displaced by KIP-82
KIP-95: Incremental Batch Processing for Kafka Streams	
KIP-269: Substitution Within Configuration Values	
KIP-349: Priorities for Source Topics	Functionality can be realized with existing API although at a lower level.

Discarded KIPs

Please insert new rows in sorted order (by KIP number).

KIP	Comment
KIP-5 - Broker Configuration Management	Superseded by KIP-21
KIP-7 - Security - IP Filtering	
KIP-9 - SSL Support	Adopted via KAFKA-1690 but not via this KIP process
KIP-18 - JBOD Support	Superseded by KIP-112 and KIP-113
KIP-24 - Remove ISR information from TopicMetadataRequest and add broker level metadata request	
KIP-29 - Add an IsrPropagateIntervalMs configuration to KafkaConfig	No longer need after KAFKA-2722
KIP-34 Add Partitioner Change Listener to Partitioner Interface for Multiple Use Case	Per feedback it is better do this parition() method and avoid Thread Coordination etc.
KIP-80: Kafka Rest Server	
KIP-83 - Allow multiple SASL authenticated Java clients in a single JVM process	After KIP-85, no interface changes, reduced to KAFKA-4180
KIP-111: Kafka should preserve the Principal generated by the PrincipalBuilder while processing the request received on socket channel, on the broker.	Covered by KIP-189
KIP-116: Add State Store Checkpoint Interval Configuration	Currently not needed as checkpointing can be done on commit interval.

KIP-127: Pluggable JAAS LoginModule configuration for SSL	Similar functionality can be implemented using a custom PrincipalBuilder.
KIP-132: Augment KStream.print to allow extra parameters in the printed string	Duplicated by KIP-160
KIP-147: Add missing type parameters to StateStoreSupplier factories and KGroupedStream/Table methods	Covered by KIP-182
KIP-165: Extend Interactive Queries for return latest update timestamp per key	Covered by KIP-258
KIP-170: Enhanced TopicCreatePolicy and introduction of TopicDeletePolicy	Superseded by KIP-201: Rationalising Policy interfaces
KIP 172: Add regular-expression topic support for sink connector	Superseded by KIP-215: Add topic regex support for Connect sinks
KIP-179: Change ReassignPartitionsCommand to use AdminClient	
KIP-184: Rename LogCleaner and related classes to LogCompactor	A lot of configuration change will cause more trouble and doesn't seems to be worth it
KIP 230: Name Windowing Joins	Covered by KIP-372: Naming Repartition Topics for Joins and Grouping
KIP-232: Detect outdated metadata using per-partition leaderEpoch field	Merged into KIP-320: Allow fetchers to detect and handle log truncation
KIP-246: Connect producers and consumers should inherit worker configs	Withdrawn because proposal could not be made backward compatible with existing behavior
KIP-248 - Create New ConfigCommand That Uses The New AdminClient	Discarded by author as it's partly covered in KIP-339 and KIP-422
KIP-262: Metadata should include number of state stores for task	Rejected since metadata is no longer required.
KIP-263: Allow broker to skip sanity check of inactive segments on broker startup	Withdrawn because the solution we agreed on does not require interface change
KIP-286: producer.send() should not block on metadata update	Withdrawn because the benefit of not having to wait for metadata is probably not worth the complexity added in producer.
KIP-288: [DISCARDED] Consumer.poll() timeout semantic change and new waitForAssignment method	Discarded in deference to KIP-266
KIP-310: Add a Kafka Source Connector to Kafka Connect	Withdrawn in favor of KIP-382.
KIP-327: Add describe all topics API to AdminClient	Withdrawn in-favor of filter support in Metadata API and KIP-142
KIP-344: The auto-generated client id should be passed to MetricsReporter	Bug fix so KIP not needed
KIP-364: Remove implicit Materialized, Consumed and Produced	Subsumed by KIP-365
KIP-404: Add Kafka Connect configuration parameter for disabling WADL output on OPTIONS request	Discarded, reported as a bug by **KAFKA-7759** - Getting issue details... STATUS
KIP-432: Additional Broker-Side Opt-In for Default, Unsecure SASL /OAUTHBEARER Implementation	A "security sanity check tool" is a more generic and appropriate solution.
KIP-451: Make TopologyTestDriver output iterable	Discarded in favor of KIP-456
KIP-472: Add header to RecordContext/ProducerRecord	Discarded due to change being far more complex than necessary for a simple bug

