KIP-536: Propagate broker start time to Admin API

Status

Current state: Under Discussion

Discussion thread: here

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

Motivation

Since the early days of Apache Kafka the way to obtain information about a running cluster was to query zookeeper state. One of the pieces of information available from zookeeper is the start time of each kafka broker, available in the `timestamp` field. This information is useful when building automation that provides functionality such as rolling restarts, to determine whether a broker has successfully restarted.

Public Interfaces

We propose adding an additional field `startTimeMs` to the `Node` class that is returned in the `DescribeClusterResult` return value from `AdminClient.describeCluster()`. This would be a completely backwards compatible change and a logical evolution of the interface requiring no changes to existing code using `AdminClient`. A new method

```
public long startTimeMs();
```

would be introduced to the `Node` class returning the start time, expressed in non-leap milliseconds since the start of the Unix Epoch, of the corresponding broker. If a client with this feature implemented connects to a cluster that doesn't yet have this functionality implemented, the special value `0L` would be returned.

Proposed Changes

The current unix timestamp is currently being written on creation to the `BrokerIdZNode` in zookeeper but is not currently read back by kafka code. We propose to make the following changes to propagate this piece of information.

To complicate things a bit, the broker information that gets returned by the `describeCluster()` API call is read from the metadata cache on the broker responding to the request. To be able to provide the startTimeMs value, this information needs to be propagated from zookeeper to the metadata cache. This means that the `UpdateMetadataBroker` message that is part of the `UpdateMetadataRequest` message needs to be updated to include the timestamp as well as the `Broker` class, so that a version containing startTimeMs is cached on all brokers.

The `MetadataResponseBroker` message part of the `MetadataResponse` message also needs to be updated to hold a timestamp field, as well as the `Node` class that is exposed by the `AdminClient`. An implication of these changes would be that the versions of the affected protocol message pairs affected would be incremented.

Protocol Changes

This is a poposed update to the MetadataResponse message that introduces the startTimeMs field

