Hortonworks Data Platform

Getting Started Guide

(Mar 20, 2013)

docs.hortonworks.com

Hortonworks Data Platform: Getting Started 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.

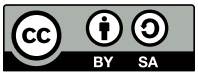
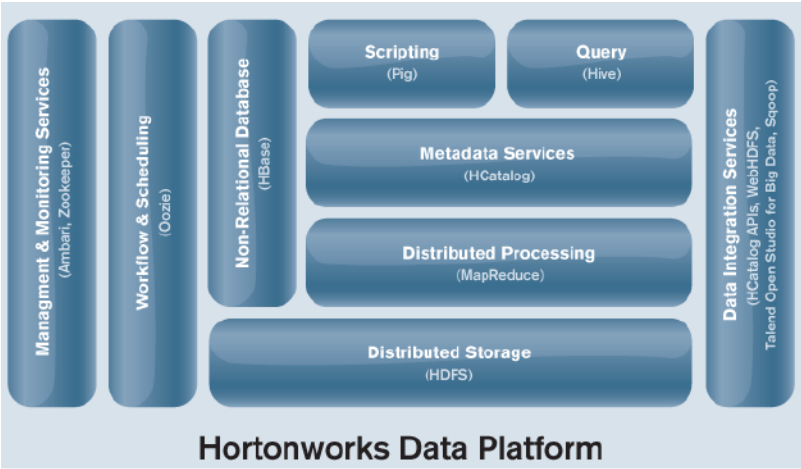


Table of Contents

1. About Hortonworks Data Platform

Hortonworks Data Platform (HDP) is an open source distribution powered by Apache Hadoop. HDP provides you with the actual Apache-released versions of the components with all the necessary bug fixes to make all the components interoperable in your production environments. It is packaged with an easy to use installer (HDP Installer) that deploys the complete Apache Hadoop stack to your entire cluster and provides the necessary monitoring capabilities using Ganglia and Nagios. The HDP distribution consists of the following components:

1. Core Hadoop platform (Hadoop HDFS and Hadoop MapReduce)
2. Non-relational database (Apache HBase)
3. Metadata services (Apache HCatalog)
4. Scripting platform (Apache Pig)
5. Data access and query (Apache Hive)
6. Workflow scheduler (Apache Oozie)
7. Cluster coordination (Apache Zookeeper)
8. Management and monitoring (Apache Ambari)
9. Data integration services (HCatalog APIs, WebHDFS, Talend Open Studio for Big Data, and Apache Sqoop)
10. Distributed log management services (Apache Flume)
11. Machine learning library (Mahout)



To learn more about the distribution details and the component versions, see the Release Notes. All components are official Apache releases of the most recent stable versions available. Hortonworks' philosophy is to do patches only when absolutely necessary to assure interoperability of the components. Consequently, there are very few patches in the

HDP, and they are all fully documented. Each of the HDP components have been tested rigorously prior to the actual Apache release. To learn more about the testing strategy adopted at Hortonworks, Inc., see: Delivering high-quality Apache Hadoop releases.

2. Understanding Hadoop Ecosystem

The following sections provides an overview of the Hadoop Ecosystem, typical cluster patterns, and deployment options.

2.1. Apache Hadoop core components

Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of commodity computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each providing computation and storage. Rather than rely on hardware to deliver high-availability, the framework itself is designed to detect and handle failures at the application layer, thus delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

HDFS (storage) and **MapReduce** (processing) are the two core components of Apache Hadoop. The most important aspect of Hadoop is that both HDFS and MapReduce are designed with each other in mind and each are co-deployed such that there is a single cluster and thus provides the ability to move computation to the data not the other way around. Thus, the storage system is not physically separate from a processing system.

Hadoop Distributed File System (HDFS)

HDFS is a distributed file system that provides high-throughput access to data. It provides a limited interface for managing the file system to allow it to scale and provide high throughput. HDFS creates multiple replicas of each data block and distributes them on computers throughout a cluster to enable reliable and rapid access.

Note

A file consists of many blocks (large blocks of 64MB and above).

The main components of HDFS are as described below:

- NameNode is the master of the system. It maintains the name system (directories and files) and manages the blocks which are present on the DataNodes.
- DataNodes are the slaves which are deployed on each machine and provide the actual storage. They are responsible for serving read and write requests for the clients.
- Secondary NameNode is responsible for performing periodic checkpoints. In the event of NameNode failure, you can restart the NameNode using the checkpoint.

