Deployment Planning Guide

Community 1.5.1 release

The purpose of this document is to educate the user about the different strategies that can be adopted to optimize the usage of Jumbune on Hadoop and also to give an overview of the activities required to get the product integrated into various stages of the development cycle.

Table of Contents

Assumptions:

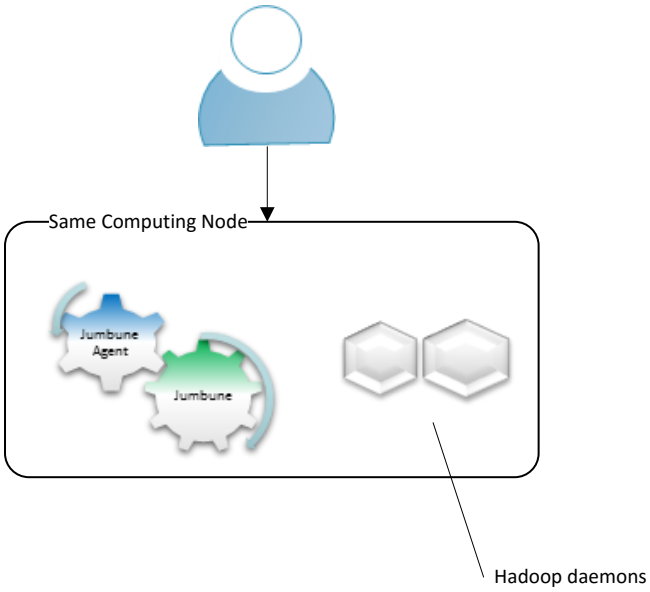
- The machine intended to run Jumbune container node is up & running and has at least 4 GB of available RAM.
- The Hadoop distribution in use should be supported by Jumbune, for compatibility specifications, please refer to the usage document.
- The user has sufficient permissions to create, remove and execute on the respective Jumbune directories i.e. Jumbune Home
- The user running the agent jar on the Name Node has sufficient permissions to the agent directories i.e. Agent Home and has permissions to run job on the Hadoop cluster.

Constraints:

- Data load partition and replica management metrics would only be available if the user running the agent on the Name Node has rights to run *dfsamin* command on the cluster.

Deployment Strategies

Jumbune and pseudo distributed cluster on a single machine.



Jumbune can be deployed on a box with a pseudo distributed Hadoop cluster running. This strategy will be beneficial for developers to profile or debug their MapReduce code.

Flow Debugging module could be used on the developer box to get a detailed data flow analysis for the various MapReduce code levels, which would help the developer fine tune the algorithm and detect faults within the same.

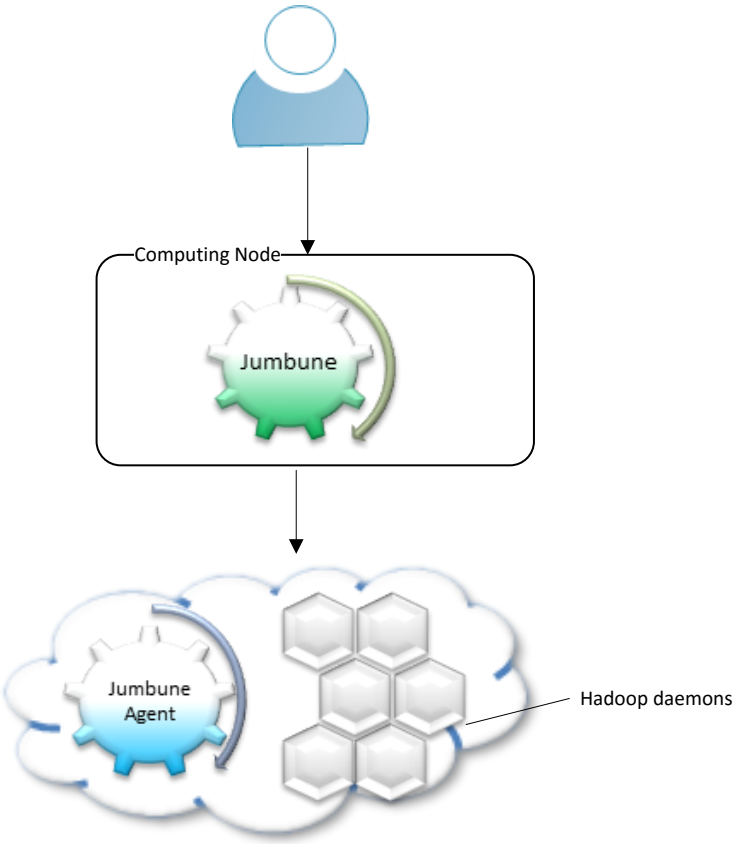
Data Validation module would be useful for the developer to verify and validate, either the data produced by his code or ones already present on the HDFS. Data validator provides an easy to use interface to validate the data, through null checks, data type checks and validations based on regular expressions.

