

KIP-449: Add connector contexts to Connect worker logs

- [Status](#)
- [Motivation](#)
- [Public Interfaces](#)
 - [Example Log File](#)
- [Proposed Changes](#)
- [Compatibility, Deprecation, and Migration Plan](#)
- [Rejected Alternatives](#)

Status

Current state: *Adopted*

Discussion thread: [here](#)

Vote thread: [here](#)

JIRA: [KAFKA-3816](#)

Released: *AK 2.3.0*

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

Motivation

Kafka Connect logs contain a mixture of content from the Connect worker's multiple threads, and it is difficult to easily identify the context of the log messages. The main worker thread, for example, is responsible for processing REST requests, validating connector configurations, and starting/stopping/updating connectors and starting/stopping tasks. The worker uses a separate thread for each assigned task to call the task methods and to call the producer and consumer methods. Finally, a separate task is used for scheduled operations, such as committing source offsets.

Including more context on each log message will dramatically improve the ability to understand what is happening within the worker, a particular connector, or a particular connector task. This context should be of a form that is easily parseable, so users can use simple tools to select only those lines that apply to specific contexts. This mechanism should also work for log messages output by connector, transform, and converter implementations, even though the Connect API has not provided a central mechanism for such implementations to obtain the logger components. Finally, these additional contexts should not be used by default so that existing users upgrading to a newer version of Apache Kafka do not have unexpected changes in their Connect log files.

Although it's not clear whether the logs or log configuration files are a formal public API, this proposal includes behavioral changes that can affect users' experiences.

Public Interfaces

Kafka Connect uses the SLF4J API for logging and the Log4J logging implementation. The `config/connect-log4j.properties` file included in the Kafka distribution defines the Log4J configuration used in Connect's distributed and standalone worker processes. This configuration file contains the following line that defines the layout pattern for Connect log files:

Existing line in the current `config/connect-log4j.properties`

```
...
log4j.appender.stdout.layout.ConversionPattern=[%d] %p %m (%c:%L)%n
...
```

A new comment will be added to the file immediately preceding the above line to help describe how to add the new Connect log context information in the layout pattern:

New comment to be added in the `config/connect-log4j.properties`

```
#
# The `%X{connector.context}` parameter in the layout includes connector-specific and task-specific information
# in the log message, where appropriate. This makes it easier to identify those log messages that apply to a
# specific connector. Simply add this parameter to the log layout configuration below to include the contextual
# information.
#
#log4j.appender.stdout.layout.ConversionPattern=[%d] %p %X{connector.context}%m (%c:%L)%n
```

By default, the log configuration will remain as in the previous release. However, when a user modifies the `config/connect-log4j.properties` file and adds the `%X{connector.context}` to the layout (as suggested in the comment), the Connect worker process will write each log message and replace this placeholder with the following information:

```
[<connectorName>|<scope>]<sp>
```

where "`<connectorName>`" is the name of the connector, "`<sp>`" indicates a trailing space, and "`<scope>`" is one of the following:

- "`task-<n>`" for the operation of the numbered task, including calling the task methods and the producer/consumer; here "`n`" is the 0-based task number
- "`task-<n>|offset`" for the committing of source offsets for the numbered task; here "`n`" is the zero-based task number
- "`worker`" for the creation and usage of connector instances

The following are examples of the connector context for a connector named "`my-connector`":

- ``[my-connector|worker]`` - used on log messages where the Connect worker is validating the configuration for or starting/stopping the "my-connector" connector via the `SourceConnector` / `SinkConnector` implementation methods.
- ``[my-connector|task-0]`` - used on log messages where the Connect worker is executing task 0 of the "my-connector" connector, including calling any of the `SourceTask` / `SinkTask` implementation methods, processing the messages for/from the task, and calling the task's producer / consumer.
- ``[my-connector|task-0|offsets]`` - used on log messages where the Connect worker is committing source offsets for task 0 of the "my-connector" connector.

Some of the worker operations are not within the context of a single connector, and thus will not include no connector context information in the associated log messages. This includes worker startup, REST request processing, and shutdown.

Example Log File

The following is an abbreviated portion of a sample Connect log with the connector context enabled in the Log4J logs. Note how all of the log lines for the "local-file-source" connector (including those from the worker, the tasks, and the offset commits) have a context beginning with "[local-file-source]", and all log messages for task 0 (the first task) of that same connector (including the calls to the connector task implementation, the processing of messages, and the writing of messages to Kafka) have a context beginning with "[local-file-source|task-0]". These contexts make it easy to select a subset of the log messages for a single connector and/or task, even if the worker is also running many other connector tasks.

