Hortonworks Data Platform

Cluster Planning Guide for Windows

(Apr 13, 2015)

docs.hortonworks.com

Hortonworks Data Platform : Cluster Planning Guide for Windows

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, ZooKeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. **All of our technology is, and will remain, free and open source.**

Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to contact us directly to discuss your specific needs.

Table of Contents

1. Hardware Recommendations for Apache Hadoop

Hadoop and HBase workloads tend to vary a lot and it takes experience to correctly anticipate the amounts of storage, processing power, and inter-node communication that will be required for different kinds of jobs.

This document provides insights on choosing the appropriate hardware components for an optimal balance between performance and both initial as well as the recurring costs. (For a brief summary of the hardware sizing recommendations, see the Conclusion of this Guide.

Hadoop is a software framework that supports large-scale distributed data analysis on commodity servers. Hortonworks is a major contributor to open source initiatives (Apache Hadoop, HDFS, Pig, Hive, HBase, Zookeeper) and has extensive experience managing production level Hadoop clusters. Hortonworks recommends following the design principles that drive large, hyper-scale deployments. For a Hadoop or HBase cluster, it is critical to accurately predict the size, type, frequency, and latency of analysis jobs to be run. When starting with Hadoop or HBase, begin small and gain experience by measuring actual workloads during a pilot project. This way you can easily scale the pilot environment without making any significant changes to the existing servers, software, deployment strategies, and network connectivity.

Use the following sections to learn more about the suggested hardware configurations for various Hadoop clusters.

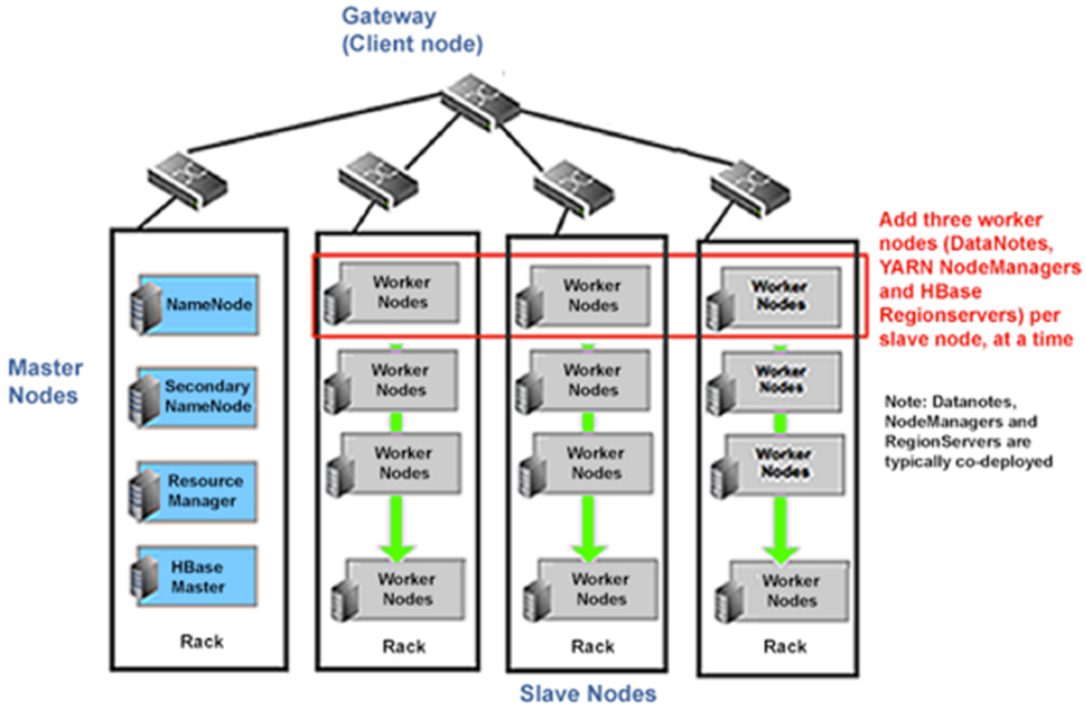
1.1. Typical Hadoop Cluster

Hadoop and HBase clusters have two types of machines:

- **Masters**: HDFS NameNode, YARN ResourceManager, and HBase Master.
- **Slaves** : HDFS DataNodes, YARN NodeManagers, and HBase RegionServers.

