YARN-3409 : Support Node Attribute functionality

Authors : Naganarasimha G R , Sunil G, Wangda Tan

With inputs from : Weiwei Yang, Bibin A Chundatt, Vinod Kumar Vavilappali, Arun Suresh, Konstantinos Karanasos, Chong chen, Lei Guo,.

What are Node Attribute Labels ?

Like Node Partition labels introduced in aYARN-2492, Node Attributes labels are non-tangible resource associated with the node.

How is it different from existing Node Labels ?

Existing Node Labels which are termed as Partitions are used for logical partitioning the cluster, we can more like treat it as “Resource Pools” in traditional scheduling terms. As its partitioning the cluster based on these labels, a node can belong to only ONE partition label.

But Node Attributes are just used to describe the attributes of a Node without resource guarantees and only aids in applications to pick up the right nodes based on expression of multitude of these attributes.

Use cases :

Hardware Constraints :

Not all resources can be isolated but we still need to support scheduling of tasks on the nodes based on the resources like GPU, FPGA, SSD, (dual) network cards, # of disks, InfiniBand etc. Further Tasks can depend on multiple of these resources hence cannot be supported by existing “Partition Labels” concept.

Task Constraints :

In many scenarios, applications requires its containers to be run on nodes having specific Operating system versions, processor architecture, software library versions etc..

In many cases applications would require combination of multiple such attributes of the nodes. We will not be able to effectively achieve this only with Partitions, hence we require Node Attribute Labels.

Node Attribute Types and expression support.

We plan to support typed Attribute Labels, Type is useful to determine what operations in the expressions can be specified and thus avoids if unwanted scheduler computations on invalid operations and also ensures validating the user input.

Node attribute will be defined as *[prefix/]key[:type][=value]*, type will be included by both of resource request for scheduling and node attributes mapping, so we can use correct method to compare requested and available attributes.

We plan to support initially with String Attribute Type which will be default type if not specified. And supported operations as IN and NOT IN. In the future we plan to support GT/LT, etc. for other types like Integer/Version, etc.

Prefix will help in seggregating the attributes provided by different sources and is a DNS name, like *nm.yarn.io/docker-supported,nm.yarn.io/container-executor-type, *.yarn.io* is limited to be defined by system instead of admin using centralized API.

Mapping Node Attributes to a Node

Node to Attribute Mapping can be done in the following ways :

a. **Centralised (Admin)** : Admin wants to specify the attributes for each node either through CLI or REST. (Some prefixes are not allowed for centralized attributes, for example *.yarn.io)

