Hortonworks Data Platform

Cluster Planning Guide

(September 2, 2016)

docs.cloudera.com

Hortonworks Data Platform: Cluster Planning Guide

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 Conclusion.)

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.

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. Additionally, 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 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.

1.3. Early 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.

For pilot deployments, you can start with 1U/machine and use the following recommendations:

Two quad core CPUs | 12 GB to 24 GB memory | 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. In order to use the spare socket for adding more CPUs in future, you can also consider using either a six or an eight core CPU.

For small to medium HBase clusters, provide each ZooKeeper server with around 1GB of RAM and, if possible, its own disk.

Jump-start - Hadoop Cluster

One way to quickly deploy Hadoop cluster, is to opt for "cloud trials" or use virtual infrastructure. Hortonworks makes the distribution available through Hortonworks Data Platform (HDP). HDP can be easily installed in public and private clouds using Whirr, Microsoft Azure, and Amazon Web Services.

To contact Hortonworks Technical Support, please log a case at: https://support.hortonworks.com/ . If you are currently not an official Hortonworks Customer or Partner, then please seek assistance on our Hortonworks Forums at: //community.hortonworks.com/answers/

However, note that cloud services and virtual infrastructures are not architected for Hadoop. Hadoop and HBase deployments in this case, might experience poor performance due to virtualization and suboptimal I/O architecture.

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 sub-set 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 are 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, also analyze ZooKeeper, because network and memory problems for HBase are often detected first in ZooKeeper.

Challenges - 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 cost 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.

Furthermore, Amdahl's Law shows how resource requirements can change in grossly non-linear ways with changing demands: a change that might be expected to reduce computation cost by 50% may instead cause a 10% change or a 90% change in net performance.

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 change. It is common to have heterogeneous Hadoop clusters, especially as they evolve in size.

Tip

To achieve a positive return on investment (ROI), ensure that the machines in your pilot clusters are less than 10% of your eventual production cluster.

1.4. 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.

1.4.1. 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. In terms of density, select server hardware that fits into a low number of rack units. Typically, 1U or 2U servers are used in 19" racks or cabinets.

Storage options

For general-purpose Hadoop applications, we recommend using a relatively large number of hard drives (typically eight to twelve SATA LFF drives) per server. Currently typical capacity in production environments is around 2 TB per drive. Highly I/O intensive environments may require using 12 x 2 TB SATA drives. The optimal balance between cost and performance is generally achieved with 7,200 RPM SATA drives. If your current or predicted storage is experiencing a significant growth rate you should also consider using 3 TB disks.

SFF disks are being adopted in some configurations for better disk bandwidth. We recommend that you monitor your cluster for any potential disk failures because more disks will increase the rate of disk failures. If you do have large number of disks per server, we recommend that you use two disk controllers, so that the I/O load can be shared across multiple cores. Hortonworks strongly recommends only using either SATA or SAS interconnects.

On an HDFS cluster using a low-cost reliable storage option, you will observe that the old data stays on the cluster indefinitely and your storage demands grow quickly. With 12-drive systems, you typically get 24 TB or 36 TB per node. Using this storage capacity in a node is only practical with Hadoop release 1.0.0 or later (because the failures are handled gracefully allowing machines to continue serving from their remaining disks).

Hadoop is storage intensive and seek efficient, but does not require fast and expensive hard drives. If your workload pattern is not I/O intensive, it is safe to add only four or six disks per node. Note that power costs are proportional to the number of disks and not storage capacity per disk. We therefore recommend that you add disks to increase storage only and not simply for seeks.

Note

RAID vs. JBOD:

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

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

Your slave nodes do not need expensive support contracts that offer services like replacement of disks within two hours or less. Hadoop is designed to adapt to slave node disk failure. Treat maintenance activity for the slave nodes as an ongoing task rather than an emergency.

It is good to be able to swap out disks without taking the server out of the rack, though switching them off (briefly) is an inexpensive operation in a Hadoop cluster.

Memory sizing

It is critical to provide sufficient memory to keep the processors busy without swapping and without incurring excessive costs for non-standard motherboards. Depending on the number of cores, your slave nodes typically require 24 GB to 48 GB of RAM for Hadoop

applications. For large clusters, this amount of memory provides sufficient extra RAM (approximately 4 GB) for the Hadoop framework and for your query and analysis processes (HBase and/or Map/Reduce).