Portion of a log file with connector contexts

```
...
[2019-04-02 17:01:38,178] INFO REST server listening at http://10.0.1.5:8083/, advertising URL http://10.0.1.5:8083/ (org.apache.kafka.connect.runtime.rest.RestServer:219)
[2019-04-02 17:01:38,178] INFO Advertised URI: http://10.0.1.5:8083/ (org.apache.kafka.connect.runtime.rest.RestServer:271)
[2019-04-02 17:01:38,187] INFO Kafka version: 2.3.0-SNAPSHOT (org.apache.kafka.common.utils.AppInfoParser:109)
[2019-04-02 17:01:38,187] INFO Kafka commitId: da2bddb1331e740c (org.apache.kafka.common.utils.AppInfoParser:110)
...
[2019-04-02 17:01:38,306] INFO Worker started (org.apache.kafka.connect.runtime.Worker:168)
[2019-04-02 17:01:38,306] INFO Herder started (org.apache.kafka.connect.runtime.standalone.StandaloneHerder:89)
[2019-04-02 17:01:38,306] INFO Kafka Connect started (org.apache.kafka.connect.runtime.Connect:55)
[2019-04-02 17:01:38,314] INFO [local-file-source|worker] ConnectorConfig values:
    config.action.reload = restart
    connector.class = FileStreamSource
    errors.log.enable = false
    errors.log.include.messages = false
    errors.retry.delay.max.ms = 60000
    errors.retry.timeout = 0
    errors.tolerance = none
    header.converter = null
    key.converter = null
    name = local-file-source
    tasks.max = 1
    transforms = []
    value.converter = null
    (org.apache.kafka.connect.runtime.ConnectorConfig:279)
[2019-04-02 17:01:38,315] INFO [local-file-source|worker] Creating connector local-file-source of type FileStreamSource (org.apache.kafka.connect.runtime.Worker:227)
[2019-04-02 17:01:38,317] INFO [local-file-source|worker] Instantiated connector local-file-source with version 2.3.0-SNAPSHOT of type class org.apache.kafka.connect.file.FileStreamSourceConnector (org.apache.kafka.connect.runtime.Worker:230)
[2019-04-02 17:01:38,318] INFO [local-file-source|worker] AbstractConfig values:
    batch.size = 2000
    file = test.txt
```

```

        topic = [connect-test]
        (org.apache.kafka.common.config.AbstractConfig:279)
[2019-04-02 17:01:38,318] INFO [local-file-source|worker] Finished creating connector local-file-source (org.
apache.kafka.connect.runtime.Worker:249)
[2019-04-02 17:01:38,320] INFO [local-file-source|task-0] Creating task local-file-source-0 (org.apache.kafka.
connect.runtime.Worker:395)
[2019-04-02 17:01:38,320] INFO [local-file-source|task-0] ConnectorConfig values:
    config.action.reload = restart
    connector.class = FileStreamSource
    errors.log.enable = false
    errors.log.include.messages = false
    errors.retry.delay.max.ms = 60000
    errors.retry.timeout = 0
    errors.tolerance = none
    header.converter = null
    key.converter = null
    name = local-file-source
    tasks.max = 1
    transforms = []
    value.converter = null
    (org.apache.kafka.connect.runtime.ConnectorConfig:279)
[2019-04-02 17:01:38,324] INFO [local-file-source|task-0] TaskConfig values:
    task.class = class org.apache.kafka.connect.file.FileStreamSourceTask
    (org.apache.kafka.connect.runtime.TaskConfig:279)
[2019-04-02 17:01:38,324] INFO [local-file-source|task-0] Instantiated task local-file-source-0 with version
2.3.0-SNAPSHOT of type org.apache.kafka.connect.file.FileStreamSourceTask (org.apache.kafka.connect.runtime.
Worker:409)
[2019-04-02 17:01:38,324] INFO [local-file-source|task-0] JsonConverterConfig values:
    converter.type = key
    schemas.cache.size = 1000
    schemas.enable = true
    (org.apache.kafka.connect.json.JsonConverterConfig:279)
[2019-04-02 17:01:38,325] INFO [local-file-source|task-0] Set up the key converter class org.apache.kafka.
connect.json.JsonConverter for task local-file-source-0 using the worker config (org.apache.kafka.connect.
runtime.Worker:422)
[2019-04-02 17:01:38,325] INFO [local-file-source|task-0] JsonConverterConfig values:
    converter.type = value
    schemas.cache.size = 1000
    schemas.enable = true
    (org.apache.kafka.connect.json.JsonConverterConfig:279)
[2019-04-02 17:01:38,325] INFO [local-file-source|task-0] Set up the value converter class org.apache.kafka.
connect.json.JsonConverter for task local-file-source-0 using the worker config (org.apache.kafka.connect.
runtime.Worker:428)
[2019-04-02 17:01:38,325] INFO [local-file-source|task-0] Set up the header converter class org.apache.kafka.
connect.storage.SimpleHeaderConverter for task local-file-source-0 using the worker config (org.apache.kafka.
connect.runtime.Worker:435)
[2019-04-02 17:01:38,329] INFO [local-file-source|task-0] Initializing: org.apache.kafka.connect.runtime.
TransformationChain{} (org.apache.kafka.connect.runtime.Worker:487)
[2019-04-02 17:01:38,355] INFO [local-file-source|task-0] Kafka version: 2.3.0-SNAPSHOT (org.apache.kafka.
common.utils.AppInfoParser:109)
[2019-04-02 17:01:38,355] INFO [local-file-source|task-0] Kafka commitId: da2bddb1331e740c (org.apache.kafka.
common.utils.AppInfoParser:110)
[2019-04-02 17:01:38,361] INFO [local-file-source|task-0] WorkerSourceTask{id=local-file-source-0} Source task
finished initialization and start (org.apache.kafka.connect.runtime.WorkerSourceTask:200)
[2019-04-02 17:01:38,361] INFO Created connector local-file-source (org.apache.kafka.connect.cli.
ConnectStandalone:110)
[2019-04-02 17:01:38,364] INFO [local-file-sink|task-0] Creating task local-file-sink-0 (org.apache.kafka.
connect.runtime.Worker:395)
[2019-04-02 17:01:38,364] INFO [local-file-sink|task-0] ConnectorConfig values:
    config.action.reload = restart
    connector.class = FileStreamSink
    ...
    transforms = []
    value.converter = null
    (org.apache.kafka.connect.runtime.ConnectorConfig:279)
[2019-04-02 17:01:38,365] INFO [local-file-sink|task-0] TaskConfig values:
    task.class = class org.apache.kafka.connect.file.FileStreamSinkTask
    (org.apache.kafka.connect.runtime.TaskConfig:279)
[2019-04-02 17:01:38,365] INFO [local-file-sink|task-0] Instantiated task local-file-sink-0 with version 2.3.0-
SNAPSHOT of type org.apache.kafka.connect.file.FileStreamSinkTask (org.apache.kafka.connect.runtime.Worker:409)
[2019-04-02 17:01:38,365] INFO [local-file-sink|task-0] JsonConverterConfig values:

```

```

        converter.type = key
        schemas.cache.size = 1000
        schemas.enable = true
    (org.apache.kafka.connect.json.JsonConverterConfig:279)
[2019-04-02 17:01:38,365] INFO [local-file-sink|task-0] Set up the key converter class org.apache.kafka.connect.
json.JsonConverter for task local-file-sink-0 using the worker config (org.apache.kafka.connect.runtime.Worker:
422)
[2019-04-02 17:01:38,365] INFO [local-file-sink|task-0] JsonConverterConfig values:
    converter.type = value
    schemas.cache.size = 1000
    schemas.enable = true
    (org.apache.kafka.connect.json.JsonConverterConfig:279)
[2019-04-02 17:01:38,366] INFO [local-file-sink|task-0] Set up the value converter class org.apache.kafka.
connect.json.JsonConverter for task local-file-sink-0 using the worker config (org.apache.kafka.connect.runtime.
Worker:428)
[2019-04-02 17:01:38,366] INFO [local-file-sink|task-0] Set up the header converter class org.apache.kafka.
connect.storage.SimpleHeaderConverter for task local-file-sink-0 using the worker config (org.apache.kafka.
connect.runtime.Worker:435)
[2019-04-02 17:01:38,366] INFO [local-file-sink|task-0] Initializing: org.apache.kafka.connect.runtime.
TransformationChain{} (org.apache.kafka.connect.runtime.Worker:501)
[2019-04-02 17:01:38,366] INFO [local-file-sink|task-0] SinkConnectorConfig values:
    config.action.reload = restart
    connector.class = FileStreamSink
...
    transforms = []
    value.converter = null
    (org.apache.kafka.connect.runtime.SinkConnectorConfig:279)
[2019-04-02 17:01:38,372] INFO [local-file-sink|task-0] ConsumerConfig values:
    auto.commit.interval.ms = 5000
    auto.offset.reset = earliest
    bootstrap.servers = [localhost:9092]
    check.crcs = true
...
    ssl.truststore.location = null
    ssl.truststore.password = null
    ssl.truststore.type = JKS
    value.deserializer = class org.apache.kafka.common.serialization.ByteArrayDeserializer
    (org.apache.kafka.clients.consumer.ConsumerConfig:279)
[2019-04-02 17:01:38,398] INFO [local-file-sink|task-0] Kafka version: 2.3.0-SNAPSHOT (org.apache.kafka.common.
utils.AppInfoParser:109)
[2019-04-02 17:01:38,398] INFO [local-file-sink|task-0] Kafka commitId: da2bddb1331e740c (org.apache.kafka.
common.utils.AppInfoParser:110)
[2019-04-02 17:01:38,402] INFO Created connector local-file-sink (org.apache.kafka.connect.cli.
ConnectStandalone:110)
[2019-04-02 17:01:38,402] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Subscribed to topic(s): connect-test (org.apache.kafka.clients.consumer.KafkaConsumer:943)
[2019-04-02 17:01:38,403] INFO [local-file-sink|task-0] WorkerSinkTask{id=local-file-sink-0} Sink task finished
initialization and start (org.apache.kafka.connect.runtime.WorkerSinkTask:301)
[2019-04-02 17:01:38,418] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Cluster ID: BmIDH-sOQ0OshXX877eSrw (org.apache.kafka.clients.Metadata:274)
[2019-04-02 17:01:38,418] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Discovered group coordinator 10.0.1.5:9092 (id: 2147483647 rack: null) (org.apache.kafka.clients.
consumer.internals.AbstractCoordinator:679)
[2019-04-02 17:01:38,420] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Revoking previously assigned partitions [] (org.apache.kafka.clients.consumer.internals.
ConsumerCoordinator:467)
[2019-04-02 17:01:38,420] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] (Re-)joining group (org.apache.kafka.clients.consumer.internals.AbstractCoordinator:492)
[2019-04-02 17:01:38,427] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] (Re-)joining group (org.apache.kafka.clients.consumer.internals.AbstractCoordinator:492)
[2019-04-02 17:01:38,451] INFO [local-file-source|task-0] [Producer clientId=producer-1] Cluster ID: BmIDH-
sOQ0OshXX877eSrw (org.apache.kafka.clients.Metadata:274)
[2019-04-02 17:01:38,481] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Successfully joined group with generation 5 (org.apache.kafka.clients.consumer.internals.
AbstractCoordinator:456)
[2019-04-02 17:01:38,483] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Setting newly assigned partitions: connect-test-0 (org.apache.kafka.clients.consumer.internals.
ConsumerCoordinator:273)
[2019-04-02 17:01:38,487] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Setting offset for partition connect-test-0 to the committed offset 2 (org.apache.kafka.clients.
consumer.internals.ConsumerCoordinator:511)

