IBM®

IBM Software > Information Management > InfoSphere Platform > Big Data >

Hadoop

Welcome | What is Hadoop? | Products | Trials | Tutorials | Developers

Hadoop Technology | IBM Hadoop Technology | **Hadoop Glossary**

What is the Hadoop Distributed File System (HDFS)?

About Hadoop Distributed File System (HDFS)™

To understand how it's possible to scale a Hadoop® cluster to hundreds (and even thousands) of nodes, you have to start with the **Hadoop Distributed File System (HDFS)**. Data in a Hadoop cluster is broken down into smaller pieces (called blocks) and distributed throughout the cluster. In this way, the map and reduce functions can be executed on smaller subsets of your larger data sets, and this provides the scalability that is needed for big data processing.

ebook: Understanding Big Data Beyond the Hype

Stay on top of all the changes including, Hadoop-based analytics, streaming analytics, warehousing (including BigSQL), data asset discovery, integration, and governance

→ Read the ebook

What's the goal?

The goal of Hadoop is to use commonly available servers in a very large cluster, where each server has a set of inexpensive internal disk drives. For higher performance, MapReduce tries to assign workloads to these servers where the data to be processed is stored. This is known as data locality. (It's because of this principle that using a storage area network (SAN), or network attached storage (NAS), in a Hadoop environment is not recommended. For Hadoop deployments using a SAN or NAS, the extra network communication overhead can cause performance bottlenecks, especially for larger clusters.) Now take a moment and think of a 1000-machine cluster, where each machine has three internal disk drives; then consider the failure rate of a cluster composed of 3000 inexpensive drives + 1000 inexpensive servers!

We're likely already on the same page here: The component mean time to failure (MTTF) you're going to experience in a Hadoop cluster is likely analogous to a zipper on your kid's jacket: it's going to fail (and poetically enough, zippers seem to fail only when you really need them). The cool thing about Hadoop is that the reality of the MTTF rates associated with inexpensive hardware is actually well understood (a design point if you will), and part of the strength of Hadoop is that it has built-in fault tolerance and fault compensation capabilities. This is the same for HDFS, in that data is divided into blocks, and copies of these blocks are stored on other servers in the Hadoop cluster. That is, an individual file is actually stored as smaller blocks that are replicated across multiple servers in the entire cluster.

An example of HDFS

Think of a file that contains the phone numbers for everyone in the United States; the people with a last name starting with A might be stored on server 1, B on server 2, and so on. In a Hadoop world, pieces of this phonebook would be stored across the cluster, and to reconstruct the entire phonebook, your program would need the blocks from every server in the cluster. To achieve availability as components fail, HDFS replicates these smaller pieces onto two additional servers by default. (This redundancy can be increased or decreased on a per-file basis or for a whole environment; for example, a development Hadoop cluster typically doesn't need any data re-

Contact IBM

Considering a purchase?

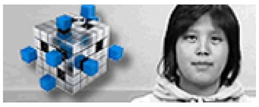
- Chat now
- Email IBM
- Request a quote
- ☎ Or call us at: 1-877-426-3774 Priority code: Info Mgmt

What is HDFS?

HDFS defined in 3 minutes with Rafael Coss, manager Big Data Enablement for IBM.

Watch the video (00:02:51)

Big Data - HBase Fundamentals



Learn the basics of HBase with Tina Chen, a solution architect in IBM's Big Data team.

Watch the video (00:03:00)

dundancy.) This redundancy offers multiple benefits, the most obvious being higher availability.

In addition, this redundancy allows the Hadoop cluster to break work up into smaller chunks and run those jobs on all the servers in the cluster for better scalability. Finally, you get the benefit of data locality, which is critical when working with large data sets. We detail these important benefits later in this chapter.

Additional Big Data Resources

→ Big data platform
→ Big data products
→ Stream computing