  The DataNodes, NodeManagers, and HBase RegionServers are co-located or co-deployed for optimal data locality.

  In addition, HBase requires the use of a separate component (ZooKeeper) to manage the HBase cluster.



Hortonworks recommends separating master and slave nodes because:

- Task/application workloads on the slave nodes should be isolated from the masters.
- Slaves nodes are frequently decommissioned for maintenance.

For evaluation purposes, it is possible to deploy Hadoop using a single-node installation (all the masters and slave processes reside on the same machine).

For a small two-node cluster, the NameNode and the ResourceManager are both on the master node, with the DataNode and NodeManager on the slave node.

Clusters of three or more machines typically use a single NameNode and ResourceManager with all the other nodes as slave nodes. A High-Availability (HA) cluster would use a primary and secondary NameNode , and might also use a primary and secondary ResourceManager .

Typically, a medium-to -large Hadoop cluster consists of a two-level or three-level architecture built with rack-mounted servers. Each rack of servers is interconnected using a 1 Gigabyte Ethernet (GbE) switch. Each rack-level switch is connected to a cluster-level switch (which is typically a larger port-density 10GbE switch). These cluster-level switches may also interconnect with other cluster-level switches or even uplink to another level of switching infrastructure.

1.2. Typical Workload Patterns For Hadoop

Disk space, I/O Bandwidth (required by Hadoop), and computational power (required for the MapReduce processes) are the most important parameters for accurate hardware

sizing. In addition, if you are installing HBase, you also need to analyze your application and its memory requirements, because HBase is a memory intensive component.

Based on the typical use cases for Hadoop, the following workload patterns are commonly observed in production environments:

Balanced Workload

If your workloads are distributed equally across the various job types (CPU bound, Disk I/O bound, or Network I/O bound), your cluster has a balanced workload pattern. This is a good default configuration for unknown or evolving workloads.

Compute-Intensive

These workloads are CPU bound and are characterized by the need of a large number of CPUs and large amounts of memory to store in-process data. (This usage pattern is typical for natural language processing or HPCC workloads.)

I/O Intensive

A typical MapReduce job (like sorting) requires very little compute power. Instead it relies more on the I/O bound capacity of the cluster (for example, if you have a lot of cold data). For this type of workload, we recommend investing in more disks per box.

Unknown or evolving workload patterns

You may not know your eventual workload patterns from the first. And typically the first jobs submitted to Hadoop in the early days are usually very different than the actual jobs you will run in your production environment. For these reasons, Hortonworks recommends that you either use the Balanced workload configuration or invest in a pilot Hadoop cluster and plan to evolve its structure as you analyze the workload patterns in your environment.

For more information, see the following section.

1.3. Early Hadoop Deployments

When a team is just starting with Hadoop or HBase, it is usually good to begin small and gain experience by measuring actual workloads during a pilot project. We recommend starting with a relatively small pilot cluster, provisioned for a **balanced** " workload.

Pilot configurations depend on the type of work you plan to do with the cluster. At a minimum, we recommend four nodes (one master and three slave nodes) with dual quad core CPUs, 24 GB memory per node, and four to six disk drives of 2 Terabyte (TB) capacity.

The minimum requirement for network is 1GigE all-to-all and can be easily achieved by connecting all of your nodes to a Gigabyte Ethernet switch. To use the spare socket for adding more CPUs in future, you can also consider using either a six- or an eight-core CPU.

Allocate at least 1 GB of RAM to each ZooKeeper server, and if possible give each ZooKeeper server its own disk.

Jumpstarting a Hadoop Cluster

One way to deploy Hadoop cluster quickly is to opt for “cloud trials”. Hortonworks distributes Hadoop through the Hortonworks Data Platform (HDP). You can install HDP in public and private clouds using Whirr, Microsoft Azure, and Amazon Web Services.

Note, however, that cloud services and virtual infrastructures are not architected for Hadoop. Cloud- or virtual-based Hadoop and HBase deployments may experience poor performance due to virtualization and suboptimal I/O architecture.

Initial Tests

The "smoke tests" that come with the Hadoop cluster are a good initial test, followed by Terasort. Some of the major server vendors offer in-factory commissioning of Hadoop clusters for an extra fee, ensuring that the cluster is working before you receive and pay for it. An indirect benefit of this approach is that if the Terasort performance is lower on-site than in the factory, the network is the most likely culprit.