Job Profiler helps the developer to profile the various phases of MapReduce job execution with fine-grained node level analysis and a cluster level resource consumption view, all of which can be used for optimizing the job.

Data Quality Timeline traces the conservation of data quality over a period of time, even in massive data offloading environment.

Data Profiling computes statistic assessment of data values within a data set for consistency, uniqueness and logic.

Jumbune on a distinct node running on top of a Hadoop cluster.



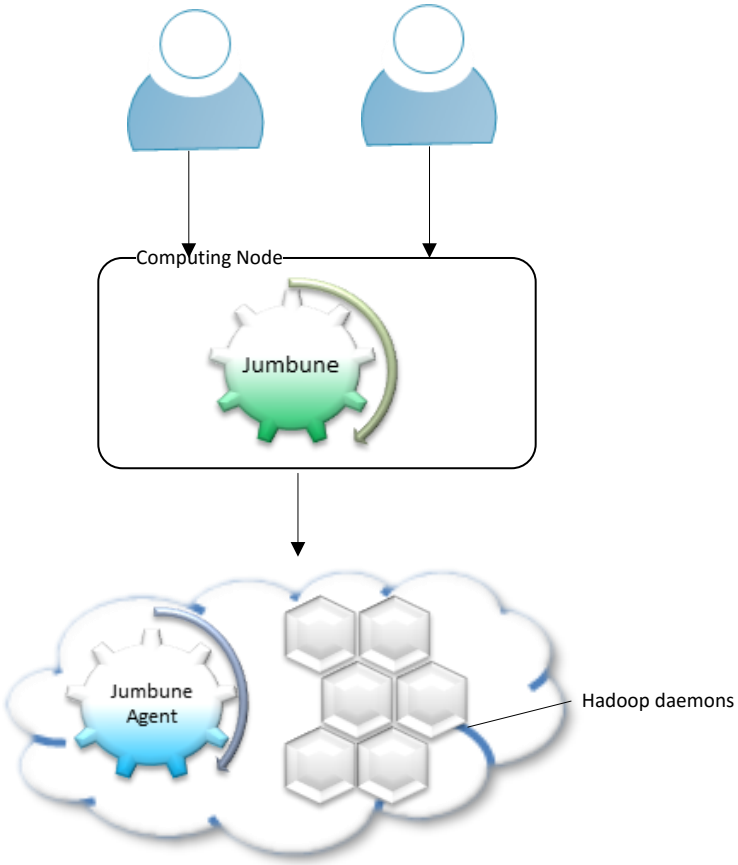
This scenario would be useful in the QA environment, where a quality engineer could connect to an agent running on Hadoop cluster and perform various levels of testing.

One of the main challenges in Big Data testing is the sheer variety and volume of data. Jumbune's **Data Validation** module provides a straightforward, simple and comprehensive way to validate and verify the data present on the HDFS. This would save a lot of man hours that would've gone into writing custom code or manual inspection of the data.

