# Apache HBase - Nanosecond Timestamps

Andrey Elenskiy - andrey.elenskiy@arista.com

## Problem

Historically HBase assumed that time is stored in milliseconds for its operations. Things like updates without provided timestamp, column family TTL and "hbase.hstore.time.to.purge.deletes" use System.currentTimeMillis() to find out current time and the options are normalized to milliseconds to carry out the expirations. However, some applications have updates timestamped at the source that happen submillisecond and truncating those to a millisecond results in overrides/loss of data. Fortunately, HBase's timestamps are 8 bytes, so a nanosecond timestamp can fit comfortably, so this use case is almost completely satisfied. Column family TTL's do not work in such a case though.

Additionally, as timestamps are stored in 8 bytes anyway, so we might as well utilize that space for better precision. And this opens a road to Hyper Logical Clocks which require a monotonic clock and that can be provided with System.nanoTime()/Instant API in Java.

## Technical Approach

The eventual goal is transition to nanoseconds in HBase everywhere, but we start small.

A new function "currentTimeNano()" has been added to EnvironmentEdge instead of having "currentTime()" return nanoseconds right away. In the first iteration, we'd like to not have any data migrations for meta tables/procedures and only update compaction/querymatching functionality to use "currentTimeNano()" depending on a table attribute. "currentTimeNano()" uses Java 8's Instant API which returns current system time. We can also modify it later to return monotonic clock time by swapping implementation of Clock (in fact the whole EnvironmentEdgeManager could be refactored to inject Clock instead of having multiple implementations of EnvironmentEdge).

A new table attribute "NANOSECOND_TIMESTAMPS" is added to tell HBase that all timestamps in that table should be handled as nanoseconds. Note that column family attribute is not used because that would complicate matters of dealing with different precisions for making queries for clients. HRegion looks at this table attribute before assigning a timestamp to a mutation (if none is provided). As well as querymatchers look at this attribute to filter out expired values in memstore during scan operations and during compactions.

Most functions and classes that deal with time, already have a reference to ColumnFamilyDescriptor, but not TableDescriptor, so some classes have been updated to pass it on:

- ScanInfo
- PartitionedMobCompactor

It resulted in more extensible code.

# Upgrade Impact

HBase now will require Java 8 as it relies on Instant API to get nanoseconds. If that is not desired we can review alternatives to keep support of Java 7.

The “NANOSECOND_TIMESTAMPS” attribute should be specified either on new tables or on existing tables which have timestamps only with nanosecond precision. Existing tables should be altered with “alter ‘<table>’, ‘NANOSECOND_TIMESTAMPS’ => ‘true’”.

HBase clients that don’t provide timestamps to their requests, don’t need to change anything. The clients looking to write into a table that supports nanoseconds will have to provide nanoseconds via “setTimestamp()” using Java 8’s Instant API. Same goes for per cell TTL and Get/Scan requests with “setTimeRange()”.

There's no migration from milliseconds to nanoseconds for already existing tables. We could add this migration as part of compaction if you think that would be useful, but that would obviously make the change more complex. We’ve also developed a mapreduce job to migrate HFiles from milliseconds to nanoseconds and can make the tool available.

Column family TTL (specified in seconds) and "hbase.hstore.time.to.purge.deletes" (specified in milliseconds) options don't need to be changed, those are adjusted automatically based on the table attribute.

# Test Strategy

Most unit tests can be reused and rerun with new table attribute and adjusting the responses. I’m going to add more specific unit tests for following logic:

- Compactions
- Memstore scans
- Mutations without timestamps
- hbase shell

# Documentation

Versions subsection will need to be updated to mention the new nanosecond option:
https://hbase.apache.org/book.html#versions