YARN-796: Node-labels - Requirements & Design doc - V2

Authors: Wangda Tan, Vinod Kumar Vavilapalli, Ram Venkatesh with inputs from Arun C Murthy, Bikas Saha, Junping Du

Last Modified Date: Aug 15, 2014

What is a node-label?

- A non-tangible-resource associated with the node.

Use cases

- **Hardware Constraints:** Many applications need specific hardware such as GPU, FPGA etc to run against. Many a times, only a subset of a cluster will have this hardware installed due to several reasons like cost and non-pervasive usage. We can use node-labels to tag machines with such specialized hardware and let users run applications against them.
- **Task Constraints:** YARN currently doesn't have the ability to segregate nodes in a cluster based on features like the Operating System, processor architecture etc. So, applications requiring to run only on specific nodes in the cluster cannot do so inside YARN. Node-labels can help applications flock to nodes of their choice.
- **Partitioning and exclusive use:** Admins may want to logically partition (not physically) a large cluster to several smaller units, each partitioned unit can be used by one or more

specialized workloads, like node[1-5] for running HBase, node[6-10] for Storm, etc. This is a use-case that has so far been antithetical to the philosophy of YARN to elastically share cluster resources (and not hard-partition them) with as high utilization and throughput as possible, but in sites where it is the only acceptable solution, node-labels can be used to achieve this.

Requirements

- One node can have multiple admin-specified labels (os, arch, dept etc.).
    - We should have limits on the number of labels per node so as to ease the management of system's book-keeping requirements
    - We should also optionally define a list of cluster-level acceptable labels to avoid configuration mistakes.
        - This list should be dynamically changeable just like node-to-label mappings
- There can be nodes that have no labels associated with them.
- One label can be associated with multiple nodes.
- There can be labels that have no nodes associated with them.
- Labels should be persistent across cluster component restarts (ResourceManager, NodeManager etc.) and cluster upgrades.
- Applications should be able to request for containers on nodes by specifying a list of labels.
- Tangible-resources (like bandwidth/memory/disk) shouldn't be a part of node-labels and should be explicitly modeled as a YARN resource.
- **Admin tools**:
    - Admins should be able to associate/un-associate labels with nodes dynamically.
    - Admins should be able to list labels associated with specified nodes and nodes associated with specified labels.
    - Admins should be able to remove a label entirely from a cluster.
        - Once removed, that label will be no longer associated with any nodes.
    - Security and access controls for managing Labels
        - Admins should be able to set a list of labels of a queue can access
        - Only admins will be allowed to create new labels and add/remove the labels associated with a node/queue.
    - Admins should be able to do all of the above ***dynamically*** at run-time without component restart.
- **Management**:
    - Admins shouldn't logically see the underlying storage formats of node-label configurations
    - Label definitions and configuration management should be scheduler-agnostic

Requirements from admin's point of view

- Admin can list/associate/un-associate labels for each node dynamically.
    - Need APIs such as,
        - RM Admin CLI
        - RM Admin REST APIs
- Each queue will have a list of labels that determines the labels that the queue can access:
    - Example:
        - <Queue-A>
        - <Capacity>100</Capacity>
        - <Labels>red,blue,green</labels>
        - </Queue-A>

Requirements on Node-labels configuration

- Decentralized configuration
    - Support per-node configuration (set in yarn-site.xml) instead of just the centralized configuration (as below) to support **legacy** decentralized way of configuration
        - Changing labels is very complex in the legacy way. Admins may be forced to manually change configs and restart nodes.
- Centralized configuration
    - Admins and end-users shouldn't ideally even know of the configuration files listing node-labels. The only endpoints should be CLI or REST.
        - RM can store it in whatever format it wants and wherever it wants
    - CLI/REST APIs
        - Set labels on a node(s)
            - Even take as input a specific format of a file
        - Add/remove labels on node(s)
        - Get labels on a node(s)
        - RM should have a list of node-labels at boot-up time. Two options
            - The CLI utility to set/modify labels has a mode which doesn't need RM to be running
            - We can pass it to the RM via yarn-site.xml when starting, similar to the present day config files
        - For a given label
            - Get the list of nodes marked with that label
- Admins have a choice of either doing a decentralized configuration or the centralized configuration - one cannot mix and match both.
- Restrictions
    - Label for each node should start with an alpha-numeric character. Character case will be ignored when comparing two labels. Letter, number, "-", "_" are only valid characters in a label name.

