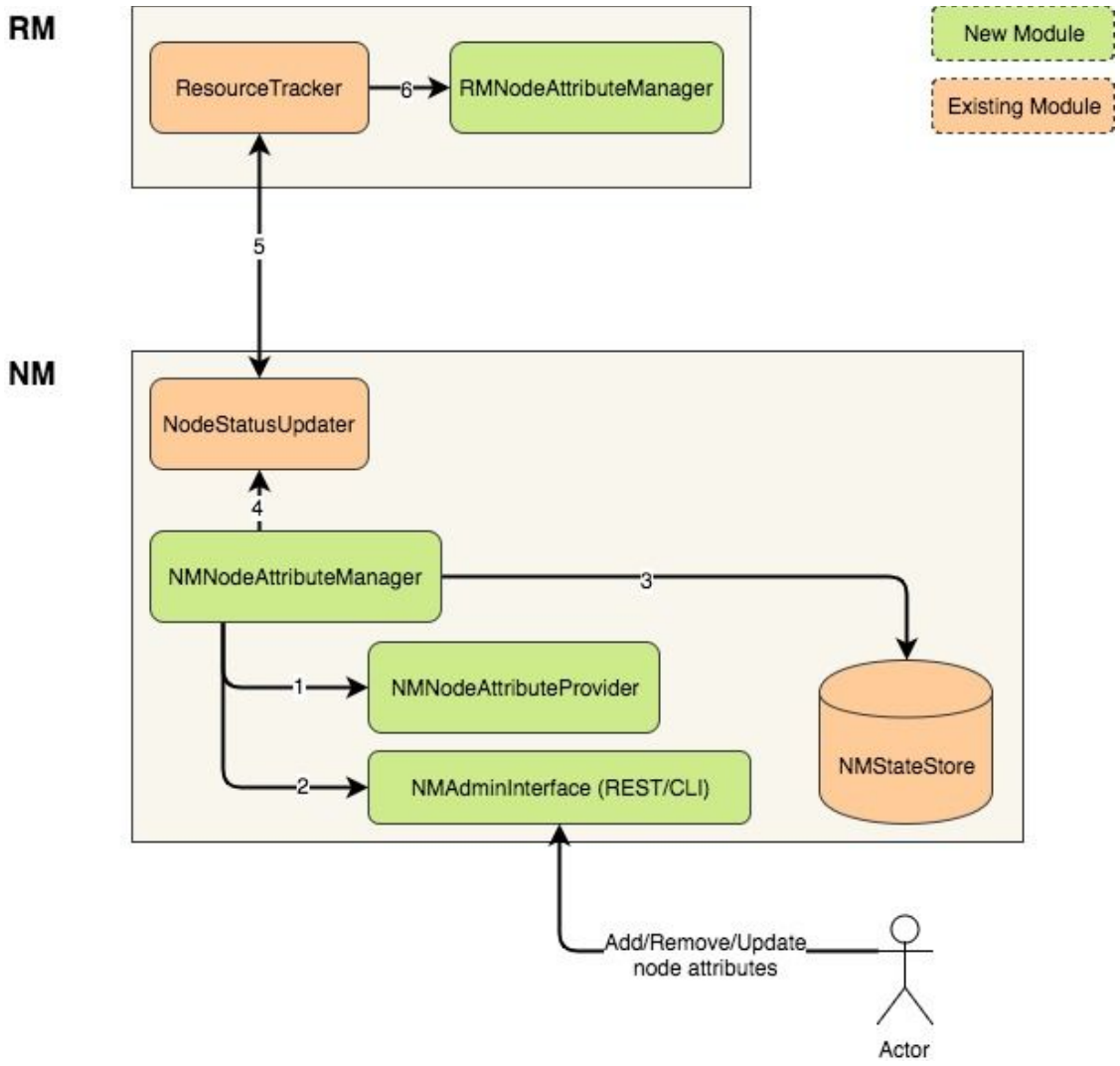
Distributed Node Attributes

Manage Node Attributes

Node labels support 3 sort of mappings, i.e centralized, distributed and delegated-centralized. In the node-attribute scenario, we prefer to use distributed approach. Because

1. In common use cases, NM has different attributes, e.g hostname, library version etc. Let NM report their attributes dynamically via HB simplifies the admin process.
2. We can persistent node-attributes info in NM state store, this way we don't need RM to persistent these info. If RM reboots, it just needs to wait until NM reports their attributes.

Components and Major Workflow



1. NMNodeAttributeManager maintains node-attributes of current node in memory, it inits with a NMNodeAttributeProvider, which mostly runs a script to parse node attributes. Each NM HB, it simply fetches attributes from the manager, it doesn't necessarily run script every time, depending on the implementation.
2. Node attribute can be modified by NM admin interface, via REST or CLI.
3. NMNodeAttributeManager takes care of persisting node-attributes to NMStateStore, in levelDB. This is required because attributes added via #2 will get lost on NM restart.
4. NodeStatusUpdater fetches node attributes info from NMNodeAttributeManager on each HB.
5. NodeStatusUpdater reports node attributes along with other HB info to RM.
6. RMNodeAttributeManager maintains node to attributes mapping in memory, the resource tracker keeps updating this mapping in each HB processing. This info will be used by scheduler when it makes allocation decisions that contains placement constraints.