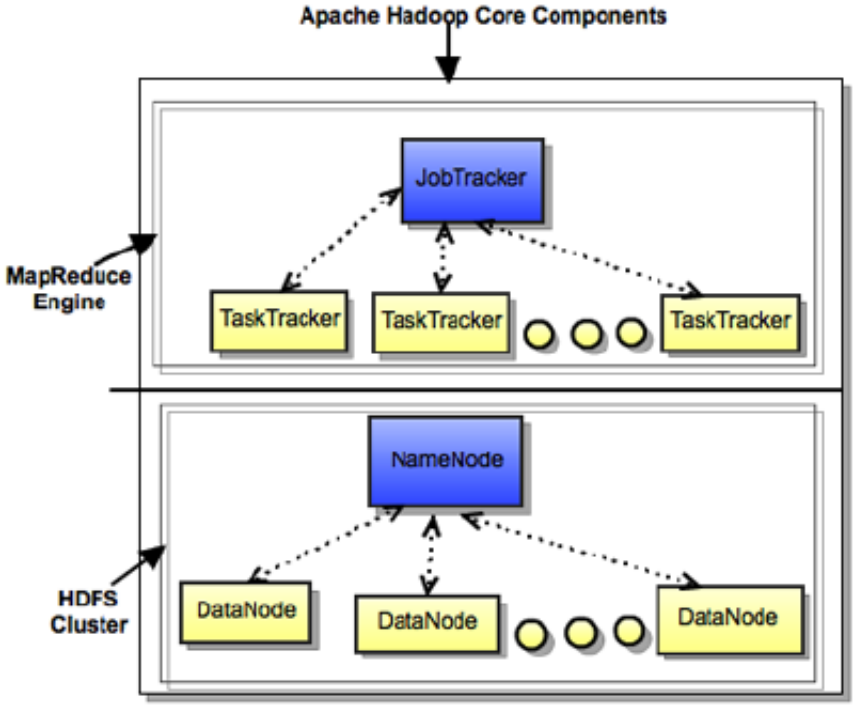
MapReduce

MapReduce is a framework for performing distributed data processing using the MapReduce programming paradigm. In the MapReduce paradigm, each job has a user-defined map phase (which is a parallel, share-nothing processing of input; followed by a user-defined reduce phase where the output of the map phase is aggregated). Typically, HDFS is the storage system for both input and output of the MapReduce jobs.

The main components of MapReduce are as described below:

- JobTracker is the master of the system which manages the jobs and resources in the cluster (TaskTrackers). The JobTracker tries to schedule each map as close to the actual data being processed i.e. on the TaskTracker which is running on the same DataNode as the underlying block.
- TaskTrackers are the slaves which are deployed on each machine. They are responsible for running the map and reduce tasks as instructed by the JobTracker.
- JobHistoryServer is a daemon that serves historical information about completed applications. Typically, JobHistory server can be co-deployed with JobTracker, but we recommend to run it as a separate daemon.

The following illustration provides details of the core components for the Hadoop stack.



2.2. Apache Pig

Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets. At the present time,

Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs. Pig's language layer currently consists of a textual language called Pig Latin, which is easy to use, optimized, and extensible.

2.3. Apache Hive

Hive is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems.

It provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. Hive also allows traditional map/reduce programmers to plug in their custom mappers and reducers when it is inconvenient or inefficient to express this logic in HiveQL.

2.4. Apache HCatalog

HCatalog is a metadata abstraction layer for referencing data without using the underlying filenames or formats. It insulates users and scripts from how and where the data is physically stored.

WebHCat provides a REST-like web API for HCatalog and related Hadoop components. Application developers make HTTP requests to access the Hadoop MapReduce, Pig, Hive, and HCatalog DDL from within the applications. Data and code used by WebHCat is maintained in HDFS. HCatalog DDL commands are executed directly when requested. MapReduce, Pig, and Hive jobs are placed in queue by WebHCat and can be monitored for progress or stopped as required. Developers also specify a location in HDFS into which WebHCat should place Pig, Hive, and MapReduce results.

2.5. Apache HBase

HBase (Hadoop DataBase) is a distributed, column oriented database. HBase uses HDFS for the underlying storage. It supports both batch style computations using MapReduce and point queries (random reads).

The main components of HBase are as described below:

- HBase Master is responsible for negotiating load balancing across all Region Servers and maintain the state of the cluster. It is not part of the actual data storage or retrieval path.
- RegionServer is deployed on each machine and hosts data and processes I/O requests.

2.6. Apache ZooKeeper

ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services which are very useful for a variety of distributed systems. HBase is not operational without ZooKeeper.

2.7. Apache Oozie

Apache Oozie is a workflow/coordination system to manage Hadoop jobs.