```

```

...
[2019-04-02 17:01:40,901] INFO [local-file-source|task-0|offsets] WorkerSourceTask{id=local-file-source-0}
Committing offsets (org.apache.kafka.connect.runtime.WorkerSourceTask:398)
[2019-04-02 17:01:40,901] INFO [local-file-source|task-0|offsets] WorkerSourceTask{id=local-file-source-0}
flushing 0 outstanding messages for offset commit (org.apache.kafka.connect.runtime.WorkerSourceTask:415)
...
[2019-04-02 17:11:46,385] INFO Kafka Connect stopping (org.apache.kafka.connect.runtime.Connect:65)
[2019-04-02 17:11:46,385] INFO Stopping REST server (org.apache.kafka.connect.runtime.rest.RestServer:227)
[2019-04-02 17:11:46,391] INFO Stopped http_8083@6f099cef{HTTP/1.1,[http/1.1]}{0.0.0.0:8083} (org.eclipse.jetty.
server.AbstractConnector:341)
[2019-04-02 17:11:46,391] INFO node0 Stopped scavenging (org.eclipse.jetty.server.session:167)
[2019-04-02 17:11:46,397] INFO Stopped o.e.j.s.ServletContextHandler@2b9b7f1f{/,null,UNAVAILABLE} (org.eclipse.
jetty.server.handler.ContextHandler:1045)
[2019-04-02 17:11:46,398] INFO REST server stopped (org.apache.kafka.connect.runtime.rest.RestServer:245)
[2019-04-02 17:11:46,398] INFO Herder stopping (org.apache.kafka.connect.runtime.standalone.StandaloneHerder:94)
[2019-04-02 17:11:46,398] INFO [local-file-source|task-0] Stopping task local-file-source-0 (org.apache.kafka.
connect.runtime.Worker:590)
[2019-04-02 17:11:46,398] INFO [local-file-source|task-0] WorkerSourceTask{id=local-file-source-0} Committing
offsets (org.apache.kafka.connect.runtime.WorkerSourceTask:398)
[2019-04-02 17:11:46,398] INFO [local-file-source|task-0] WorkerSourceTask{id=local-file-source-0} flushing 0
outstanding messages for offset commit (org.apache.kafka.connect.runtime.WorkerSourceTask:415)
[2019-04-02 17:11:46,399] INFO [local-file-source|task-0] [Producer clientId=producer-1] Closing the Kafka
producer with timeoutMillis = 30000 ms. (org.apache.kafka.clients.producer.KafkaProducer:1141)
[2019-04-02 17:11:46,401] INFO [local-file-source|worker] Stopping connector local-file-source (org.apache.
kafka.connect.runtime.Worker:334)
[2019-04-02 17:11:46,401] INFO [local-file-source|worker] Stopped connector local-file-source (org.apache.kafka.
connect.runtime.Worker:350)
[2019-04-02 17:11:46,402] INFO [local-file-sink|task-0] Stopping task local-file-sink-0 (org.apache.kafka.
connect.runtime.Worker:590)
[2019-04-02 17:11:46,402] INFO [local-file-sink|task-0] [Consumer clientId=consumer-1, groupId=connect-local-
file-sink] Member consumer-1-d366d86d-e0d5-4355-9cde-3354fcfa5b64 sending LeaveGroup request to coordinator
10.0.1.5:9092 (id: 2147483647 rack: null) (org.apache.kafka.clients.consumer.internals.AbstractCoordinator:826)
[2019-04-02 17:11:46,408] INFO [local-file-sink|worker] Stopping connector local-file-sink (org.apache.kafka.
connect.runtime.Worker:334)
[2019-04-02 17:11:46,408] INFO [local-file-sink|worker] Stopped connector local-file-sink (org.apache.kafka.
connect.runtime.Worker:350)
[2019-04-02 17:11:46,408] INFO Worker stopping (org.apache.kafka.connect.runtime.Worker:175)
[2019-04-02 17:11:46,409] INFO Stopped FileOffsetBackingStore (org.apache.kafka.connect.storage.
FileOffsetBackingStore:66)
[2019-04-02 17:11:46,409] INFO Worker stopped (org.apache.kafka.connect.runtime.Worker:196)
[2019-04-02 17:11:46,409] INFO Herder stopped (org.apache.kafka.connect.runtime.standalone.StandaloneHerder:111)
[2019-04-02 17:11:46,409] INFO Kafka Connect stopped (org.apache.kafka.connect.runtime.Connect:70)