The Hadoop cluster can be monitored on a variety of parameters using the **Cluster profiling** module of Jumbune. It's lightweight and needs to be deployed only on the Namenode, it also can turned on at will and profiling interval can be set to the users requirements.

This ensures that the monitoring is easy to perform and isn't a memory hog. Replica management and data load partition modules can be used to ensure that the data is distributed evenly across the nodes.

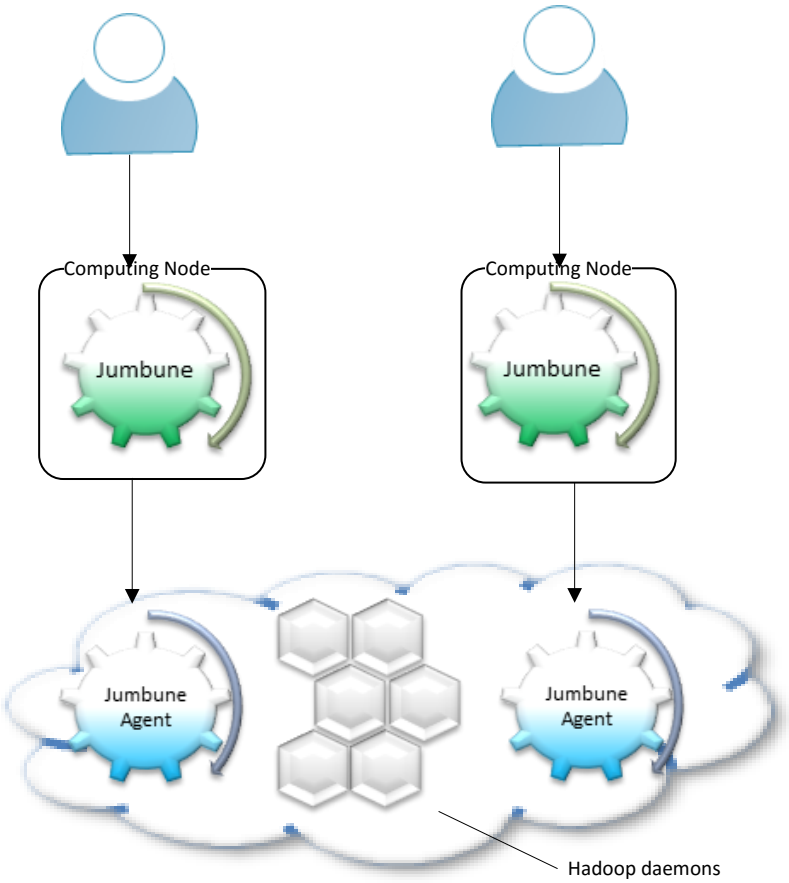
Multiple users accessing the same deployment on the same cluster



Jumbune's **Debugger** and **Job Profiler** can be used on the **staging environment** to ensure that there are no performance bottlenecks or flaws in the job when it comes to realistic data. The QA and dev team could use the same deployment of Jumbune to access the cluster and perform various functions on it as described in the previous sections.

Apart from the job profiling module, all other modules can be run from same deployment of Jumbune and accessed on a different tab or a browser. As Job profiling helps us to profile the resource usage during the various phases of job execution, it wouldn't be ideal to run it whilst another instance of the job profiling module is running. There is a check in place to stop, as this would skew the result.