Tracking Resource Usage for Pilot Deployments

Hortonworks recommends that you monitor your pilot cluster using Ganglia, Nagios, or other performance monitoring frameworks that may be in use in your data center. Use the following guidelines to decide what to monitor in your Hadoop and HBase clusters:

- Measure resource usage for CPU, RAM, Disk I/O operation per second (IOPS), and network packets sent and received. Run the actual kinds of query or analysis jobs that are of interest to your team.
- Ensure that your data subset is scaled to the size of your pilot cluster.
- Analyze the monitoring data for resource saturation. Based on this analysis, you can categorize your jobs as CPU bound, Disk I/O bound, or Network I/O bound.

Note

Most Java applications expand RAM usage to the maximum allowed. However, such jobs should not be analyzed as memory bound unless swapping happens or the JVM experiences full-memory garbage collection events. (Full-memory garbage collection events typically occur when the node appears to cease all useful work for several minutes at a time.)

- Optionally, customize your job parameters or hardware or network configurations to balance resource usage. If your jobs fall in the various workload patterns equally, you may also choose to manipulate only the job parameters and keep the hardware choices “balanced”.
- For your HBase cluster, analyze ZooKeeper as well. (Network and memory problems for HBase are often detected first in ZooKeeper.)

Tuning Job Characteristics to Resource Usage

Relating job characteristics to resource requirements can be complex. How the job is coded or the job data is represented can have a large impact on resource balance. For example, resource usage can be shifted between disk IOPS and CPU based on your choice

of compression scheme or parsing format. Per-node CPU and disk activity can be traded for inter-node bandwidth, depending on the implementation of the Map/Reduce strategy.

Reusing Pilot Machines

With a pilot cluster in place, you can start analyzing workloads patterns to identify CPU and I/O bottlenecks. Later these machines can be reused in production clusters, even if your base specs c hange. It is common to have heterogeneous Hadoop clusters, especially as they evolve in size.

1.4. System Stack Recommendations

Achieving optimal results from your Hadoop implementation begins with choosing appropriate hardware and software. The effort involved in the planning stages can pay off dramatically in terms of the performance and the total cost of ownership (TCO) associated with the environment.

The following system stack recommendations can help during planning stages:

Machine Type	Workload Pattern/ Cluster Type	Storage	Processor (# of Cores)	Memory (GB)	Network
Slave Nodes	Balanced workload	Twelve 2-3 TB disks	8	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
Slave Nodes	Compute-intensive workload	Twelve 1-2 TB disks	10	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
Slave Nodes	Storage-heavy workload	Twelve 4+ TB disks	8	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
NameNode	Balanced workload	Four or more 2-3 TB RAID 10 with spares	8	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
ResourceManager	Balanced workload	Four or more 2-3 TB RAID 10 with spares	8	128-256	1 GB onboard, 2x10 GBE mezzanine/ external

1.5. Server Node Hardware Recommendations

Use the following recommendations as best practices for selecting the number of nodes, storage options per node (number of disks, size of disks, MTBF, and the replication cost of disk failures), compute power per node (sockets, cores, clock speed), RAM per node, and network capability (number, speed of ports).

Note

Hadoop cluster nodes do not require many features typically found in an enterprise data center server.

1.5.1. Hardware for Slave Nodes

The following recommendations are based on Hortonworks' experience in production data centers:

Server Platform

Typically, dual-socket servers are optimal for Hadoop deployments. For medium to large clusters, using these servers is a best choice over entry-level servers, because of their load-balancing and parallelization capabilities.

Storage Options

Your disk drives should have good MTBF numbers, as slave nodes in Hadoop suffer routine probabilistic failures.

SFF disks are being adopted in some configurations for better disk bandwidth. Hortonworks strongly recommends using either SATA or SAS interconnects.

If you have a large number of disks per server, we recommend using two disk controllers, so that the I/O load can be shared across multiple cores.

Note

We do not recommend using RAID on Hadoop slave machines. Hadoop assumes probabilistic disk failure and orchestrates data redundancy across all the slave nodes.

Memory

Memory can be provisioned at commodity prices on low-end server motherboards. Extra RAM will be consumed either by your Hadoop applications (typically when more processes are run in parallel) or by the intrastructure used for caching disk data to improve performance.

To retain the option of adding more memory to your servers in the future, ensure that you have space to do this alongside the initial memory modules.

