**Introduction to Big Data**

Big data is a phrase(like software development or testing) which describes about capturing, storing, querying, updating and analysing of huge data sets that are so voluminous and complex that traditional [data-processing](https://en.wikipedia.org/wiki/Data_processing) [application software](https://en.wikipedia.org/wiki/Application_software) are inadequate to deal with them.

**Big data challenges:**

[capturing data](https://en.wikipedia.org/wiki/Automatic_identification_and_data_capture), [data storage](https://en.wikipedia.org/wiki/Computer_data_storage), [data analysis](https://en.wikipedia.org/wiki/Data_analysis), Search, [sharing](https://en.wikipedia.org/wiki/Data_sharing), [transfer](https://en.wikipedia.org/wiki/Data_transmission), [visualization](https://en.wikipedia.org/wiki/Data_visualization), [querying,](https://en.wikipedia.org/wiki/Query_language) updating, [information privacy](https://en.wikipedia.org/wiki/Information_privacy) and data source.

**Data Sources:**

**1.Employees**

Text internal to your company: Think of all the text within documents, logs, survey results, and e-mails. Enterprise information represents a large percent of the text information in the world today.

**2. Users**

Social media data: This data is generated from the social media platforms such as YouTube, Facebook, Twitter, LinkedIn, and Flickr.

Mobile data: This includes data such as text messages and location information.

website content: This comes from any site delivering unstructured content, like YouTube, Flickr, or Instagram.

**3. Machines:**

Satellite images: This includes weather data or the data that the government captures in its satellite surveillance imagery. Just think about Google Earth, and you get the picture.

Scientific data: This includes seismic imagery, atmospheric data, and high energy physics.

Photographs and video: This includes security, surveillance, and traffic video.

Radar or sonar data: This includes vehicular, meteorological, and oceanographic seismic profiles.

**Size Hierarchy:**

|  |  |  |
| --- | --- | --- |
| **Unit** | **Value** | **Size** |
| bit**(b)** | **0 or 1** | **1/8 of a byte** |
| byte**(B)** | **8 bits** | **1 byte** |
| kilobyte**(KB)** | **10001 bytes** | **1,000 bytes** |
| megabyte**(MB)** | **10002 bytes** | **1,000,000 bytes** |
| gigabyte**(GB)** | **10003 bytes** | **1,000,000,000 bytes** |
| terabyte**(TB)** | **10004 bytes** | **1,000,000,000,000 bytes** |
| petabyte**(PB)** | **10005 bytes** | **1,000,000,000,000,000 bytes** |
| exabyte**(EB)** | **10006 bytes** | **1,000,000,000,000,000,000 bytes** |
| zettabyte**(ZB)** | **10007 bytes** | **1,000,000,000,000,000,000,000 bytes** |
| yottabyte**(YB)** | **10008 bytes** | **1,000,000,000,000,000,000,000,000 bytes** |

**Categories of Big Data:**

1. Structured
2. Semi-structured
3. Un-structured

**Structured Data**

It concerns all data which can be stored in database SQL in table with rows and columns. They have relational key and can be easily mapped into pre-designed fields. Today, those data are the most processed in development and the simplest way to manage information.

***Structured data represent only 1 to 5% of all data.***

**Semi structured data**

Semi-structured data is information that doesn’t reside in a relational database but that does have some organizational properties that make it easier to analyse. With some process, you can store them in relation database (it could be very hard for somme kind of semi structured data), but the semi structure exists to ease space, clarity or compute…

Examples of semi-structured: CSV, XML and JSON documents are semi structured documents, NoSQL databases are considered as semi structured.

***Semi-Structured data represent only 5 to 10% of all data.***

**Unstructured data**

Unstructured data represent around 80% of data. It often includes text and multimedia content. Examples include e-mail messages, word processing documents, videos, photos, audio files, presentations, webpages and many other kinds of business documents. Note that while these sorts of files may have an internal structure, they are still considered « unstructured » because the data they contain doesn’t fit neatly in a database.

Unstructured data is everywhere. In fact, most individuals and organizations conduct their lives around unstructured data. Just as with structured data, unstructured data is either machine generated or human generated.

**Characteristics of Big Data (4Vs):**

**How are predicting a data set or collection of data sets as ‘Big Data’**

1. **Volume**:

The name 'Big Data' itself is related to a size which is huge or enormous in nature. Size of data plays very crucial role in determining value out of data. Also, whether a data can be considered as a Big Data or not, is dependent upon volume of data. Hence, **'Volume'** is one characteristic which needs to be considered while dealing with 'Big Data'.