Multiple Jumbune users sharing the same cluster using different deployments

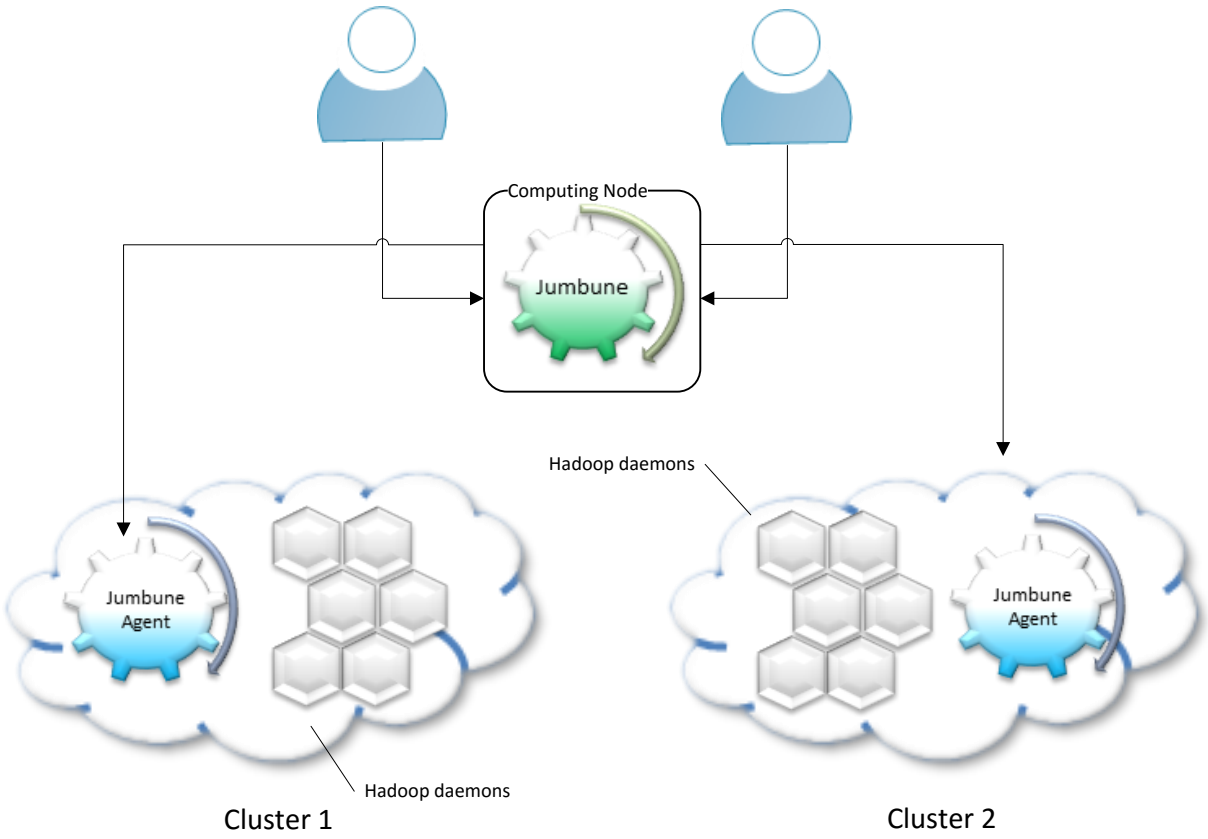


Multiple users running multiple deployments of Jumbune can use the same cluster, either to monitor it whilst running a debugging or profiling job or run validations on the HDFS data. Performance engineers or cluster administrators could keep a watch on the various cluster metrics using the **cluster monitoring tool**.

There are applications of this setup on the production cluster as well. HDFS validation module could be used to validate and verify the data that get dumped on to the production cluster's HDFS. This would ensure the quality of the data and make sure that only sane data would get processed.

The production cluster can be monitored by the cluster admins using the **Cluster Monitoring** module of Jumbune. It also provides heat maps for the nodes based on the user selection criteria. Network latency between the nodes could also be monitored using this module.

Multiple Jumbune users sharing the different cluster using same deployments



Multiple users running single deployments of Jumbune can use or share the different clusters, either to monitor it whilst running a debugging or profiling job or run validations on the HDFS data.

There are applications of this setup on the production cluster as well. It benefits both Cluster administrator and MapReduce developer by launching job and monitor cluster at the same time from different or same location.