        - This is to avoid the pain of rendering by UIs (potentially also security holes and inconvenience on the web-UIs)
        - Should we limit max length of node labels?

Requirements from a user's point of view

- **User can specify dependency on node-labels via ResourceRequests**
    - *At application level*: During application-submission, user should be able to specify a fixed list of labels to apply for all containers of that application.
        - ResourceManager uses the app-level label requirements as requirement for all containers including the ApplicationMaster
    - *At container level*: Every time some requests are made by the AM, user should be able to set node-label requirements at some ResourceRequest levels as described below.
    - Force admins to use node-labels at queue-level (see end of this design doc).

- **Strength of the label-requirement**
    - Labels can logically be hard or soft constraints on the task.
    - To simplify, to start with, we will only support **hard** requirement on node-labels. That is -- when a resource request asked for a node with label "foo", it will only be allocated to nodes with label "foo". [1], [2]
    - When a node has a label specified on it, only tasks that specify the label will be allowed to execute on that node.

- **Request validity**
    - ResourceRequest with a non-existent label (point in time) is illegal: results in InvalidResourceRequestException
    - ResourceRequest with a label but queue belongs to do not have permission to access the label results in IllegalResourceRequestException

- **Number of labels per request**: Multiple node-labels can be specified as requirements at same time. User can specify an expression like "red || (blue && green)". In short term, we will support "&&" only.

Topics for discussion

Config labels that a queue can access

- Configuration may looks like (in existing scheduler configuration like capacity-scheduler.xml

<Queue-A>
<Capacity>100</Capacity>
<Labels>red,blue,green</labels>
</Queue-A>

- By default, if admin doesn't set the "labels" field, it will be "", which means a queue can only use node without labels
- (TBD) If admin set label to <Labels>*</Labels> means a queue can use all labels

Respect node labels when preemption resources

Preemption logic in existing schedulers need to be aware of node-labels

Respect node-labels when calculate headroom/user-limit

Calculation of per-application headroom needs to take node-labels into account.

Hierarchy of queue configuration

- If there's a list of labels set in a leaf queue's configuration, we will use the list directly
- If not set, we will inherit parent queue's label configuration

Forward compatibility

- In case all the nodes in a cluster are assigned one or more labels, existing applications that didn't depend on labels may be starved because they don't set any label in ResourceRequest.
    - Maybe a configuration option to user: If no label set in ResourceRequest, should we automatically add a default label for such user/group/queue?

Support locality within a set of nodes selected via labels,

Applications may want to specify node-locality within the context of node-labels.

Queue cannot use guaranteed capacity

Assuming following case,

- a cluster has 3 labels, red/green/blue, one node has one labe;
- 10 nodes has red, 3 has green and 3 has blue
- A queue with label set is labels = "green, blue", and it's guaranteed resource = 80%. So its guaranteed resource cannot be satisfied.

We should print warnings to web UI or logs to let admin/user know that.

Sequence to check if a nodeX can be used by a resource request

1. Assume a ResourceRequest from qA asks for label="red && blue", check if red and blue can be accessed by qA
2. If yes, check if nodeX contains all labels included by the RR
3. If yes, check if it has enough resource
4. If yes, allocate resource.

References

[1] Choosy: Max-Min Fair Sharing for Datacenter Jobs with Constraints, Ali Ghodsi et al.
[2] B. Sharma, V. Chudnovsky, J. L. Hellerstein, R. Rifaat, and C. R. Das. Modeling and Synthesizing Task Placement Constraints in Google Compute Clusters. In ACM SoCC, 2011