1. **Variety**:

Variety refers to heterogeneous(mixed) nature of data, both structured and unstructured. During earlier days, excels and databases were the only sources of data considered by most of the applications. Now days, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. is also being considered in the analysis applications. This variety of unstructured data poses certain issues for storage, mining and analysing data.

1. **Velocity**:

Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks and social media sites, sensors,[Mobile](https://www.guru99.com/mobile-testing.html)devices, etc. The flow of data is massive and continuous.

1. **Veracity**:

Big Data Veracity refers to the biases, noise and abnormality in data. Is the data that is being stored, and mined meaningful to the problem being analysed. In scoping out your big data strategy you need to have your team and partners work to help keep your data clean and processes to keep ‘dirty data’ from accumulating in your systems

**Benefits of Big Data Processing and Real-time examples**

Ability to process 'Big Data' brings in multiple benefits, such as-

**• Businesses can utilize outside intelligence while taking decisions**

Access to social data from search engines and sites like Facebook and twitter are enabling organizations to fine tune their business strategies.

**• Improved customer service**

Traditional customer feedback systems are getting replaced by new systems designed with 'Big Data' technologies. In these new systems, Big Data and natural language processing technologies are being used to read and evaluate consumer responses.

**• Early identification of risk to the product/services, if any**

**• Better operational efficiency**

'Big Data' technologies can be used for creating staging area or landing zone for new data before identifying what data should be moved to the data warehouse. In addition, such integration of 'Big Data' technologies and data warehouse helps organization to offload infrequently accessed data.

**HADOOP (The Yellow Elephant)**

****

**What is Hadoop?**

Hadoop is an open source apache software stack that facilitates using a network of many computers to solve problems involving massive amount of data storage and its processing. It provides a software framework for distributed data storage (**HDFS**) and distributed processing of Big Data using MapReduce (**MR**) programming model.

**History of Hadoop:**

* Hadoop was created by Doug Cutting, the creator of Apache Lucene (In 1997), the widely used text search library.
* Hadoop has its origin in Apache Nutch, an open source web search engine, itself a part of the Lucene project.

****

* The Name Hadoop is a made-up name. Doug explained about, how the name derived,

“The name my kid gave a stuffed yellow elephant. Short, relatively easy to spell and pronounce, meaningless, and not used elsewhere: those are my naming criteria. Kids are good at generating such. Googol is a kid’s term”

* Doug joined hands with Mike Cafarella to build web crawler and ambitious goal was to develop a web search engine.
* In 2002 - Nutch was started but the architecture wouldn’t support the growing counts of web pages which was in billions at that time.
* In 2003 - Google published white paper on Google Distributed File System (GFS)
* In 2004 - based on GFS idea, an open source implementation - Nutch Distributed File System (NDFS) started
* In 2004 - Google again published another white paper on distributed data processing model which called MapReduce
* In 2005 – The team of Nutch had a working MapReduce implementation in Nutch with NDFS storage
* In 2006 February – Dough Cutting moved out from Nutch to form an independent sub-project of Lucene called “**Hadoop**”
* In 2006 last quarter – Dough joined “**Yahoo**!” which provided dough with huge hardware, software and man power to move hadoop to the next level
* In 2008 – Yahoo! Announced that it successfully implemented hadoop in its search engine project
* In 2008 – Hadoop was made its own top-level project at Apache and after that, so many organizations started using Hadoop such as Facebook, New York Times

**Features of Hadoop:**

1. **Open Source**

Apache Hadoop is an open source project. It means its code can be modified according to business requirements.

1. **Distributed Processing**

As data is stored in a distributed manner in HDFS across the cluster, data is processed in parallel on a cluster of nodes.

1. **Fault Tolerance**

Hadoop has built-in fault tolerance. If a program fails while retrieving a block of data in one node, this can be handled automatically by build-in hadoop algorithm and re-executing the same piece of code on another data node which is holding a replication of the same data block.

1. **Reliability**

Due to replication of data in the cluster, data is reliably stored on the cluster of machine despite machine failures. If your machine goes down, then also your data will be stored reliably due to this characteristic of Hadoop.

1. **High Availability**

Data is highly available and accessible despite hardware failure due to multiple copies of data. If a machine or few hardware crashes, then data will be accessed from another path.

1. **Scalability**

Hadoop is highly scalable in the way new hardware can be easily added to the nodes. This feature of Hadoop also provides horizontal scalability which means new nodes can be added on the fly without any downtime.

1. **Economic**

Apache Hadoop is not very expensive as it runs on a cluster of commodity hardware. We do not need any specialized machine for it. Hadoop also provides huge cost saving also as it is very easy to add more nodes on the fly here. So, if requirement increases, then you can increase nodes as well without any downtime and without requiring much of pre-planning.

