KIP-452: Tool to view cluster status

Status

Current state: Under Discussion

Discussion thread: here

JIRA: KAFKA-6393

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

Motivation

The purpose of this KIP is to introduce new command-line tool which enables users to view current status of all active Kafka brokers in the cluster. Currently, without directly accessing ZooKeeper or JMX metrics from each broker, there is no easy way to know basic operational facts. Proposed utility shall provide information about:

1. List of active brokers
2. Where brokers are deployed (rack)
3. Which broker acts as controller
4. Basic listeners configuration
5. Release version of Apache Kafka
6. Configured version of inter-broker protocol and message format
7. Total number of topics and partitions in the cluster
8. For how many partitions given broker acts as leader or follower

Information about Kafka version, inter-broker protocol and message format, may be useful during cluster upgrade when operator needs to restart each broker multiple times.

Statistics displaying number of assigned partitions per broker help to judge whether partition re-assignment shall be executed.

Public Interfaces

We are considering to introduce new `kafka.admin.ClusterStatusCommand`. Supported parameters:

- *bootstrap-server* - servers to use for bootstrapping
- *command-config* - property file containing configuration to be passed to `org.apache.kafka.clients.admin.AdminClient`
- *include-topics-statistics* - option to include topic-level statistics. Command will output total number of topics, partitions and replicas in the cluster, as well as count for how many partitions given broker acts as leader or follower.

Examples

Section presents examples of new tool in-action.

kafka-cluster-status

```
$ kafka-cluster-status.sh --bootstrap-server broker1:9092
Cluster ID: 325U-eVNQVmV40rK0zP9QQ, Active Brokers: 3
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+
| Broker ID | Address      | Rack | Controller | Release | Inter-broker Protocol | Log Format |
Listeners         |
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+
| 0         | broker1:9092 | r1   | *          | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9092 |
| 1         | broker2:9093 | r2   |            | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9093 |
| 2         | broker3:9094 | r3   |            | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9094 |
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+

$ kafka-cluster-status.sh --bootstrap-server broker1:9092 --include-topics-statistics
Cluster ID: 325U-eVNQVmV40rK0zP9QQ, Active Brokers: 3, Topics: 3, Partitions: 18, Replicas: 41
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+----------+--------+
| Broker ID | Address      | Rack | Controller | Release | Inter-broker Protocol | Log Format |
Listeners         | Replicas | Leader |
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+----------+--------+
| 0         | broker1:9092 | r1   | *          | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9092 | 13       | 6      |
| 1         | broker2:9093 | r2   |            | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9093 | 14       | 6      |
| 2         | broker3:9094 | r3   |            | 2.2.0   | 2.2-IV1               | 2.2-IV1    | PLAINTEXT://:
9094 | 14       | 6      |
+-----------+-------------+------+------------+---------+----------------------+------------
+-------------------+----------+--------+
```

Proposed Changes

Kafka Protocol Changes

Add new field to `ApiVersionResponse` representing release version of Apache Kafka. Associated JIRA ticket suggests that version could be deducted from supported API versions, but there might be a release where internal APIs do not change, and still operators would like to verify that they copied new binaries to every broker (e.g. upgrade from 2.1.0 to 2.1.1).

ApiVersion V3

```
ApiVersionRequest => ApiKeys
  // empty

ApiVersionResponse => ErrorCode [ApiVersions] Release   <= ADDED Release
  ErrorCode = INT16
  ApiVersions = ApiKey MinVersion MaxVersion
    ApiKey = INT16
    MinVersion = INT16
    MaxVersion = INT16
  Release = STRING                                      <= ADDED
```

Public API Changes

Support topic metadata in `KafkaAdminClient#describeCluster(...)`. Preserve backward-compatibility and by default do not request topic details (empty list initialized).

```
public class DescribeClusterOptions extends AbstractOptions<DescribeClusterOptions> {
    ...
    private List<MetadataRequestData.MetadataRequestTopic> topicsRequest = new ArrayList<>();       /* Added */
    ...

    public DescribeClusterOptions withTopicNames(String ... topics) {                               /* Added */
        for (String topic : topics) {
            topicsRequest.add(new MetadataRequestData.MetadataRequestTopic().setName(topic));
        }
        return this;
    }

    public List<MetadataRequestData.MetadataRequestTopic> getTopicsRequest() {                      /* Added */
        return topicsRequest;
    }
}

public class DescribeClusterResult {
    ...
    private final KafkaFuture<Set<TopicDescription>> topics;                    /* Added */
    ...

    DescribeClusterResult(KafkaFuture<Collection<Node>> nodes,
                          KafkaFuture<Node> controller,
                          KafkaFuture<String> clusterId,
                          KafkaFuture<Set<AclOperation>> authorizedOperations,
                          KafkaFuture<Set<TopicDescription>> topics) {          /* Added */
        this.nodes = nodes;
        this.controller = controller;
        this.clusterId = clusterId;
        this.authorizedOperations = authorizedOperations;
        this.topics = topics;
    }

    public KafkaFuture<Set<TopicDescription>> getTopics() {                     /* Added */
        return topics;
    }
}
```

Introduce new method `KafkaAdminClient#listBrokerApiVersions(...)` which allows to query API versions from new `org.apache.kafka.clients.admin.AdminClient` interface. Currently, `kafka.admin.BrokerApiVersionsCommand` uses deprecated `kafka.admin.AdminClient`.

```
public abstract class AdminClient implements AutoCloseable {
        ...

        /**
         * List versions of internal API supported by given brokers.
         *
         * @param nodes     Queried broker IDs.
         * @param options   The options to use during remote call.
         * @return          The ApiVersionsResult.
         */
        public abstract ApiVersionsResult listBrokerApiVersions(Collection<Integer> nodes, ApiVersionsOptions
options);
}

public class ApiVersionsOptions extends AbstractOptions<ApiVersionsOptions> {
}

public class ApiVersionsResult {
    final Map<Integer, KafkaFutureImpl<String>> releases;
    final Map<Integer, KafkaFutureImpl<Collection<ApiVersionsResponse.ApiVersion>>> apiVersions;

    ...

    /**
     * Return a future which yields a string representing Apache Kafka release version for each broker.
     */
    public KafkaFuture<Map<Integer, String>> getReleases() {
        ...
    }

    /**
     * Return a future which yields API versions supported by each broker.
     */
    public KafkaFuture<Map<Integer, Collection<ApiVersionsResponse.ApiVersion>>> getApiVersions() {
        ...
    }
}
```

> *Note: I am not particularly satisfied with design of `ApiVersionsResult`, but wanted to limit number of added POJOs. Improvements welcome!*

Compatibility, Deprecation, and Migration Plan

None.

Rejected Alternatives

None.