System-Services

Goal

As YARN evolved to support diverse workloads such as services (YARN-5079), there are certain services which mandatorily need to be started during RM startup or to be dynamically added as System-Services. These are similar to init.d services in linux. One of the example for System-Services is starting flow collector service for ATSv2.
YARN-1593 gives detailed discussion on system containers and brief about system services. This document majorly covers only on system service design.

Characteristics

- Long running services, never be exit for system failures.
- Started during YARN startup or added dynamically by admin via CLI.
- Lifetime of services is same as cluster lifetime by default.
- Run as the user specified by the admin
- Restarted in case of machine crashes.
- Actions such as start/stop/restart/kill can be performed by Admin only. No other user can execute these actions even run-as user.
- No two versions of same service are allowed to run.

Proposal

Where to locate System-Services specifications

YARN-5079 allows to start new service via REST endpoint with service specifications. For System-Services, the same service specifications are configured by admin and stores as json formated file in configured services folder on a FileSystem (HDFS).

- During RM startup, scans this directory location to get all the configured services.
- Each file is treated as one service, validated and started these services.
- Each resource-specification can be configured with placement policies, affinity/anti-affinity, resources and others. Dependency can also be specified so that it is resolved during service start.

When System-Services can be added

System services could be added either during cluster startup or dynamically by admin only.

Cluster startup

Admin need to place all the System-Service specifications in configured service directory. During YARN cluster startup, RM scans this directory and starts configured services. In subsequent RM restart/HA, applications will be recovered. However, scanning process will happen, and to be skipped if service is already running.

Dynamically adding services

Admin can add services dynamically. Admin need to place resource specifications file into configured service directory and execute *rmadmin* command to start a service along with file name.

Admin CLI Actions

Admin CLI should able to support service actions.

- *Register/deregister of system-service* are done via rmadmin CLI. This way, admin will be able to start new system-services for running cluster by adding service-specification file into directory and execute CLI *rmadmin* command.

  ./yarn rmadmin service create fileName
  ./yarn rmadmin service delete serviceName

Native service tool

There is necessity of tool is CLI tool that interact with native services. This is a tool that interacts with native service web service to post actions or for flexing the service. Admin user can make use of this tool for actions such as start/stop/update of services. This tool should also be able to list running service details so that service discovery of system services should be smooth.

Capacity allocation for System-Services

System services are critical in cluster. These services have to be started with highest priority. So these should get highest priority in allocation. Admin needs to configure and allocate a capacity. This queue is SYSTEM queue with highest queue priority.

Capacity requirements

SYSTEM queue characteristics and behavior is same as user queue, only difference is with queue-priority. SYSTEM queue is configured by Admin with capacity specified.

Limits of the SYSTEM queue

- What if admin forgets to configure SYSTEM queue? All the system service startup will fail.
- Since capacity is configured to be same as user-queues, number of system services that can launched will be restricted by SYSTEM queue capacity. If capacity is full, then services wait for allocation. In such cases, admin has to identity those services and act upon those waiting services.
- System QUEUE undergo all the scheduler features such as priority, AMMaxlimit, preemption.

Security

System services rely on the native service security system only. In secured cluster, authenticated Kerberos user should be able to connect to these services.

Which user owns System-Services

We will have separately identifies users for all the System-Service to run. These users configurable by admin through Queue ACLs on the SYSTEM queue. By default, this queue is restricted to something like “yarn-system-apps” user.
Note: System service can not be run as user “yarn”. LCE doesn’t let apps run by “yarn” user.

ResourceManager

Native services is going to be merged with RM. So, all the system service management will come under ResourceManager startup.
On ResourceManager side, some changes are needed.

- Required to add new service YarnServiceManager started during RM startup. This service is responsible for scanning services directory and starting System-Services.
- During RM restart or work-preserving-restart, RM should be able to identify running System-Services. And also identify new services added and launches newly added service.
- At any point service errors, RM will be keep running.

Inter System-Service dependencies

All System-Services that depend on each other should become one single complex application (app of apps in YARN-5079 API)

Service Discovery of system services

System services are registered with yarn-registry with service name. Let's say, ATSv2FlowCollector is service name. Given service name, Yarn-registry client get details about collector service details. By this way yarn-clients get address of services which can be used later.