```

Proposed Changes

Other than the changes to the Connect Log4J configuration file and the log file content mentioned above, the only other changes are to the Connect runtime.

The SLF4J API includes "Mapped Diagnostic Contexts" (MDC) that allow injection of a series of parameters that can be included in *every* log message written using that thread, regardless of how the SLF4J *Logger* instance was obtained. The MDC parameters are thread-local state that can be changed at any time, and various parts of the Connect runtime codebase will be changed to set a single "connector.context" MDC parameter with the relevant connector name and scope, as defined above. A new *LoggingContext* class will be added to the Connect runtime (not exposed in the public API) and will implement *AutoCloseable* to make it easy to set the context and restore the old context using try-with-resource blocks. For example:

Sample use of *LoggingContext* in the Connect runtime

```

ConnectorTaskId taskId = ...
try (LoggingContext ctx = LoggingContext.forTask(taskId)) {
    log.info("Creating task {}", id);
    ...
}

```

Compatibility, Deprecation, and Migration Plan

The existing `config/connect-log4j.properties` file is modified only to add a comment describing how to use the ``%X{connector.context}`` MDC parameter in the log format, so that users can more easily enable the additional information. However, users must modify this `config/connect-log4j.properties` file in order to have this additional context information in the Connect logs.

Rejected Alternatives

One alternative considered was to use the existing `"org.apache.kafka.common.utils.LogContext"` utility that is in the clients JAR. Although it would be simpler to reuse an existing mechanism, the existing `LogContext` doesn't satisfy the requirements in two specific ways:

1. its prefix is fixed; and
2. it only affects log messages written by `Logger` instances instantiated with the `LogContext`.

This works great in a Kafka client like the producer or consumer because the client has a single fixed ID and the Apache Kafka code that creates the `Logger` instances can all be changed to use `LogContext`. Connect is quite different, because the connector implementations would also have to change how they create their `Logger` instances, and because there is some code in Connect where a single thread does switch contexts from one connector/task to another. MDC, on the other hand, is designed to make the additional context state available to **all** log messages written by **all** components by being thread-based rather than based upon the `Logger` instance that is being used.