Adding PAUSE and RESUME states for YARN containers

Arun Suresh, Roni Burd, Konstantinos Karanasos, Hitesh Sharma

In this document we discuss the design for pausing and resuming opportunistic containers (OC). When an opportunistic container is preempted then the default behavior is to kill the container. This may not be desirable from work preservation perspective and we are looking add an option to PAUSE the container. When resources free up on the node then we RESUME the container. Whether pause and resume are supported or not is specific to the container implementation and if they don't support these operations then pause would result in killing of the container.

Here are the key points for this design:

- Only an OC can be paused and resumed. Pause and resume are not supported for Guaranteed containers.
- Pause and resume events would be raised from the container manager and handled in the container launcher, which will pass them to the container executor, where they will be handled.
- Only the container manager can pause and resume a container. Currently we don't see any generic reason for the Application Master to pause/resume a container and thus the AM-NM protocol is unchanged.
- How pause and resume is done is specific to the executor. For instance, if the container is a virtual machine, then preempt would pause the VM and resume would restore it back to the running state. If the container doesn't support pause/resume then it would default to killing the container.
- The default implementation of pause and resume in the ContainerExecutor class would be to throw an exception and kill the container.
- When the OC is paused then from resource accounting perspective the resources used by it are taken back. And when the OC container is resumed then the container manager will account for the resources used by it.
- Paused (opportunistic) containers are reported to the RM, and they are treated the same as running opportunistic containers, in the sense that their resources are not considered as allocated.
- We introduce the following states and transitions for pause and resume events.
    - New states for ContainerState enum:
        - PAUSED, PAUSING, and RESUMING
    - New container events:
        - PAUSE_CONTAINER and RESUME_CONTAINER can be raised from container manager
- When a container is paused/resumed then CONTAINER_PAUSED and CONTAINER_RESUMED events would be raised from the container launcher.

- The application master can query whether the container is in paused state via NMClient.getContainerStatus API, but as mentioned earlier the AM can’t pause/resume container on its own.
- When resources free up on the node then queued guaranteed containers are started first, followed by paused opportunistic, and then the queued opportunistic ones.