1. **Easy to use**

No need of client to deal with distributed computing, the framework takes care of all the things. So, this feature of Hadoop is easy to use.

1. **Data Locality**

This is a unique feature of Hadoop that made it easily handle the Big Data. Hadoop works on data locality principle which states that move computation to data instead of data to computation. When a client submits the MapReduce algorithm, this algorithm is moved to data in the cluster rather than bringing data to the location where the algorithm is submitted and then processing it.

1. **Data Integrity**

Data integrity is achieved by comparing the checksum value of the data at the time of data movement between data nodes and clients.

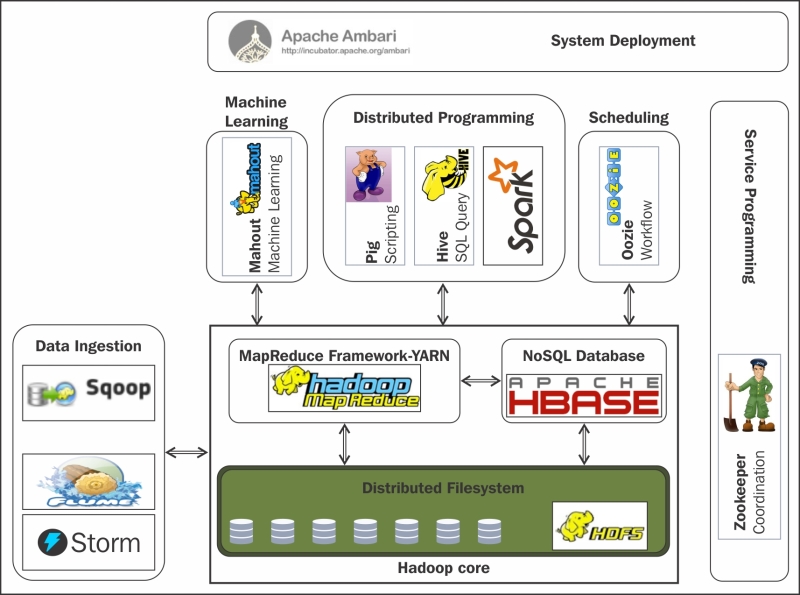
**Hadoop Ecosystems:**

Hadoop Ecosystem is neither a programming language nor a service, it is a platform or framework which solves big data problems. You can consider it as a suite which encompasses number of services (ingesting, storing, analysing and maintaining) inside it.

Refer the below link for complete set of hadoop ecosystems,

<https://hadoopecosystemtable.github.io/>

* [HDFS](https://www.edureka.co/blog/hadoop-ecosystem#hdfs) -> Hadoop Distributed File System
* [YARN](https://www.edureka.co/blog/hadoop-ecosystem#yarn)-> Yet Another Resource Negotiator
* [MapReduce](https://www.edureka.co/blog/hadoop-ecosystem#mapreduce)-> Data processing using programming
* [Spark](https://www.edureka.co/blog/hadoop-ecosystem#apache_spark)-> In-memory Data Processing
* [PIG](https://www.edureka.co/blog/hadoop-ecosystem#apache_pig), [HIVE](https://www.edureka.co/blog/hadoop-ecosystem#apache_hive)-> Data Processing Services using Query (SQL-like)
* [HBase](https://www.edureka.co/blog/hadoop-ecosystem#apache_hbase)-> NoSQL Database
* [Mahout](https://www.edureka.co/blog/hadoop-ecosystem#apache_mahout), Spark MLlib -> Machine Learning
* [Zookeeper](https://www.edureka.co/blog/hadoop-ecosystem#apache_zookeeper)-> Managing Cluster
* [Oozie](https://www.edureka.co/blog/hadoop-ecosystem#apache_oozie)-> Job Scheduling
* [Flume](https://www.edureka.co/blog/hadoop-ecosystem#apache_flume), [Sqoop](https://www.edureka.co/blog/hadoop-ecosystem#apache_sqoop), Nifi, Kafka -> Data Ingesting Services
* [Ambari](https://www.edureka.co/blog/hadoop-ecosystem#apache_ambari)-> Provision, Monitor and Maintain cluster

****

**When to use Hadoop?**

Hadoop is used for:

* Search – Yahoo, Google, Amazon
* Log processing – Facebook, Yahoo
* Data Warehouse – Facebook
* Video and Image Analysis – New York Times

Till now, we have seen how Hadoop has made Big Data handling possible. But there are some scenarios where Hadoop implementation is not recommended.