KIP Discussion Recordings

Date (link to recording)	Summary

2018-04-09	• KIP-253 - partition expansion: We discussed a few things. (1) Is it useful to backfill a compacted topic? The main use case is to rebuild the application states. If the new partition has the existing data, rebuilding the state can be done easily by reading from a single partition. Otherwise, an application has to read both the child and the parent partition to rebuild the state. This is possible, but can be complicated. Jan will do an exercise to see how complicated this is. (2) What's the best way to add the backfilling support if we want to do it? We can do this on the server side or on the client side. The former potentially makes the coordination easier. The latter potentially reduces the memory footprint on the server for reshuffling. We need to think through how to support EOS message format and how to throttle the process to avoid overwhelming the cluster. (3) Linear hashing vs doubling partitions. It seems that Linear hashing is more general. (4) Partition splitting in Kinesis. This is done differently since it doesn't allow customized partitioning. It doesn't support compacted topic either. (5) Sticky partition assignment. It could be useful to support a partition assignment strategy where the child partition is assigned together with the parent partition to a consumer instance so that the local state doesn't have to be moved immediately. (6) Consumer callback on partition splitting. This could still be useful if the states are maintained globally.
2017-01-07	• KIP-112 - Handle disk failure for JBOD: We discussed whether we need to support JBOD directly in Kafka or just rely on the 1 disk per broker model. The general consensus is that direct JBOD support in Kafka is needed. There is some concern on the complexity added to Kafka. So, we have to be careful with the implementation details. We discussed how directory failure should be detected, where the failure state is kept, and whether the state should be reset on broker restart. There is a bit confusing on what's written in the wiki. Dong is going to clarify the proposal based on the feedback and we will follow up on the details in the mailing list.
2016-10-19	• KIP-82 - add record header: We agreed that there are use cases for third-party vendors building tools around Kafka. We haven't reached the conclusion whether the added complexity justifies the use cases. We will follow up on the mailing list with use cases, container format people have been using, and details on the proposal.
2016-09-13	• KIP-54 (Sticky Partition Assignment): aims to minimise partition movement so that resource reinitialisation (e.g. caches) is minimised. It is partially sticky and partially fair. Some concerns around the fact that user code for partitionsRevoked and partitionsAssigned would have to be changed to work correctly with this assignment strategy. Good: more complex usage of an assigner that takes advantage of the user data field. Vahid will start the vote. • KIP-72 (Allow Sizing Incoming Request Queue in Bytes): large requests can kill the broker, no control over how much memory is allocated. Client quotas don't help as damage may already have been done by the time they kick in. There was a discussion on whether it was worth it to avoid the immediate return from select when there was no memory available in the pool. Radai will update the KIP to describe this aspect in more detail as well as the config validation that is performed. • KIP-79 (ListOffsetRequest/ListOffsetResponse v1 and add timestamp search methods to the new consumer): we discussed the option of passing multiple timestamps for the same partition in the same request. Becket thinks it's a rare use case and not worth supporting. Gwen said that it would be nice to have, but not essential. We talked about validation of duplicate topics. Becket will check the approach taken by the create topics request and evaluate if it can be adopted here too. PR will be available today and Jason will evaluate if it's feasible to include it in the next release once it's available.
2016-08-30	• KIP48 (delegation tokens): Harsha will update the wiki with more details on how to use delegation tokens and how to configure it. • KIP-78 (cluster id): There was discussion on adding human readable tags later. No major concerns.
2016-08-23	• time-based release: No one seems to have objections. Ismael will follow up with a release wiki. • KIP-4: We discussed having separate ACL requests of add and delete. No one seems to object to it. We discussed the admin client. Grant will send a PR. We discussed how KStream can use the ACL api. It seems that we will need some kind of regex or namespace support in ACL to make the authorization convenient in KStream. • KIP-50: There is some discussion for further changes in the PR. Ashish will reply to the KIP email thread with the recommended changes. Ashish/Grant plan to look into whether it's possible to make the authorizer api change backward compatible. However, it seems that people are in general ok with a non-compatible api change. • KIP-74: No objections on the current proposal. • Java 7 support timeline: The consensus is to defer dropping the Java 7 support until the next major release (which will be next year). Ismael will follow up on the email thread. • KIP-48 delegation token : Ashish will ping Harsh to see if this is still active. • Some of the KIPs have been idle. Grant will send a proposal on tagging them properly (e.g., blocked, inactive, no resource, etc).
2016-05-24	• KIP-58 - Make Log Compaction Point Configurable: We want to start with just a time-based configuration since there is no good usage for byte-based or message-based configuration. Eric will change the KIP and start the vote. • KIP-4 - Admin api: Grant will pick up the work. Initially, he plans to route the write requests from the admin clients to the controller directly to avoid having the broker forward the requests to the controller. • KIP-48 - Delegation tokens: Two of the remaining issues are (1) how to store the delegation tokens and (2) how token expiration works. Since Parth wasn't able to attend the meeting. We will follow up in the mailing list.