2.8. Apache Sqoop

Apache Sqoop is a tool designed for efficiently transferring bulk data between Hadoop and structured datastores such as relational databases.

2.9. Apache Flume

Flume is a top level project at the Apache Software Foundation. While it can function as a general purpose event queue manager, in the context of Hadoop it is most often used as a log aggregator, collecting log data from many diverse sources and moving them to a centralized data store.

2.10. Apache Mahout

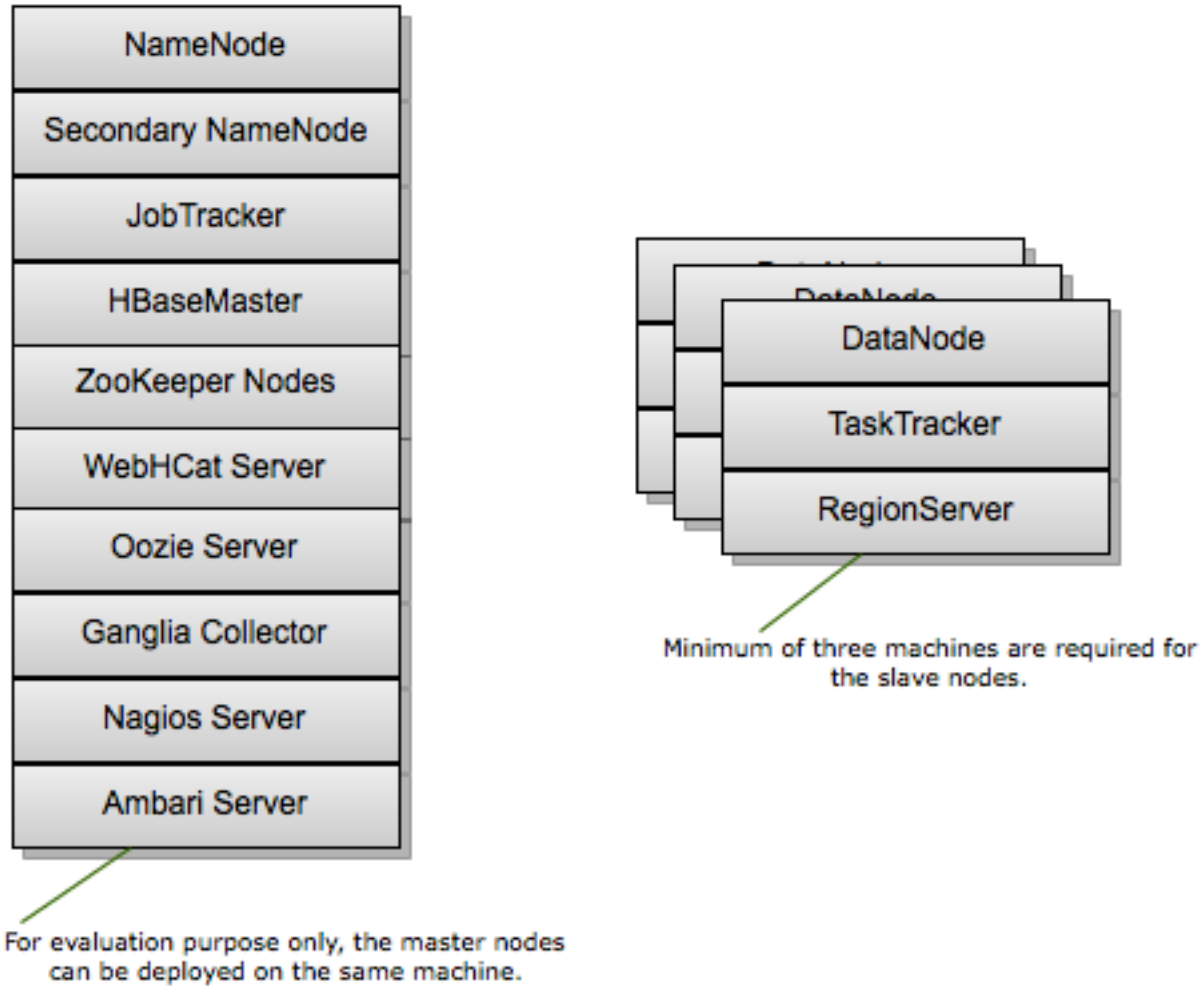
Mahout is a scalable machine learning library that implements many different approaches to machine learning. The project currently contains implementations of algorithms for classification, clustering, frequent item set mining, genetic programming and collaborative filtering. Mahout is scalable along three dimensions: It scales to reasonably large data sets by leveraging algorithm properties or implementing versions based on Apache Hadoop.

