Proposal of Storing YARN Metrics into the Timeline Store

Motivation

We previously store the generic application history into HDFS. Later on, we came up with a more generalized data model, made a so-called timeline store, and used it store the application specific metrics. Now we propose to store the generic application history here as well. There're the following benefits:

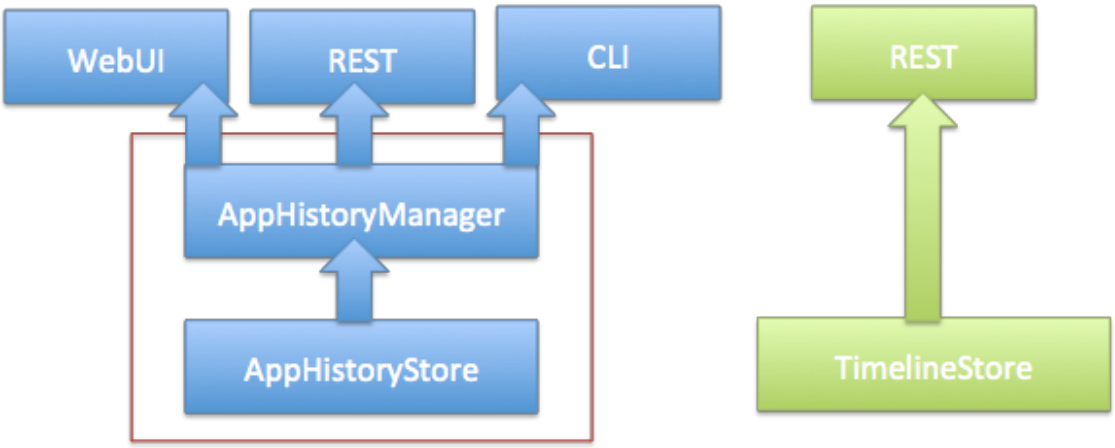
- We just need to maintain a single type of store instead of two, and ApplciationHistoryStore and TimelineStore are two completely different interface sets;
- HDFS application history store has limitations:
    - No data retention mechanism,
    - Not efficient for search if there are history files of a large number of application,
    - No caching support, such that all queries go to HDFS;
- It is going to be easy for us to link the generic data with the per-framework metrics.

Approach

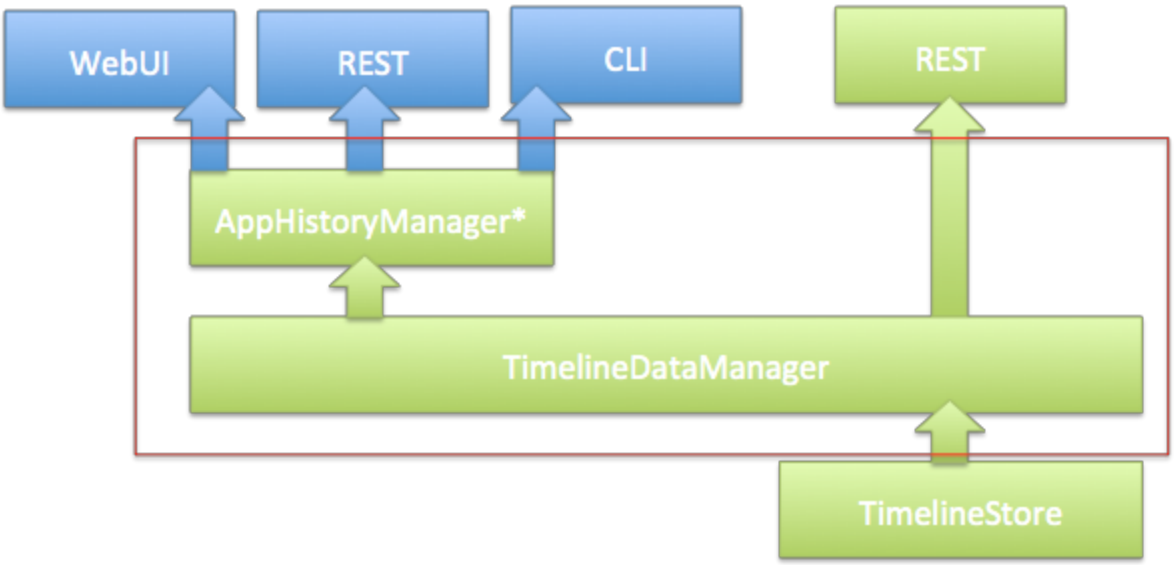
We should discard ApplicationHistoryStore. The reason why ApplicationHistoryStore is no longer suitable:

- Reader and writer are heterogeneous
    - RM (maybe NM as well) uses TimelineClient to post entities
    - Entities will be pulled from REST APIs (not really send the REST request, but reuse the logic)
- It's more flexible to define and extend events that happen to app/appattempt/container. We can further split the whole app/appattempt/container metrics into a number of small events and post them to the timeline server.

The current structure in the timeline server:



The structure after the proposed change:

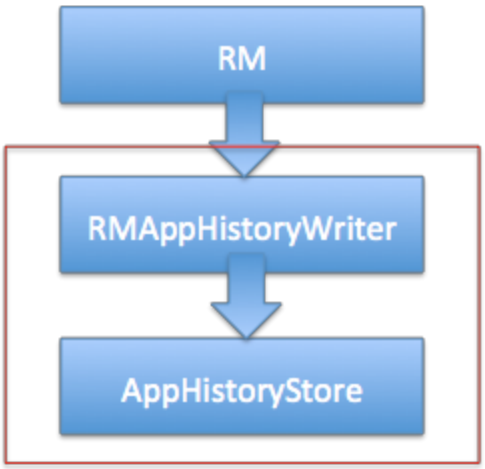


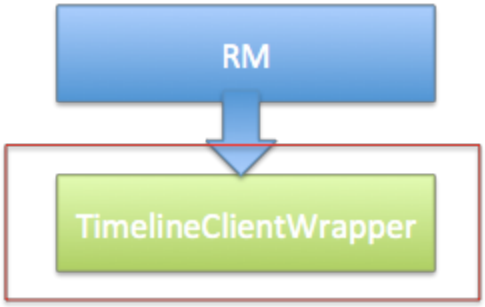
We need to do:

- Extract the logic in TimelineWebServices and put it into TimelineDataManager, such that it can be either invoked by the REST interface (TimelineWebServices) and the generic history service (AppHistoryManager) internally.
- Make an AppHistoryManager implementation v2. Instead of using AppHistoryStore interface, it extracts information from TimelineDataManager and translate it into Application(Attempt)/ContainerHistoryData.

From the view of components on top of AppHistoryManager, the store change is transparent. The existing interfaces should just work.

On RM side, the change will be as follows, from left diagram to right one. As we did in RMAppHistoryWriter, we can invoke TimelineClient in (a) separate thread(s) to post entities.





Last but not least, we need to get rid of the app history store configuration in yarn-default.xml, given users don't set them explicitly, the generic history service uses timeline store by default.

Miscs

Previously the generic history service doesn't work in a secure mode. However, the timeline server is able to run in secure mode now. Therefore, to access of the timeline store in the secure mode, the generic history service needs to work with Kerberos authentication.

Another part of security is ACLs for the generic history service. Ideally, we'd like to enforce the same access check for getting the application history information from the timeline server as we did for getting the running application information from RM. This may go beyond the scope of rebasing generic history information to the timeline store. We can deal with it separately.