# KIP-871: Fix ByteBufferSerializer#serialize(String, ByteBuffer) compatible problem


- Status
- Motivation
- Public Interfaces
- Proposed Changes
- Test Plan


## Status

**Current state**: *"Discard"*

**Discussion thread**: https://lists.apache.org/thread/l2zkd375818gkg5753xqhcqf4boqhbqm

**JIRA**: Unable to render Jira issues macro, execution error.

**PR:** https://github.com/apache/kafka/pull/12683

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

## Motivation

*Currently `ByteBufferSerializer#serialize(String, ByteBuffer)` has compatible problem, If the ByteBuffer#capacity is 7 and only has 5 bytes data `ByteBufferSerializer#serialize(String, ByteBuffer)` will return all 7 bytes in the ByteBuffer instead of valid 5 bytes*

```
@Test
public void testByteBufferSerializer() {
    final byte[] bytes = "Hello".getBytes(UTF_8);
    final ByteBuffer buffer = ByteBuffer.allocate(7);
    buffer.put(bytes);

    try (final ByteBufferSerializer serializer = new ByteBufferSerializer()) {
        assertArrayEquals(bytes, serializer.serialize(topic, buffer));
    }
}
```

Executing the above test case will throw the following exception:

```
array lengths differ, expected: <5> but was: <7>
Expected :5
Actual   :7
<Click to see difference>org.opentest4j.AssertionFailedError: array lengths differ, expected: <5> but was: <7>
    ...
    at org.apache.kafka.common.serialization.SerializationTest.testByteBufferSerializer(SerializationTest.java:
397)
    ...
    at java.util.ArrayList.forEach(ArrayList.java:1259)
    ...
    at java.util.ArrayList.forEach(ArrayList.java:1259)
    ...
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.run(GradleWorkerMain.java:69)
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.main(GradleWorkerMain.java:74)
```

## Public Interfaces

There are no new interfaces and no existing interfaces that will be removed or changed.

## Proposed Changes

Change the implement of `*ByteBufferSerializer#serialize(String, ByteBuffer)*`:

```
package org.apache.kafka.common.serialization;

import org.apache.kafka.common.utils.Utils;

import java.nio.ByteBuffer;

public class ByteBufferSerializer implements Serializer<ByteBuffer> {

    @Override
    public byte[] serialize(String topic, ByteBuffer data) {
        if (data == null) {
            return null;
        }

        if (data.hasArray()) {
            final byte[] arr = data.array();
            if (data.arrayOffset() == 0 && arr.length == data.remaining()) {
                return arr;
            }
        }

        data.flip();
        return Utils.toArray(data);
    }
}
```

# Compatibility, Deprecation, and Migration Plan

- If the old behavior is correct, then the new behavior is also correct.

## Test Plan

`Perform serialization tests with HeapByteBuffer&DirectByteBuffer of suitable and larger capacity:`

```
    @Test
    public void testByteBufferSerializer() {
        final byte[] bytes = "Hello".getBytes(UTF_8);
        final ByteBuffer heapBuffer0 = ByteBuffer.allocate(bytes.length + 1).put(bytes);
        final ByteBuffer heapBuffer1 = ByteBuffer.allocate(bytes.length).put(bytes);
        final ByteBuffer heapBuffer2 = ByteBuffer.wrap(bytes);
        final ByteBuffer directBuffer0 = ByteBuffer.allocateDirect(bytes.length + 1).put(bytes);
        final ByteBuffer directBuffer1 = ByteBuffer.allocateDirect(bytes.length).put(bytes);
        try (final ByteBufferSerializer serializer = new ByteBufferSerializer()) {
            assertArrayEquals(bytes, serializer.serialize(topic, heapBuffer0));
            assertArrayEquals(bytes, serializer.serialize(topic, heapBuffer1));
            assertArrayEquals(bytes, serializer.serialize(topic, heapBuffer2));
            assertArrayEquals(bytes, serializer.serialize(topic, directBuffer0));
            assertArrayEquals(bytes, serializer.serialize(topic, directBuffer1));
        }
    }
```