To detect and correct random transient errors introduced due to thermodynamic effects and cosmic rays, we strongly recommend using error correcting code (ECC) memory. Error-correcting RAM allows you to trust the quality of your computations. Some parts (chip-kill/ chip spare) have been shown to offer better protection than traditional designs, as they show less recurrence of bit errors. (See, DRAM Errors in the Wild: A Large-Scale Field Study, Schroeder et al, 2009 .)

If you want to retain the option of adding more memory to your servers in future, ensure there is space to do this alongside the initial memory modules.

Memory provisioning

Memory can also be provisioned at commodity prices on low-end server motherboards. It is typical to over-provision memory. The unused RAM will be consumed either by your Hadoop applications (typically when you run more processes in parallel) or by the infrastructure (used for caching disk data to improve performance).

Processors

Although it is important to understand your workload pattern, for most systems we recommend using medium clock speed processors with less than two sockets. For most workloads, the extra performance per node is not cost-effective. For large clusters, use at least two quad core CPU for the slave machines.

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.

Vendors today are building machines for cloud data centers that are designed to reduce cost, power, and are light-weight. Supermicro, Dell, and HP all have such product lines for cloud providers. So if you are buying in large volume, we recommend evaluating these stripped-down “cloud servers”.

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.

Some co-location sites bill based on the maximum-possible power budget and not the actual budget. In such a location the benefits of the power saving features of the latest CPUs are not realized completely. We therefore recommend checking the power billing options of the site in advance.

Note

Power consumption of the cluster:

Electricity and cooling account for 33.33% to 50% of the equipment total life cycle cost in the modern data centers.

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. Large clusters typically use dual 1 GB links for all nodes in each 20-node rack and 2*10 GB interconnect links per rack going up to a pair of central switches.

A good network design considers the possibility of unacceptable 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. Additionally, we also recommend having 1 GE oversubscription between racks.

It is critical to have dedicated switches for the cluster instead of trying to allocate a VC in existing switches - the load of a Hadoop cluster would impact the rest of the users of the switch. It is also equally critical to work with the networking team to ensure that the switches suit both Hadoop and their monitoring tools.

Design the networking so as to retain the option of adding more racks of Hadoop/HBase servers. Getting the networking wrong can be expensive to fix. The quoted bandwidth of a switch is analogous to the miles per gallon ratings of an automobile - you are unlikely to replicate it. ''Deep buffering'' is preferable to low-latency in switches. Enabling Jumbo Frames across the cluster improves bandwidth through better checksums and possibly may also provide packet integrity.

Note

Network strategy for your Hadoop clusters

Analyze the ratio of network-to-computer cost. Ensure that the network cost is always around 20% of your total cost. Network costs should include your complete network, core switches, rack switches, any network cards needed, and so forth. Hadoop was designed with commodity hardware in mind.

1.4.2. Hardware for Master Nodes

The master nodes, being unique, have significantly different storage and memory requirements than the slave nodes. The following paragraphs discuss some of the memory/storage trade-offs in some detail.

For hard sizing guidelines for small (5-50 nodes) and medium-to-large (100s to 1000s of nodes) clusters, see the Conclusion .

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. Typically, 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, but each can be relatively small (typically 1TB).

Note

The RAID disks on the master nodes are a good place to consider support contracts. We recommend including an on-site disk replacement option in your support contract so that a failed RAID disk can be replaced quickly.

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

Storage options for ResourceManager servers

In actuality 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, using the same hardware specifications for the ResourceManager servers as for the NameNode server provides the possibility of migrating the NameNode to the same server as the ResourceManager in the case of NameNode failure and a copy of the NameNode's state can be saved to the network storage.

Memory sizing

The amount of memory required for the master nodes depends on the number of file system objects (files and block replicas) to be created and tracked by the NameNode. 64 GB of RAM supports approximately 100 million files. Some sites are now experimenting with 128GB of RAM, for even larger namespaces.

Processors

NameNodes and their clients are very "chatty". We therefore recommend providing 16 or even 24 CPU cores to handle messaging traffic for the master nodes.

Network

Providing multiple network ports and 10 GB bandwidth to the switch is also acceptable (if the switch can handle it).

