**Apache Hadoop**

## What Is Apache Hadoop?

The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

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

The project includes these modules:

* Hadoop Common: The common utilities that support the other Hadoop modules.
* Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.
* Hadoop YARN: A framework for job scheduling and cluster resource management.
* Hadoop MapReduce: A YARN-based system for parallel processing of large data sets.

**Apache Hadoop** (High-availability distributed object-oriented platform) is an [open-source](http://en.wikipedia.org/wiki/Open_source) [software framework](http://en.wikipedia.org/wiki/Software_framework) that supports [data-intensive](http://en.wikipedia.org/wiki/Big_data) [distributed applications](http://en.wikipedia.org/wiki/Distributed_computing), licensed under the Apache v2 license. It supports the running of applications on large clusters of [commodity hardware](http://en.wikipedia.org/wiki/Commodity_hardware). Hadoop was derived from [Google](http://en.wikipedia.org/wiki/Google)'s [MapReduce](http://en.wikipedia.org/wiki/MapReduce) and [Google File System](http://en.wikipedia.org/wiki/Google_File_System) (GFS) papers.

The Hadoop framework transparently provides both reliability and data motion to applications. Hadoop implements a computational paradigm named [MapReduce](http://en.wikipedia.org/wiki/MapReduce),[[2]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-2) where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster. In addition, it provides a distributed file system that stores data on the compute nodes, providing very high aggregate bandwidth across the cluster. Both map/reduce and the distributed file system are designed so that node failures are automatically handled by the framework.[[3]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-3) It enables applications to work with thousands of computation-independent computers and [petabytes](http://en.wikipedia.org/wiki/Petabytes) of data. The entire Apache Hadoop “platform” is now commonly considered to consist of the Hadoop kernel, [MapReduce](http://en.wikipedia.org/wiki/MapReduce) and [Hadoop Distributed File System](http://en.wikipedia.org/wiki/HDFS" \l "Hadoop_Distributed_File_System) (HDFS), as well as a number of related projects – including [Apache Hive](http://en.wikipedia.org/wiki/Apache_Hive), Apache [HBase](http://en.wikipedia.org/wiki/HBase), and others.[[4]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-4)

Hadoop is written in the [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) programming language and is an [Apache top-level project](http://en.wikipedia.org/wiki/Apache_Software_Foundation" \l "Projects) being built and used by a global community of contributors.[[5]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-poweredby-5) Hadoop and its related projects (Hive, HBase, [Zookeeper](http://en.wikipedia.org/wiki/Apache_ZooKeeper), and so on) have many contributors from across the ecosystem.[[6]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-6) Though Java code is most common, any programming language can be used with "streaming" to implement the "map" and "reduce" parts of the system.[[7]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-7)

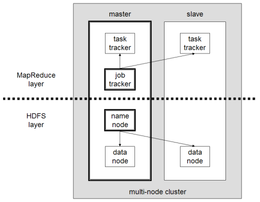
**History**

Hadoop was created by [Doug Cutting](http://en.wikipedia.org/wiki/Doug_Cutting) and [Mike Cafarella](http://en.wikipedia.org/wiki/Mike_Cafarella)[[8]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-8) in 2005. Cutting, who was working at Yahoo at the time,[[9]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-9) named it after his son's toy elephant.[[10]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-10) It was originally developed to support distribution for the [Nutch](http://en.wikipedia.org/wiki/Nutch) search engine project.[[11]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-11)

**Architecture**

Hadoop consists of the *Hadoop Common* which provides access to the file systems supported by Hadoop. The Hadoop Common package contains the necessary [Java ARchive (JAR)](http://en.wikipedia.org/wiki/JAR_(file_format)) files and scripts needed to start Hadoop. The package also provides source code, documentation and a contribution section that includes projects from the Hadoop Community.[[*citation needed*](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

For effective scheduling of work, every Hadoop-compatible file system should provide location awareness: the name of the rack (more precisely, of the network switch) where a worker node is. Hadoop applications can use this information to run work on the node where the data is, and, failing that, on the same rack/switch, reducing backbone traffic. The [Hadoop Distributed File System](http://en.wikipedia.org/wiki/Hadoop_Distributed_File_System) (HDFS) uses this method when replicating data to try to keep different copies of the data on different racks. The goal is to reduce the impact of a rack power outage or switch failure, so that even if these events occur, the data may still be readable.[[12]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-12)



http://bits.wikimedia.org/static-1.22wmf12/skins/common/images/magnify-clip.png

A multi-node Hadoop cluster

A small Hadoop cluster will include a single master and multiple worker nodes. The master node consists of a JobTracker, TaskTracker, NameNode and DataNode. A slave or *worker node* acts as both a DataNode and TaskTracker, though it is possible to have data-only worker nodes and compute-only worker nodes. These are normally used only in nonstandard applications.[[13]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-michael-noll.com_2-13) Hadoop requires [Java Runtime Environment (JRE)](http://en.wikipedia.org/wiki/JRE) 1.6 or higher. The standard start-up and shutdown scripts require [Secure Shell](http://en.wikipedia.org/wiki/Secure_Shell) (ssh) to be set up between nodes in the cluster.[[14]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-michael-noll.com_1-14)

