What's New in Apache Kafka 2.6.0

On behalf of the Apache Kafka® community, it is my pleasure to announce the release of Apache Kafka 2.6.0. The community has created another exciting release with many new features and improvements. We'll highlight some of the more prominent features in this blog post, but see the release notes for the full list of changes.

We've made quite a few significant performance improvements in this release, particularly when the broker has larger partition counts. Broker shutdown performance is significantly improved, and performance is dramatically improved when producers use compression. Various aspects of ACL usage are faster and require less memory. And we've reduced memory allocations in several other places within the broker.

This release also adds support for Java 14. And over the past few releases, the community has switched to using Scala 2.13 by default and now recommends using Scala 2.13 for production.

Finally, these accomplishments are only one part of a larger active roadmap in the run up to Apache Kafka 3.0, which may be one of the most significant releases in the project's history. The work to replace Zookeeper with built-in Raft-based consensus is well underway with eight KIPs in active development. Kafka's new Raft protocol for the metadata quorum is already available for review. Tiered Storage unlocks infinite scaling and faster rebalance times via KIP-405, and is up and running in internal clusters at Uber.

Kafka broker, producer, and consumer

KIP-546: Add Client Quota APIs to the Admin Client

Managing quotas today in Kafka can be challenging because they can map to any combination of user and client. This feature adds a native API for managing quotas, making the process more intuitive and less error prone. A new kafka-client-quotas.sh command line tool lets users describe existing quotas, resolve the effective quotas for an entity with contextual information about how those quotas were derived, and modify a quota configuration entry by specifying which entries to add, update, and/or remove. For example:

```
$ /bin/kafka-client-quotas.sh --bootstrap-server localhost:9092 \
                              --alter --names=client-id=my-client \
                              --defaults=user \
                              --add=consumer_byte_rate=2000000 \
                              --delete=producer_byte_rate
```

See KIP-546 for more details.

KIP-551: Expose disk read and write metrics

Disk access on the Kafka broker machines may impact latency and throughput. This change adds metrics that track how many bytes Kafka is reading and writing from the disk.

See KIP-551 for more details.

KIP-568: Explicit rebalance triggering on the Consumer

The Kafka consumer coordinates which topic partitions are assigned to each client in the same consumer group. This feature allows applications using the consumer to explicitly trigger a rebalance, such as if an application uses some system condition to determine whether it is ready to receive partitions.

See KIP-568 for more details.

KIP-573: Enable TLSv1.3 by default

TLS 1.3 is now the default TLS protocol when using Java 11 or higher, and TLS 1.2 remains the default for earlier Java versions. As with Apache Kafka 2.5.0, TLS 1.0 and 1.1 are disabled by default due to known security vulnerabilities, though users can still enable them if required.

See KIP-573 for more details.

KIP-574: CLI Dynamic Configuration with file input

Kafka configs for the most part are defined by a single value that maps to a config name. Before this change, it was hard to set configs that are better defined by more complex structures such as nested lists or JSON. Kafka now supports using the kafka-configs.sh command line tool to set configs defined in a file. For example:

```
$ bin/kafka-configs.sh --bootstrap-server localhost:9092 \
                       --entity-type brokers --entity-default \
                       --alter --add-config-file new.properties
```

See KIP-574 for more details.

KIP-602: Change default value for client.dns.lookup

Apache Kafka 2.1.0 and KIP-302 introduced the use_all_dns_ips option for the client.dns.lookup client property. With this change, the use_all_dns_ips option is now the default so that it will attempt to connect to the broker using all of the possible IP addresses of a hostname. The new default will reduce connection failure rates and is

more important in cloud and containerized environments where a single hostname may resolve to multiple IP addresses.

See KIP-602 for more details.

Kafka Connect

KIP-158: Kafka Connect should allow source connectors to set topic-specific settings for new topics

This widely requested feature allows Kafka Connect to automatically create Kafka topics for source connectors that write records, if those topics do not yet exist. This is enabled by default but does require connector configurations to define the rules used by Connect when creating these topics. For example, simply including the following will cause Connect to create any missing topics with 5 partitions and a replication factor of 3:

```
topic.creation.default.replication.factor=3
topic.creation.default.partitions=5
```

Additional rules with topic matching expressions and topic-specific settings can be defined, making this a powerful and useful feature, especially when Kafka brokers have disabled topic auto creation.

See KIP-158 for more details.

KIP-605: Expand Connect Worker Internal Topic Settings

Speaking of creating topics, the Connect worker configuration can now specify additional topic settings, including using the Kafka broker defaults for partition count and replication factor, for the internal topics used for connector configurations, offsets, and status.

See KIP-605 for more details.

KIP-610: Error Reporting in Sink Connectors

Kafka Connect already had the ability to write records to a dead letter queue (DLQ) topic if those records could not be serialized or deserialized, or when a Single Message Transform (SMT) failed. Now Connect gives sink connectors the ability to send individual records to the DLQ if the connector deems the records to be invalid or problematic. Sink connectors need to explicitly make use of this feature, but doing so will allow sink connectors to continue operating even if some records in the consumed topics are somehow incompatible with the sink connector.