1.5. Hardware for HBase

HBase uses different types of caches to fill up memory, and 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.

In addition, when running HBase cluster with Hadoop core, you must ensure that you over-provision the memory for Hadoop MapReduce by at least 1 GB to 2 GB per task on top of the HBase memory.

1.6. Other Issues

Weight

The storage density of the latest generation of servers means that the weight of the racks needs to be taken into account. You should verify that the weight of a rack is not more than the capacity of the data center's floor.

Scalability

It is easy to scale a Hadoop cluster by adding new servers or whole server racks to the cluster and increasing the memory in the master nodes to deal with the increased load. This will generate a lot of "rebalancing traffic" at first, but will deliver extra storage and computation. Because the master nodes do matter, we recommend that you pay the premiums for those machines.

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 the network to cope with more servers
- It might be possible to add more disks and RAM to the existing servers - and extra CPUs if the servers have spare sockets. This can expand an existing cluster without adding more racks or network changes.
- To perform a hardware upgrade in a live cluster can take considerable time and effort. We recommend that you plan the expansion one server at a time.
- CPU parts do not remain on the vendors price list forever. If you do plan to add a second CPU, consult with your reseller on when they will cut the price of CPUs that your existing parts and buy these parts when available. This typically takes at least 18 months time period.
- You are likely to need more memory in the master servers.

Support contracts

The concept to consider here is "care for the master nodes, keep an eye on the slave nodes". You do not need traditional enterprise-class support contracts for the majority of the nodes in the cluster, as their failures are more of a statistics issue than a crisis. The money saved in support can go into more slave nodes.

Commissioning

Hortonworks plans to cover the best practices commissioning a Hadoop cluster in a future document. For now, note that 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. This can have a direct benefit in ensuring that the cluster is working before you receive and pay for it. There is an indirect benefit in that if the Terasort performance is lower on-site than in-factory, the network is the likely culprit which makes it is possible to track down the problem faster.

1.7. Conclusion

Achieving optimal results from a Hadoop implementation begins with choosing the correct hardware and software stacks. 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 composite system stack recommendations can help benefit organizations in the planning stages:

Table 1.1 Sizing Recommendations

Machine Type	Workload Pattern/ Cluster Type	Storage[1] [10]	Processor (# of Cores)	Memory (GB)	Network
Slaves	Balanced workload	Twelve 2-3 TB disks	8	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
	Compute-intensive workload	Twelve 1-2 TB disks	10	128-256	1 GB onboard, 2x10 GBE mezzanine/ external
	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] Reserve at least 2.5 GB of hard drive space for each version of HDP to be installed.

2. File System Partitioning Recommendations

Setting Up File System Partitions

Use the following as a base configuration for all nodes in your cluster:

- Root partition: OS and core program files
- Swap: Size 2X system memory

Partitioning Recommendations for Slave Nodes

- Hadoop Slave node partitions: Hadoop should have its own partitions for Hadoop files and logs. Drives should be partitioned using ext3, ext4, or XFS, in that order of preference. HDFS on ext3 has been publicly tested on the Yahoo cluster, which makes it the safest choice for the underlying file system. The ext4 file system may have potential data loss issues with default options because of the "delayed writes" feature. XFS reportedly also has some data loss issues upon power failure. Do not use LVM; it adds latency and causes a bottleneck.
- On slave nodes only, all Hadoop partitions should be mounted individually from drives as "/grid/[0-n]".
- Hadoop Slave Node Partitioning Configuration Example:
  - /root - 20GB (ample room for existing files, future log file growth, and OS upgrades)
  - /grid/0/ - [full disk GB] first partition for Hadoop to use for local storage
  - /grid/1/ - second partition for Hadoop to use
  - /grid/2/ - ...

Redundancy (RAID) Recommendations

- Master nodes – Configured for reliability (RAID 10, dual Ethernet cards, dual power supplies, etc.)
- Slave nodes – RAID is not necessary, as failure on these nodes is managed automatically by the cluster. All data is stored across at least three different hosts, and therefore redundancy is built-in. Slave nodes should be built for speed and low cost.

Further Reading

The following additional documentation may be useful:

- CentOS partitioning documentation
- Reference architectures from other Hadoop clusters: Hadoop Reference Architectures