In a larger cluster, the HDFS is managed through a dedicated NameNode server to host the file system index, and a secondary NameNode that can generate snapshots of the namenode's memory structures, thus preventing file-system corruption and reducing loss of data. Similarly, a standalone JobTracker server can manage job scheduling. In clusters where the Hadoop MapReduce engine is deployed against an alternate file system, the NameNode, secondary NameNode and DataNode architecture of HDFS is replaced by the file-system-specific equivalent. Lead statistician Joe Toner has famously referred to Hadoop as "Poor man's [Teradata](http://en.wikipedia.org/wiki/Teradata)".[[*citation needed*](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

**File Systems**

**Hadoop Distributed File System**

HDFS is a distributed, scalable, and portable file system written in [Java](http://en.wikipedia.org/wiki/Java_(software_platform)) for the Hadoop framework. Each node in a Hadoop instance typically has a single namenode; a cluster of datanodes form the HDFS cluster. The situation is typical because each node does not require a datanode to be present. Each datanode serves up blocks of data over the network using a block protocol specific to HDFS. The file system uses the [TCP/IP](http://en.wikipedia.org/wiki/TCP/IP) layer for communication. Clients use [Remote procedure call](http://en.wikipedia.org/wiki/Remote_procedure_call) (RPC) to communicate between each other. HDFS stores large files (an ideal file size is a multiple of 64 [MB](http://en.wikipedia.org/wiki/Megabyte)[[15]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-hadoop.apache.org-15)), across multiple machines. It achieves reliability by [replicating](http://en.wikipedia.org/wiki/Replication_(computer_science)) the data across multiple hosts, and hence does not require [RAID](http://en.wikipedia.org/wiki/RAID) storage on hosts. With the default replication value, 3, data is stored on three nodes: two on the same rack, and one on a different rack. Data nodes can talk to each other to rebalance data, to move copies around, and to keep the replication of data high. HDFS is not fully [POSIX](http://en.wikipedia.org/wiki/POSIX) compliant, because the requirements for a POSIX file system differ from the target goals for a Hadoop application. The tradeoff of not having a fully POSIX-compliant file system is increased performance for data [throughput](http://en.wikipedia.org/wiki/Throughput) and support for non-POSIX operations such as Append.[[16]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-openlibrary1-16) HDFS was designed to handle very large files.[[15]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-hadoop.apache.org-15)

HDFS added high-availability capabilities, as announded for release 2.0 in May 2012[[17]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-failover-17) allowing the main metadata server (the NameNode) to be failed over manually to a backup in the event of failure. Automatic fail-over is being developed as well. Additionally, the file system includes what is called a *secondary namenode,* which misleads some people into thinking that when the primary namenode goes offline, the secondary namenode takes over. In fact, the secondary namenode regularly connects with the primary namenode and builds snapshots of the primary namenode's directory information, which is then saved to local or remote directories. These checkpointed images can be used to restart a failed primary namenode without having to replay the entire journal of file-system actions, then to edit the log to create an up-to-date directory structure. Because the namenode is the single point for storage and management of metadata, it can be a bottleneck for supporting a huge number of files, especially a large number of small files. HDFS Federation is a new addition that aims to tackle this problem to a certain extent by allowing multiple name spaces served by separate namenodes.

An advantage of using HDFS is data awareness between the job tracker and task tracker. The job tracker schedules map or reduce jobs to task trackers with an awareness of the data location. An example of this would be if node A contained data (x,y,z) and node B contained data (a,b,c). Then the job tracker will schedule node B to perform map or reduce tasks on (a,b,c) and node A would be scheduled to perform map or reduce tasks on (x,y,z). This reduces the amount of traffic that goes over the network and prevents unnecessary data transfer. When Hadoop is used with other file systems this advantage is not always available. This can have a significant impact on job-completion times, which has been demonstrated when running data-intensive jobs.[[18]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-18)

HDFS was designed for mostly immutable files[[16]](http://en.wikipedia.org/wiki/Apache_Hadoop" \l "cite_note-openlibrary1-16) and may not be suitable for systems requiring concurrent write operations.

Another limitation of HDFS is that it cannot be [mounted](http://en.wikipedia.org/wiki/Mount_(computing)) directly by an existing operating system. Getting data into and out of the HDFS file system, an action that often needs to be performed before and after executing a job, can be inconvenient. A [Filesystem in Userspace](http://en.wikipedia.org/wiki/Filesystem_in_Userspace) (FUSE) [virtual file system](http://en.wikipedia.org/wiki/Virtual_file_system) has been developed to address this problem, at least for [Linux](http://en.wikipedia.org/wiki/Linux) and some other [Unix](http://en.wikipedia.org/wiki/Unix) systems.

File access can be achieved through the native Java [API](http://en.wikipedia.org/wiki/API), the [Thrift](http://en.wikipedia.org/wiki/Thrift_(protocol)) API to generate a client in the language of the users' choosing (C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, Smalltalk, and OCaml), the [command-line interface](http://en.wikipedia.org/wiki/Command-line_interface), or browsed through the HDFS-UI [webapp](http://en.wikipedia.org/wiki/Web_application) over [HTTP](http://en.wikipedia.org/wiki/HTTP).