KIP-984: Add pluggable compression interface to Kafka

Status

Current state: *["Under Discussion"]*

Discussion thread: *here*

JIRA: *here [Change the link from KAFKA-1 to your own ticket]*

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

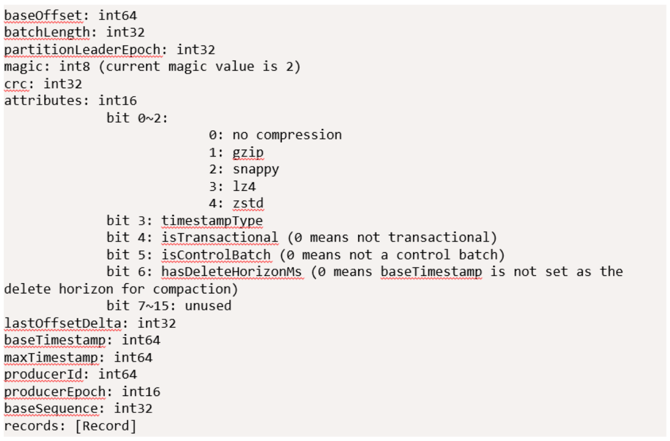
Motivation

Although compression is not a new problem, it has continued to be an important research topic.

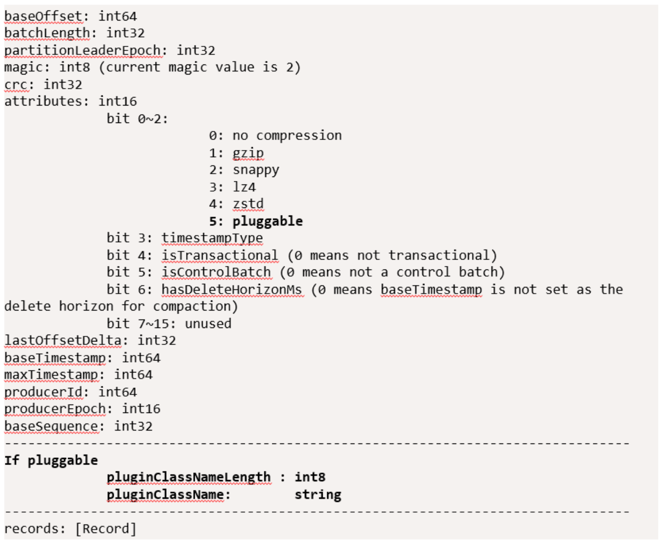
The integration and testing of new compression algorithms into Kafka currently requires significant code changes and rebuilding of the distribution package for Kafka.

This KIP give users the ability to use different compressors without needing future changes in Kafka.

This proposal suggest modifying the message format by adding a new attribute for compression.



```
baseOffset: int64
batchLength: int32
partitionLeaderEpoch: int32
magic: int8 (current magic value is 2)
crc: int32
attributes: int16
             bit 0~2:
                          0: no compression
                          1: gzip
                          2: snappy
                          3: lz4
                          4: zstd
             bit 3: timestampType
             bit 4: isTransactional (0 means not transactional)
             bit 5: isControlBatch (0 means not a control batch)
             bit 6: hasDeleteHorizonMs (0 means baseTimestamp is not set as the
delete horizon for compaction)
             bit 7~15: unused
lastOffsetDelta: int32
baseTimestamp: int64
maxTimestamp: int64
producerId: int64
producerEpoch: int16
baseSequence: int32
records: [Record]
```



```
baseOffset: int64
batchLength: int32
partitionLeaderEpoch: int32
magic: int8 (current magic value is 2)
crc: int32
attributes: int16
             bit 0~2:
                          0: no compression
                          1: gzip
                          2: snappy
                          3: lz4
                          4: zstd
                          5: pluggable
             bit 3: timestampType
             bit 4: isTransactional (0 means not transactional)
             bit 5: isControlBatch (0 means not a control batch)
             bit 6: hasDeleteHorizonMs (0 means baseTimestamp is not set as the
delete horizon for compaction)
             bit 7~15: unused
lastOffsetDelta: int32
baseTimestamp: int64
maxTimestamp: int64
producerId: int64
producerEpoch: int16
baseSequence: int32
--------------------------------------------------------------------------------
If pluggable
             pluginClassNameLength : int8
             pluginClassName:        string
--------------------------------------------------------------------------------
records: [Record]
```

This KIP does not supercede KIP-390 and KIP-780. It can build on top of KIP-780 as the scope of this KIP goes beyond the scope of KIP-390/KIP-780

Public Interfaces

Current compression is NOT changed. The existing codecs are not affected because their code path is NOT affected.