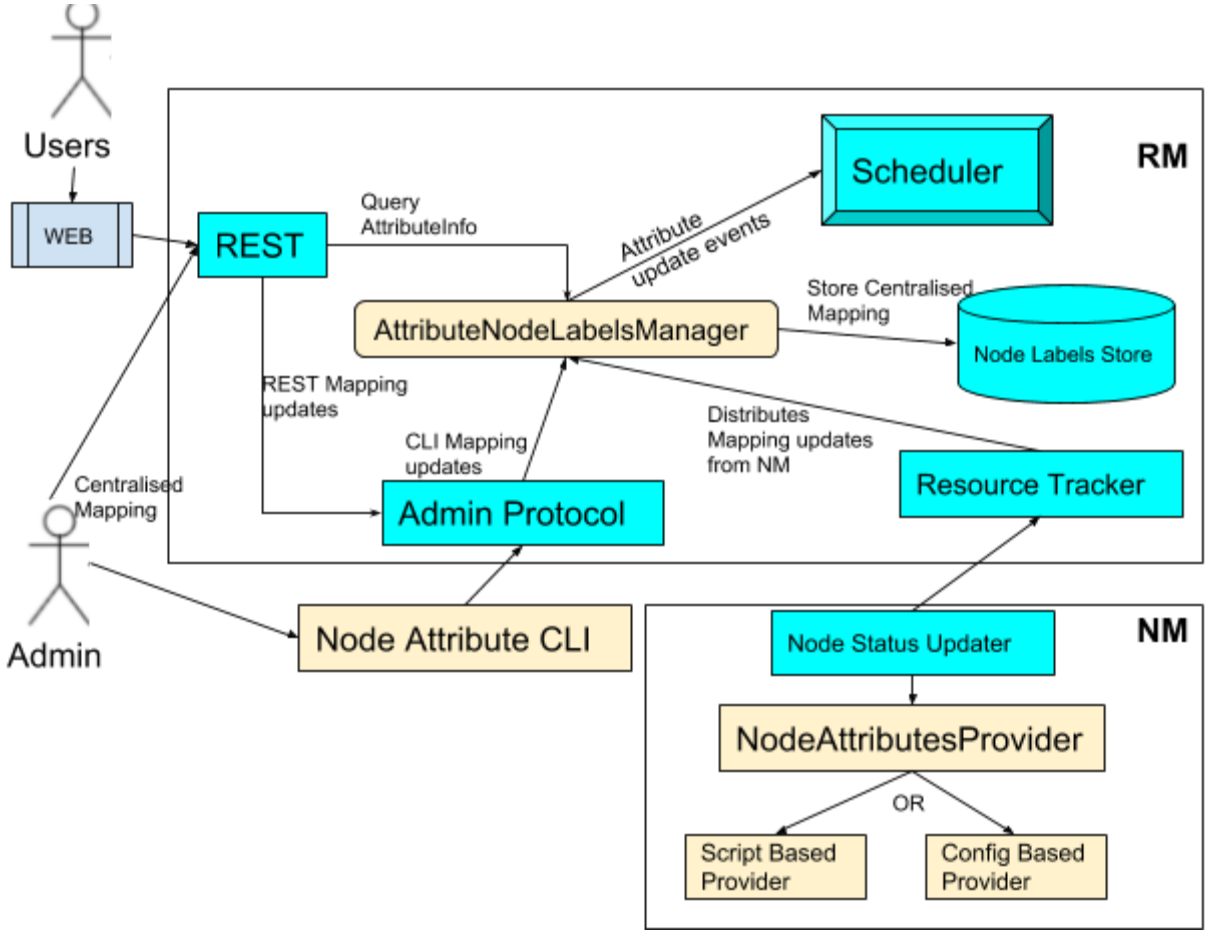
b. **Distributed (Admin)**: Admin wants to automate the Node to attribute Mapping through scripts or configure for group of nodes with specific attributes. Prefix would be "*distributed.yarn.io"* .

c. **System**: System daemons (scheduler/nodemanager, etc.) can attach labels to nodes. Examples can be hostname / rackname, failure domain, spot instance. Prefix would be "**.yarn.io"*

d.

e. User Based Attribute Mapping : <Need to be discussed further>

Architecture



- Attribute Node Attributes Manager in RM manages all node attributes.
- Centralised Node Attributes Mapping can be done through CLI and REST which will be updated to Attribute Node Labels Manager via AdminProtocol.
- Attribute information can be queried through CLI and WEBUI and information will be retrieved from Attribute Node Attributes Manager.
- Attribute Node Labels Manager persists attribute mapping information for the centralised labels only.
- Scheduler will be informed on modification of attributes on Nodes based on which they can adjust their scheduling decisions.

For Distributed Node Attributes Mapping:

- Node status updater queries for NodeAttributesProvider for Node attributes and sends it across to RM as part of heartbeat response.
- NodeAttributesProvider is configured to fetch the attributes either from script or from configuration. And if there is any change in attribute mapping it will reply back to provider.
- ResourceTracker detects if any attributes are sent by NM as part of Heartbeat response and if so updates the Attributes NodeLabels Manager.
- Scheduler will be informed on modification of attributes on Nodes based on which they can adjust their scheduling decisions.

Note :

1. We had come across a case from Intel where in **Delegated-Centralized** option was provided at RM to map Nodes to Partitions based on external interface and periodically we were refreshing the mapping through this interface. But IIUC its not required for attributes.
2. Validation failures for Distributed Node Attributes Mapping needs to be discussed in detail in the jira. <options: remove only invalid mapping and let the valid mappings be there , Remove all attribute mappings or Keep the previous mappings >.

CLI syntaxes:

Replace command : This will help in replacing the existing mapping with the new mapping as specified in the command.
yarn node-attributes -replace
<node1:[prefix/]attribute[(type)][=value],[prefix/]attribute1[=value],attribute2
node2:attribute2[=value],attribute3>,

Add command : This will help in adding or updating one or more mapping(s) without impacting other existing mapping for a node.
yarn node-attributes -add
<node1:[prefix/]attribute[(type)][=value],[prefix/]attribute1[=value],attribute2
<node2:attribute2[=value],attribute3>

Remove command : This will help in removing one or more mapping(s) without impacting other existing mapping for node.

yarn node-attributes -remove <node1:attribute,attribute1 node2:attribute2>