3. Typical Hadoop Cluster

A typical Hadoop cluster comprises of machines based on the various machine roles (master, slaves, and clients).

For more details, see: Machine Roles In A Typical Hadoop Cluster [7]. The following illustration provides information about a typical Hadoop cluster.

A smallest cluster can be deployed using a minimum of four machines (for evaluation purpose only). One machine can be used to deploy all the master processes (NameNode, Secondary NameNode, JobTracker, HBase Master), Hive Metastore, WebHCat Server, and ZooKeeper nodes. The other three machines can be used to co-deploy the slave nodes (TaskTrackers, DataNodes, and RegionServers).



A minimum of three machines is required for the slave nodes in order to maintain the replication factor of three. For more details, see: Data Replication in Apache Hadoop

Machine roles in a typical Hadoop cluster:

In Hadoop and HBase, the following two types of machines are available:

- Masters (HDFS NameNode, Secondary NameNode, MapReduce JobTracker, and the HBase Master)

Note

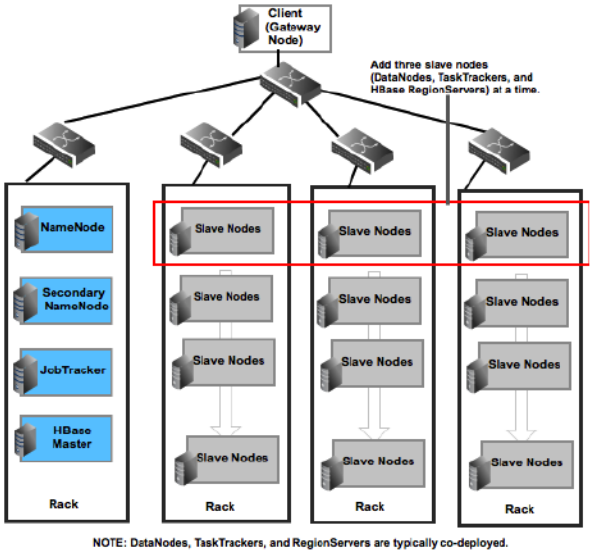
It is recommended to add only limited number of disks to the master nodes, because the master nodes do not have high storage demands.

- Slaves (HDFS DataNodes, MapReduce TaskTrackers, and HBase RegionServers)

Additionally, we strongly recommend that you use separate client machines for performing the following tasks:

- Load data in the HDFS cluster
- Submit MapReduce jobs (describing how to process the data)
- Retrieve or view the results of the job after its completion
- Submit Pig or Hive queries

Based on the recommended settings for the client machines, the following illustration provides details of a Hadoop cluster:



4. HDP Deployment Options

Hortonworks Data Platform (HDP), powered by Apache Hadoop, is a massively scalable 100% open source platform for storing, processing, and analyzing large volumes of data. HDP provides an open, stable, and highly extensible platform that makes it easier to integrate Apache Hadoop with existing data architectures and maximize the value of the data flowing through your business.

HDP can be deployed using any one of the following options:

- Easy tryouts for HDP
- Automated Install (Ambari)
- Deploying Manually (RPMs)
- Using Shell Scripts (gsInstaller)

4.1. Easy tryouts for HDP

HDP offers a Virtual Sandbox environment where you can get up and running with Apache Hadoop in a simplified, pre-installed and configured environment. Note that this is a single-node virtual environment for learning and experimenting with Apache Hadoop. To deploy Hortonworks Data Platform across multiple nodes for proof-of-concept, we recommend that you use other deployment options.

For more information on easy tryouts for HDP, see: http://hortonworks.com/get-started/developer/.

4.2. Automated Install (Ambari)

Hortonworks Data Platform provides Apache Ambari for end-to-end management and monitoring application of Apache Hadoop. With Ambari, you can deploy Hadoop, centrally manage configuration changes, monitor host metrics and create alerts for all the nodes in your cluster.

For more information, see: Using automated install (Ambari).

4.3. Deploying Manually (RPMs)

Hortonworks Data Platform also provides the option to install Apache Hadoop manually using RPMs.

For more information, see: Deploying Hortonworks Data Platform manually (RPMs).

4.4. Using Shell Scripts (gsInstaller)

Hortonworks Data Platform also provides a command line installer (shell scripts) for deploying your Apache Hadoop stack. You can use gsInstaller to deploy secure and non-secure Hadoop clusters.

For more information, see: Deploying Hortonworks Data Platform using shell scripts (gsInstaller).

Important

gsInstaller is **deprecated** as of HDP 1.2.0 and will not be made available in future minor and major releases of HDP. We encourage you to consider Manual Install (RPMs) or Automated Install (Ambari).