MetadataResponse version 10

```
// Licensed to the Apache Software Foundation (ASF) under one or more
// contributor license agreements.  See the NOTICE file distributed with
// this work for additional information regarding copyright ownership.
// The ASF licenses this file to You under the Apache License, Version 2.0
// (the "License"); you may not use this file except in compliance with
// the License.  You may obtain a copy of the License at
//
//    http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
```

```
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

{
  "apiKey": 3,
  "type": "response",
  "name": "MetadataResponse",
  // Version 1 adds fields for the rack of each broker, the controller id, and
  // whether or not the topic is internal.
  //
  // Version 2 adds the cluster ID field.
  //
  // Version 3 adds the throttle time.
  //
  // Version 4 is the same as version 3.
  //
  // Version 5 adds a per-partition offline_replicas field. This field specifies
  // the list of replicas that are offline.
  //
  // Starting in version 6, on quota violation, brokers send out responses before throttling.
  //
  // Version 7 adds the leader epoch to the partition metadata.
  //
  // Starting in version 8, brokers can send authorized operations for topic and cluster.
  //
  // Version 9 is the first flexible version.
  //
  // Version 10 introduces per broker startTimeMs field.
  "validVersions": "0-10",
  "flexibleVersions": "9+",
  "fields": [
    { "name": "ThrottleTimeMs", "type": "int32", "versions": "3+",
      "about": "The duration in milliseconds for which the request was throttled due to a quota violation, or
zero if the request did not violate any quota." },
    { "name": "Brokers", "type": "[]MetadataResponseBroker", "versions": "0+",
      "about": "Each broker in the response.", "fields": [
      { "name": "NodeId", "type": "int32", "versions": "0+", "mapKey": true, "entityType": "brokerId",
        "about": "The broker ID." },
      { "name": "Host", "type": "string", "versions": "0+",
        "about": "The broker hostname." },
      { "name": "Port", "type": "int32", "versions": "0+",
        "about": "The broker port." },
      { "name": "Rack", "type": "string", "versions": "1+", "nullableVersions": "1+", "ignorable": true,
"default": "null",
        "about": "The rack of the broker, or null if it has not been assigned to a rack." },
      { "name": "StartTimeMs", "type": "int64", "versions": "10+",
        "about": "The time when this broker was started, expressed as the number of non-leap milliseconds since
the UNIX Epoch" }
    ]},
    { "name": "ClusterId", "type": "string", "nullableVersions": "2+", "versions": "2+", "ignorable": true,
"default": "null",
      "about": "The cluster ID that responding broker belongs to." },
    { "name": "ControllerId", "type": "int32", "versions": "1+", "default": "-1", "ignorable": true,
"entityType": "brokerId",
      "about": "The ID of the controller broker." },
    { "name": "Topics", "type": "[]MetadataResponseTopic", "versions": "0+",
      "about": "Each topic in the response.", "fields": [
      { "name": "ErrorCode", "type": "int16", "versions": "0+",
        "about": "The topic error, or 0 if there was no error." },
      { "name": "Name", "type": "string", "versions": "0+", "mapKey": true, "entityType": "topicName",
        "about": "The topic name." },
      { "name": "IsInternal", "type": "bool", "versions": "1+", "default": "false", "ignorable": true,
        "about": "True if the topic is internal." },
        { "name": "ErrorCode", "type": "int16", "versions": "0+",
          "about": "The partition error, or 0 if there was no error." },
        { "name": "PartitionIndex", "type": "int32", "versions": "0+",
          "about": "The partition index." },
        { "name": "LeaderId", "type": "int32", "versions": "0+", "entityType": "brokerId",
          "about": "The ID of the leader broker." },
        { "name": "LeaderEpoch", "type": "int32", "versions": "7+", "default": "-1", "ignorable": true,
```

```
          "about": "The leader epoch of this partition." },
        { "name": "ReplicaNodes", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
          "about": "The set of all nodes that host this partition." },
        { "name": "IsrNodes", "type": "[]int32", "versions": "0+",
          "about": "The set of nodes that are in sync with the leader for this partition." },
        { "name": "OfflineReplicas", "type": "[]int32", "versions": "5+", "ignorable": true,
          "about": "The set of offline replicas of this partition." }
      ]},
      { "name": "TopicAuthorizedOperations", "type": "int32", "versions": "8+", "default": "-2147483648",
        "about": "32-bit bitfield to represent authorized operations for this topic." }
    ]},
    { "name": "ClusterAuthorizedOperations", "type": "int32", "versions": "8+", "default": "-2147483648",
      "about": "32-bit bitfield to represent authorized operations for this cluster." }
  ]
}
```