This feature introduces a new option, 'pluggable.factory.plugin' to the producer, topic and broker configuration. The type of this option is a string, with default value of null

compression.type=pluggable

pluggable.factory.plugin=<pluginClassname path>

The following public packages are affected:

- org/apache/kafka/common
- org/apache/kafka/clients/producer
- org/apache/kafka/clients/consumer

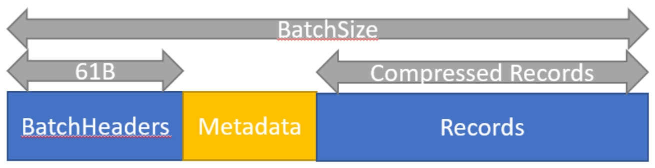
Proposed Changes

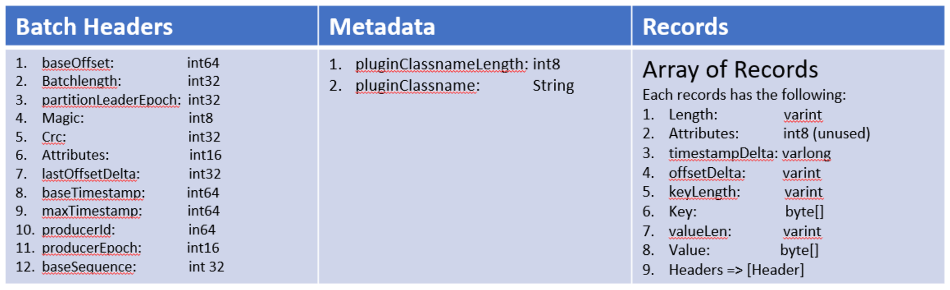
This option doesn't change the following processes because pluggable compression is implemented as a complimentary new codec:

- Producer compresses the batch of user-given messages
- Broker decompresses the batch when validating messages
- Broker recompresses the batch when compression of producer different from broker
- Producer decompresses the batch of user-given messages

When the pluggable interface is used, metadata is added to the message format to indicate the plugin class name path.

Add metadata to batch of records





Batch Headers	Metadata	Records
1. baseOffset: int64	1. pluginClassnameLength: int8	Array of Records
2. Batchlength: int32	2. pluginClassname: String	Each records has the following:
3. partitionLeaderEpoch: int32		1. Length: varint
4. Magic: int8		2. Attributes: int8 (unused)
5. Crc: int32		3. timestampDelta: varlong
6. Attributes: int16		4. offsetDelta: varint
7. lastOffsetDelta: int32		5. keyLength: varint
8. baseTimestamp: int64		6. Key: byte[]
9. maxTimestamp: int64		7. valueLen: varint
10. producerId: in64		8. Value: byte[]
11. producerEpoch: int16		9. Headers => [Header]
12. baseSequence: int 32		

Compatibility, Deprecation, and Migration Plan

- *What impact (if any) will there be on existing users?*
- *If we are changing behavior how will we phase out the older behavior?*
- *If we need special migration tools, describe them here.*
- *When will we remove the existing behavior?*

1. The current proposal does not have any impact on existing users.
2. We are not changing the behavior of current compression. We are simply adding a new codec.
3. If message format changes in Kafka then this implementation will need to be updated
4. Regression testing show pluggable interface did not affect default performance using Snappy as a plugin

Rejected Alternatives

If there are alternative ways of accomplishing the same thing, what were they? The purpose of this section is to motivate why the design is the way it is and not some other way.