**When not to use Hadoop?**

Following are some of those scenarios:

* Low Latency data access: Quick access to small parts of data
* Multiple data modification: Hadoop is a better fit only if we are primarily concerned about reading data and not modifying data.
* Lots of small files: Hadoop is suitable for scenarios, where we have few but large files.

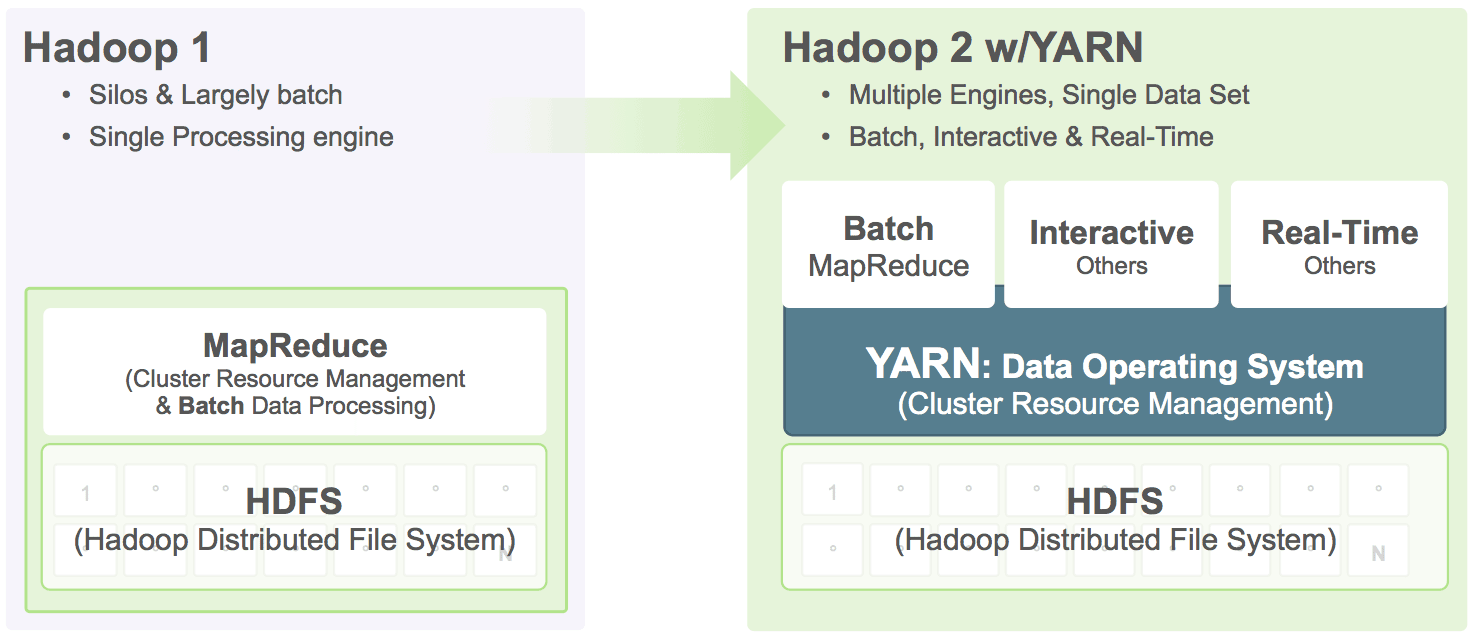
**Components of Hadoop:**

In the first version of Hadoop (Hadoop 1.0), the core components included Hadoop Common, HDFS, and MapReduce, but the second version of Hadoop came out with a new technology called YARN which was an acronym for **Yet Another Resource Negotiator (YARN)**.

* **Hadoop Common**: Hadoop Common refers to the collection of common utilities and libraries that support other Hadoop modules. It is an essential part or module of the Apache Hadoop Framework, along with the Hadoop Distributed File System (HDFS), Hadoop YARN and Hadoop MapReduce. Like all other modules, Hadoop Common assumes that hardware failures are common and that these should be automatically handled in software by the Hadoop Framework. Hadoop Common is also known as **Hadoop Core**.
* **Hadoop MapReduce**: MapReduce is a computational model and software framework for writing applications which are run on Hadoop. These MapReduce programs are capable of processing enormous data in parallel on large clusters of computation nodes.
* **HDFS (Hadoop Distributed File System):** HDFS takes care of storage part of Hadoop applications. MapReduce applications consume data from HDFS. HDFS creates multiple replicas of data blocks and distributes them on compute nodes in cluster. This distribution enables reliable and extremely rapid computations.
* **YARN** **(Yet Another Resource Negotiator)**: The technology became an Apache Hadoop subproject within the Apache Software Foundation (ASF) in 2012 and was one of the key features added in Hadoop 2.0, which was released for testing that year and became generally available in October 2013.