See KIP-610 for more details.

KIP-585: Filter and Conditional SMTs

Defining SMTs for connectors that use multiple topics can be challenging, since not every SMT may apply for every record on every topic. With this feature, each SMT can define a predicate with the conditions when that SMT should be applied. It also defines a “filter” SMT that works with the predicates to drop records that match certain conditions.

See KIP-585 for more details.

KIP-577: Allow HTTP Response Headers to be Configured for Kafka Connect

It is now possible to add custom headers to all Kafka Connect REST API responses. This allows users to ensure REST API responses comply with corporate security policies.

See KIP-577 for more details.

Kafka Streams

KIP-441: Smooth Scaling Out of Kafka Streams

Prior to this change, when Kafka Streams assigns a stateful task, Streams had to catch it up to the head of its changelog before beginning to process it. This feature avoids stop-the-world rebalances by allowing the prior owner of a stateful task to keep it even if the assignment is unbalanced, until the new owner gets caught up, then changing ownership after the catch-up phase.

See KIP-441 for more details.

KIP-444: Augment metrics for Kafka Streams

This feature adds more out-of-the-box metrics and removes some that are not useful. It also improves the APIs that Streams applications use to register custom metrics.

See KIP-444 for more details.

KIP-447: Producer scalability for exactly once semantics

This release adds additional work on this KIP to simplify the API for applications that read from and write to Kafka transactionally. Previously, this use case typically required separate producer instances for each input partition, but now there is no special requirement. This makes it much easier to build exactly-once semantics (EOS) applications that consume large numbers of partitions. This is foundational for a similar improvement in Kafka Streams in the next release.

See KIP-447 for more details.

KIP-557: Add emit on change support for Kafka Streams

This change adds an emit-on-change processing option to Kafka Streams and complements the existing emit-on-update and emit-on-window-close options. This new option drops idempotent updates where the prior and updated record have identical byte arrays. This feature helps eliminate high numbers of identical operations that forward an enormous number of unnecessary results down the topology.

See KIP-557 for more details.

Conclusion

To learn more about what's new in Apache Kafka 2.6 and to see all the KIPs included in this release, be sure to check out the release notes and highlights video.

To download Apache Kafka 2.6.0, visit the project's download page.

This was a huge community effort, so thank you to everyone who contributed to this release, including all of our users and the 127 people that contributed code or documentation changes in this release (according to git shortlog):

17hao, A. Sophie Blee-Goldman, Aakash Shah, Adam Bellemare, Agam Brahma, Alaa Zbair, Alexandra Rodoni, Andras Katona, Andrew Olson, Andy Coates, Aneel Nazareth, Anna Povzner, Antony Stubbs, Arjun Satish, Auston, avalsa, Badai Aqrandista, belugabehr, Bill Bejeck, Bob Barrett, Boyang Chen, Brian Bushree, Brian Byrne, Bruno Cadonna, Charles Feduke, Chia-Ping Tsai, Chris Egerton, Colin Patrick McCabe, Daniel, Daniel Beskin, David Arthur, David Jacot, David Mao, dengziming, Dezhi "Andy" Fang, Dima Reznik, Dominic Evans, Ego, Eric Bolinger, Evelyn Bayes, Ewen Cheslack-Postava, fantayeneh, feyman2016, Florian Hussonnois, Gardner Vickers, Greg Harris, Gunnar Morling, Guozhang Wang, high.lee, Hossein Torabi, huxi, Ismael Juma, Jason Gustafson, Jeff Huang, jeff kim, Jeff Widman, Jeremy Custenborder, Jiamei Xie, jiameixie, jiao, Jim Galasyn, Joel Hamill, John Roesler, Jorge Esteban Quilcate Otoya, José Armando García Sancio, Konstantine Karantasis, Kowshik Prakasam, Kun Song, Lee Dongjin, Leonard Ge, Lev Zemlyanov, Levani

Kokhreidze, Liam Clarke-Hutchinson, Lucas Bradstreet, Lucent-Wong, Magnus Edenhill, Manikumar Reddy, Mario Molina, Matthew Wong, Matthias J. Sax, maulin-vasavada, Michael Viamari, Michal T, Mickael Maison, Mitch, Navina Ramesh, Navinder Pal Singh Brar, nicolasguyomar, Nigel Liang, Nikolay, Okada Haruki, Paul, Piotr Fras, Radai Rosenblatt, Rajini Sivaram, Randall Hauch, Rens Groothuijsen, Richard Yu, Rigel Bezerra de Melo, Rob Meng, Rohan, Ron Dagostino, Sanjana Kaundinya, Scott, Scott Hendricks, sebwills, Shailesh Panwar, showuon, SoontaekLim, Stanislav Kozlovski, Steve Rodrigues, Svend Vanderveken, Sönke Liebau, THREE LEVEL HELMET, Tom Bentley, Tu V. Tran, Valeria, Vikas Singh, Viktor Somogyi, vinoth chandar, Vito Jeng, Xavier Léauté, xiaodongdu, Zach Zhang, zhaohaidao, zshuo, 阿洋