The corresponding change to the UpdateMetadataRequest message

UpdateMetadataRequest version 6

```
// Licensed to the Apache Software Foundation (ASF) under one or more
// contributor license agreements.  See the NOTICE file distributed with
// this work for additional information regarding copyright ownership.
// The ASF licenses this file to You under the Apache License, Version 2.0
// (the "License"); you may not use this file except in compliance with
// the License.  You may obtain a copy of the License at
//
//    http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

{
  "apiKey": 6,
  "type": "request",
  "name": "UpdateMetadataRequest",
  // Version 1 allows specifying multiple endpoints for each broker.
  //
  // Version 2 adds the rack.
  //
  // Version 3 adds the listener name.
  //
  // Version 4 adds the offline replica list.
  //
  // Version 5 adds the broker epoch field and normalizes partitions by topic.
  //
  // Version 6 adds the per broker StartTimeMs field.
  "validVersions": "0-7",
  "flexibleVersions": "7+",
  "fields": [
    { "name": "ControllerId", "type": "int32", "versions": "0+", "entityType": "brokerId",
      "about": "The controller id." },
    { "name": "ControllerEpoch", "type": "int32", "versions": "0+",
      "about": "The controller epoch." },
    { "name": "BrokerEpoch", "type": "int64", "versions": "5+", "ignorable": true, "default": "-1",
      "about": "The broker epoch." },
    { "name": "UngroupedPartitionStates", "type": "[]UpdateMetadataPartitionState", "versions": "0-4",
      "about": "In older versions of this RPC, each partition that we would like to update." },
    { "name": "TopicStates", "type": "[]UpdateMetadataTopicState", "versions": "5+",
      "about": "In newer versions of this RPC, each topic that we would like to update.", "fields": [
      { "name": "TopicName", "type": "string", "versions": "5+", "entityType": "topicName",
        "about": "The topic name." },
      { "name": "PartitionStates", "type": "[]UpdateMetadataPartitionState", "versions": "5+",
```

```
        "about": "The partition that we would like to update." }
    ]},
    { "name": "LiveBrokers", "type": "[]UpdateMetadataBroker", "versions": "0+", "fields": [
        { "name": "Id", "type": "int32", "versions": "0+", "entityType": "brokerId",
          "about": "The broker id." },
        // Version 0 of the protocol only allowed specifying a single host and
        // port per broker, rather than an array of endpoints.
        { "name": "V0Host", "type": "string", "versions": "0", "ignorable": true,
          "about": "The broker hostname." },
        { "name": "V0Port", "type": "int32", "versions": "0", "ignorable": true,
          "about": "The broker port." },
        { "name": "Endpoints", "type": "[]UpdateMetadataEndpoint", "versions": "1+", "ignorable": true,
          "about": "The broker endpoints.", "fields": [
          { "name": "Port", "type": "int32", "versions": "1+",
            "about": "The port of this endpoint" },
          { "name": "Host", "type": "string", "versions": "1+",
            "about": "The hostname of this endpoint" },
          { "name": "Listener", "type": "string", "versions": "3+", "ignorable": true,
            "about": "The listener name." },
          { "name": "SecurityProtocol", "type": "int16", "versions": "1+",
            "about": "The security protocol type." }
        ]},
        { "name": "Rack", "type": "string", "versions": "2+", "nullableVersions": "0+", "ignorable": true,
          "about": "The rack which this broker belongs to." },
        { "name": "StartTimeMs", "type": "int64", "versions": "6+", "ignorable": true,
          "about": "The time when this broker was started, expressed as the number of non-leap milliseconds
since the UNIX Epoch"}
    ]}
  ],
  "commonStructs": [
    { "name": "UpdateMetadataPartitionState", "versions": "0+", "fields": [
      { "name": "TopicName", "type": "string", "versions": "0-4", "entityType": "topicName", "ignorable": true,
        "about": "In older versions of this RPC, the topic name." },
      { "name": "PartitionIndex", "type": "int32", "versions": "0+",
        "about": "The partition index." },
      { "name": "ControllerEpoch", "type": "int32", "versions": "0+",
        "about": "The controller epoch." },
      { "name": "Leader", "type": "int32", "versions": "0+", "entityType": "brokerId",
        "about": "The ID of the broker which is the current partition leader." },
      { "name": "LeaderEpoch", "type": "int32", "versions": "0+",
        "about": "The leader epoch of this partition." },
      { "name": "Isr", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
        "about": "The brokers which are in the ISR for this partition." },
      { "name": "ZkVersion", "type": "int32", "versions": "0+",
        "about": "The Zookeeper version." },
      { "name": "Replicas", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
        "about": "All the replicas of this partition." },
      { "name": "OfflineReplicas", "type": "[]int32", "versions": "4+", "entityType": "brokerId", "ignorable":
true,
        "about": "The replicas of this partition which are offline." }
    ]}
  ]
}
```

Compatibility, Deprecation, and Migration Plan

This is a completely backwards compatible extension of the existing API. The only compatibility consideration that needs to be taken into account is that a client with this change included connecting to an older cluster needs to handle this condition according to the public interface description above, with the `startTimeMs()` accessor returning the special value `0L`.

Rejected Alternatives

It is certainly possible to use mechanisms outside of Kafka to determine when a broker was started, using for example the operating system process table. However, such solutions would be very specific to their execution environment and it would take a lot of work to have them perform similarly well as the solution outlined above.