Power Considerations

Power is a major concern when designing Hadoop clusters. Instead of automatically purchasing the biggest and fastest nodes, analyze the power utilization for your existing hardware. We have observed huge savings in pricing and power by avoiding fastest CPUs, redundant power supplies, etc.

For slave nodes, a single power supply unit (PSU) is sufficient, but for master servers use redundant PSUs. Server designs that share PSUs across adjacent servers can offer increased reliability without increased cost.

Machines for cloud data centers are designed to reduce cost and power, and are lightweight. If you are purchasing in large volume, we recommend evaluating these stripped-down "cloud servers".

Network

This is the most challenging parameter to estimate because Hadoop workloads vary a lot. The key is buying enough network capacity at reasonable cost so that all nodes in the cluster can communicate with each other at reasonable speeds.

Design the network so that you retain the option of adding more racks of Hadoop/HBase servers.

Minimize congestion at critical points in the network under realistic loads. Generally accepted oversubscription ratios are around 4:1 at the server access layer, and 2:1 between the access layer and the aggregation layer or core. Lower oversubscription ratios can be considered if higher performance is required.

Configure dedicated switches for the cluster, instead of trying to allocate a virtual circuit in existing switches. The load of a Hadoop cluster would impact the rest of the users of the switch.

"Deep buffering" is preferable to low-latency in switches. Enabling Jumbo Frames across the cluster can improve bandwidth, and might also provide packet integrity.

1.5.2. Hardware for Master Nodes: NameNodes, ResourceManagers and HBase Masters

The master nodes have significantly different storage and memory requirements than the slave nodes. The following paragraphs discuss storage considerations.

Storage Options

We recommend using dual NameNode servers: one primary and one secondary. Both NameNode servers should have highly reliable storage for their namespace storage and edit log journaling. Hardware RAID and/or reliable network storage are justifiable options.

The master servers should have at least four redundant storage volumes, some local and some networked.

Multiple vendors sell NAS software. It is important to check their specifications before you invest in any NAS software.

Storage Options for ResourceManager Servers

ResourceManager servers do not need RAID storage because they save their persistent state to HDFS. The ResourceManager server can actually be run on a slave node with a bit of extra RAM. However, if you use the same hardware specifications for the ResourceManager servers and the NameNode server, you retain the possibility of migrating the NameNode to the same server as the ResourceManager. If there is a NameNode failure, a copy of the NameNode's state can be saved to network storage.

1.6. Hardware for HBase

As a general rule the more memory HBase has, the better it can cache read requests. Each slave node in an HBase cluster (RegionServer) maintains a number of regions (regions are the chunks of the data in memory). For large clusters, it is important to ensure that the HBase Master and the NameNode run on separate server machines.

Note that in large scale deployments, ZooKeeper nodes are not co-deployed with the Hadoop/HBase slave nodes.

Choosing Storage Options

In a distributed setup, HBase stores its data in Hadoop DataNodes. To get maximum read/write locality, HBase RegionServers and DataNodes should be co-deployed on the same machines. Therefore all the recommendations for the DataNode and NodeManager hardware setup are also applicable to the RegionServers. Depending on whether your HBase applications are read/write or processing oriented, you must balance the number of disks with the number of CPU cores available. Typically, you should have at least one core per disk.

Memory Sizing

HBase Master nodes(s) are not as compute-intensive as a typical RegionServer or the NameNode server. Therefore a more modest memory setting can be chosen for the HBase master. RegionServer memory requirements depend heavily on the workload characteristics of your HBase cluster. Although over-provisioning for memory benefits all the workload patterns, with very large heap sizes Java's stop-the-world GC pauses may cause problems.

While running HBase cluster with Hadoop core, ensure that you over-provision the memory for Hadoop MapReduce and other compute-intensive processes, in addition to HBase memory.

1.7. Scalability

You can scale a Hadoop cluster by adding new servers or whole server racks to the cluster, and by increasing memory in the master nodes. Use the following guidelines to scale your existing Hadoop cluster:

- Ensure there is potential free space in the data center near the Hadoop cluster. This space should be able to accommodate the power budget for more racks.
- Plan your network so that you can add servers over time.
- It might be possible to add more disks and RAM to the existing servers, and extra CPUs if the servers have spare sockets, expanding an existing cluster without adding more racks or modifying the network.
- When performing a hardware upgrade in a live cluster, plan your expansion carefully beforehand. The process can take considerable time and effort.
- As your cluster grows, consider adding more memory to the master servers.