KIP-1004: Enforce tasks.max property in Kafka Connect

Status

Current state: *Accepted*

Discussion thread: https://lists.apache.org/thread/scx75cjwm19jyt19wxky41q9smf5nx6d

Vote thread: https://lists.apache.org/thread/dgq332o5j25rwddbvfydf7ttrclldw17

JIRA: KAFKA-15575

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

Motivation

Since its inception, Kafka Connect has supported horizontal scaling for connectors by allowing them to generate one or more "tasks", each of which is responsible for handling a disjoint subset of the workload for the connector. Users can specify an upper bound for the number of tasks generated by a connector by setting the `tasks.max` property in connector configurations. This value is then passed to connectors when invoking their taskConfigs method , which, as the name suggests, returns a list of task configurations. Each task configuration corresponds to a task that should be brought up on the Kafka Connect cluster.

Also since its inception, Kafka Connect has not actually enforced that the list of task configurations generated by connectors respects the user-specified value for the `tasks.max` property. It is unclear if this was intentional or due to oversight during implementation, but the benefits of enforcing this property have increased over time:

- As Kafka Connect has matured, focus on security and robustness has increased: a hostile or buggy connector that generates too many task configurations is a serious threat to a Kafka Connect cluster
- With KIP-987: Connect Static Assignments, users and Kafka Connect maintainers will need to be able to trust that there is an upper bound on the number of tasks generated by a connector
  - Though this KIP is currently up for discussion, it still serves as an example of how being able to rely on the `tasks.max` property not as a suggestion but as a guarantee aids us in developing new features for Kafka Connect

Public Interfaces

Next minor (x.y.0) release

When a Connector generates an excessive number of tasks, none of the generated tasks will be brought up, and the Connector will be marked `FAILED`. An error message explaining why the Connector has failed will be included in the status for the connector, where users will also be encouraged to report this behavior as a bug to the maintainers of the connector, and instructions on how to disable this logic (detailed below) will be provided.

If the connector generated excessive tasks after being reconfigured, then any existing tasks for the connector will be allowed to continue running, unless that existing set of tasks also exceeds the `tasks.max` property.

If an existing set of tasks for any connector exceeds the maximum-permitted number of tasks for it, all tasks for the connector will be marked `FAILED` with a similar error message to the one used to fail the Connector.

A new connector property will be introduced as an emergency measure to disable this behavior:

Name	Type	Default	Importance	Description (sample; subject to change during code review)

tasks.max.enforce	boolean	true	low	(Deprecated) Whether to enforce that the tasks.max property is respected by the connector. By default, connectors that generate too many tasks will fail, and existing sets of tasks that exceed the tasks.max property will also be failed. If this property is set to false, then connectors will be allowed to generate more than the maximum number of tasks, and existing sets of tasks that exceed the tasks.max property will be allowed to run. This property is deprecated and will be removed in an upcoming major release.

A future major (x.0.0) release

In an upcoming major release, the `tasks.max.enforce` property will be removed and it will become impossible to disable enforcement of the `tasks.max` property.

Assuming the KIP is implemented and released in 3.8.0, this may take place in 4.0.0, but may also be delayed until 5.0.0 or a subsequent major version bump in order to allow users to more gracefully adapt to the change in runtime behavior by Kafka Connect.

All patch (x.y.z, z >= 1) releases on older versions

All older branches that are not yet EOL and that do not enforce the `tasks.max` property will be updated to emit warning messages to users when connectors generate excessive tasks. These messages will state that the connector should be considered buggy, that it may not work with future releases of Kafka Connect, and that users should report this behavior to the maintainers of the connector.

Compatibility, Deprecation, and Migration Plan

All connectors that respect the `tasks.max` property will be unaffected by this change. Given how fundamental the `tasks.max` property and the `taskConfigs` Connector method are, this should be the overwhelming majority of connectors in the Kafka Connect ecosystem today.

All connectors that do not respect the `tasks.max` property will begin to fail once Kafka Connect clusters are upgraded to a version that includes the changes from this KIP. As an emergency measure, users will be able to temporarily rescue these connectors by setting the `tasks.max.enforce` property to false in their connector configurations.

Test Plan

Integration testing will cover these scenarios:

- A connector that generates excessive tasks will be failed with an expected error message
- An existing set of tasks that exceeds the `tasks.max` property (this will be simulated during testing by manually writing task configs to the config topic before the Kafka Connect cluster is started–dirty, but fun!) will be failed with an expected error message
- That same existing set of tasks will be allowed to run (possibly after being manually restarted) once the connector is reconfigured with `tasks.max.enforce` set to false
- A connector will be allowed to generate excessive tasks when `tasks.max.enforce` is set to false
- A connector that generates excessive tasks after being reconfigured will be failed, but its existing tasks will continue running

Rejected Alternatives

Opt-in to enforcement

Summary: Set the default of the `tasks.max.enforce` property to false instead of true, then change the default to true in a major release, then drop it in the next major release.

Rejected because: This approach is somewhat safer, but it comes at the cost of having to wait for two major releases to take place before we can guarantee that the `tasks.max` property is respected by connectors, which would severely delay work (such as KIP-987) that relies on this. Given how rare it should be for connectors to have buggy `taskConfigs` implementations, the reduced risk of this approach is not a large-enough benefit to warrant its cost.

Drop tasks instead of failing connectors

Summary: Instead of failing the connector, bring up only the maximum-permitted number of tasks, silently (with the possible exception of a logged warning message) dropping the rest.

Rejected because: While it's true that most sink connectors will be able to transparently adjust to a reduced set of task configs, it's not guaranteed that all will: some sink connectors may assign special responsibilities to the Nth task, like periodically triggering flushes from a shared buffer to the external system. On the source connector side, silently halting the flow of data for a subset of tasks with no notice besides a warning message in logs seems likely to lead to headaches for users that see the flow of data suddenly stop but no other obvious indications of failure.