2016-04-05	• KIP-4: There is a slight debate on the metadata request schema, as well as the internal ZK based implementation, which we will wait for Jun to comment on the mailing list thread. • KIP-52: We decided to start a voting process for this. • KIP-35: Decided on renaming ApiVersionQuery api to ApiVersion. Consensus on using the api in java client to only check for availability of current versions. ApiVersion api's versions will not be deprecated. Update KIP-35 wiki will be updated with latest info and vote thread will be initiated.
2016-03-15	• KIP-33 - Add a time based log index to Kafka: We decided NOT to include this in 0.10.0 since the changes may have performance risks. • KIP-45 - Standardize all client sequence interaction on j.u.Collection: There is no consensus in the discussion. We will just put it to vote. • KIP-35 - Retrieving protocol version: This gets the longest discussion. There is still no consensus. Magnus thinks the current proposal of maintaining a global protocol version won't work and will try to submit a new proposal. • KIP-43 - Kafka SASL enhancements: Rajini will modify the KIP to only support native SASL mechanisms and leave the changes to Login and CallbackHandler to KIP-44 instead.
2016-02-23	• KIP-33 and KIP-47: No issues. Will start the voting thread. • KIP-43: We discussed whether there is a need to support multiple SASL mechanisms at the same time and what's the best way to implement this. Will discuss this in more details in the email thread. • KIP-4: Grant gave a comprehensive summary of the current state. We have gaps on how to make the admin request block on the broker, how to integrate admin requests with ACL (especially with respect to client config changes for throttling and ACL changes), how to do the alter topic request properly. Grant will update the KIP with an interim plan and a long term plan. • KIP-43: We briefly discussed on to support multiple sasl mechanisms on the broker. Harsha will follow up with more details on the email thread. • Everyone seems to be in favor of making the next major release 0.10.0, instead of 0.9.1.
2016-01-26	• KIP-42: We agreed to leave the broker side interceptor for another KIP. On the client side, people favor the 2nd option in Anna's proposal. Anna will update the wiki accordingly. • KIP-43: We discussed whether there is a need to support multiple SASL mechanisms at the same time and what's the best way to implement this. Will discuss this in more details in the email thread. • Jiangjie brought up an issue related to KIP-32 (adding timestamp field in the message). The issue is that currently there is no convenient way for the consumer to tell whether the timestamp in a message is the create time or the server time. He and Guozhang propose to use a bit in the message attribute to do that. Jiangjie will describe the proposal in the email thread.
2016-01-12	• KIP-41: Discussed whether the issue of long processing time between poll calls is a common issue and whether we should revisit the poll api. Also discussed whether the number of records returned in poll calls can be made more dynamic. In the end, we feel that just adding a config that controls the number records returned in poll() is the simplest approach at this moment. • KIP-36: Need to look into how to change the broker JSON representation in ZK w/o breaking rolling upgrades. Otherwise, ready for voting.
2015-10-20	• KIP-38: No concerns with this KIP. Flavio will initiate the voting on this. • KIP-37: There are questions on how ACL, configurations, etc will work, and whether we should support "move" or not. We will discuss the details more in the mailing list. • KIP-32/KIP-33: Jiangjie raised some concerns on the approach that Jay proposed. Guozhang and Jay will follow up on the mailing list.
2015-10-13	• 0.9.0 release: We discussed if KAFKA-2397 should be a blocker in 0.9.0. Jason and Guozhang will follow up on the jira. • KIP-32 and KIP-33: We discussed Jay's alternative proposal of just keeping CreateTime in the message and having a config to control how far off the CreateTime can be from the broker time. We will think a bit more on this and Jiangjie will update the KIP wiki. • KIP-36: We discussed an alternative approach of introducing a new broker property to designate the rack. It's simpler and potentially can work in the case when the broker to rack mapping is maintaining externally. We need to make sure that we have an upgrade plan for this change. Allen will update the KIP wiki
2015-10-06	• We only had the time to go through KIP-35. The consensus is that we will add a BrokerProtocolRequest that returns the supported versions for every type of requests. It's up to the client to decide how to use this. Magnus will update the KIP wiki with more details.
2015-09-22	• KIP-31: Need to figure out how to evolve inter.broker.protocol.version with multiple protocol changes within the same release, mostly for people who are deploying from trunk. Becket will update the wiki. • KIP-32/KIP-33: Having both CreateTime and LogAppendTime per message adds significant overtime. There are a couple of possibilities to improve this. Becket will follow up on this. • LinkedIn has been testing SSL in MirrorMaker (SSL is only enabled in the producer). So far, MirrorMaker can keep up with the load. LinkedIn folks will share some of the performance results.

2015-09-14	• KIP-28: Discussed the improved proposal including 2 layers of API (the higher layer is for streaming DSL), and stream time vs processor time. Ready for review. • KIP-31, KIP-32: (1) Discussed whether the timestamp should be from the client or the broker. (2) Discussed the migration path and whether this requires all consumers to upgrade before the new message format can be used. (3) Since this is too big a change, it will NOT be included in 0.9.0 release. Becket will update the wiki.
2015-08-18	• client-side assignment strategy: We discussed concerns about rebalancing time due to metadata inconsistency, especially when lots of topics are subscribed. Will discuss a bit more on the mailing list. • CopyCat data api: The discussions are in KAFKA-2367 for people who are interested. • 0.8.2.2: We want to make this a low risk bug fix release since 0.8.3 is coming. So, will only include a small number of critical and small fixes. • 0.8.3: The main features will be security and the new consumer. We will be cutting a release branch when the major pieces for these new features have been committed.
2015-08-11	• KIP-29: we will do a quick fix for unblocking production issues with hard-coded interval values, still needs further discussion on KIP itself. • KIP-26: RP-99 (https://github.com/apache/kafka/pull/99) pending for reviews. • KIP-28: RP-130 (https://github.com/apache/kafka/pull/130) needs further discussion. • KIP-4: KAFKA-1695 / 2210 pending for reviews.