The addition of YARN significantly expanded Hadoop's potential uses. The original incarnation of Hadoop closely paired the Hadoop Distributed File System (HDFS) with the batch-oriented MapReduce programming framework and processing engine, which also functioned as the big data platform's resource manager and job scheduler. As a result, Hadoop 1.0 systems could only run MapReduce applications -- a limitation that Hadoop YARN eliminated.

Before getting its official name, YARN was informally called MapReduce 2 or NextGen MapReduce. But it introduced a new approach that decoupled cluster resource management and scheduling from MapReduce's data processing component, enabling Hadoop to support varied types of processing and a broader array of applications. For example, Hadoop clusters can now run interactive querying, streaming data and real-time analytics applications on Apache Spark and other processing engines simultaneously with MapReduce batch jobs.

****

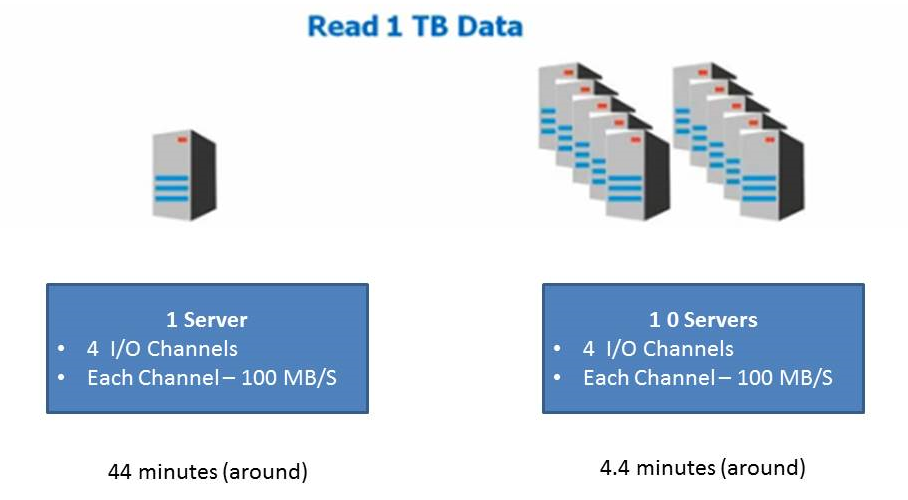
**HDFS (Hadoop Distributed File System)**

The Hadoop Distributed File System (HDFS) is the primary data storage system used by Hadoop applications. It employs a NameNode and DataNode architecture to implement a distributed file system that provides high-performance access to data across highly scalable Hadoop clusters.

HDFS creates multiple replicas of each data block and distributes them on throughout a cluster of computer nodes to enable resilient (strong and robust) access hence improved fault tolerance.

**Why HDFS?**

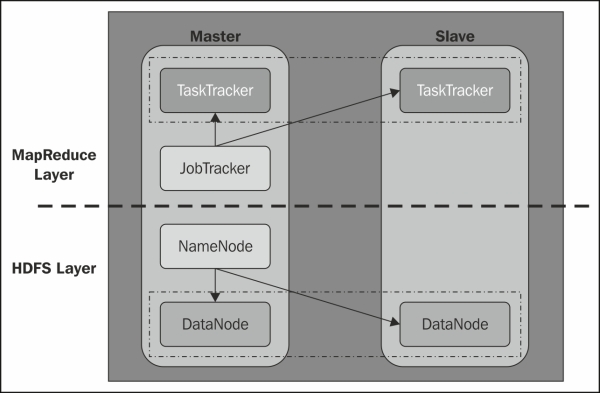
When HDFS takes in data, it breaks the information down into separate blocks and distributes them to different nodes in a cluster, thus enabling highly efficient parallel processing. Moreover, the Hadoop Distributed File System is specially designed to be highly fault-tolerant.



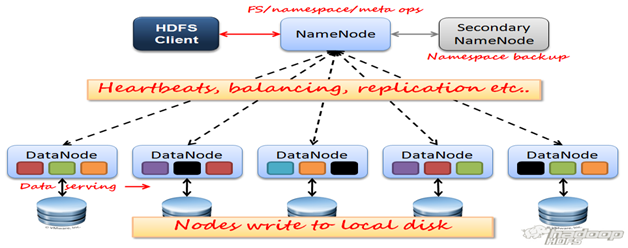
**Data Visibility and Accessibility:**

**For Example:** When you create folder in HDFS, the created folder will be displayed throughout the cluster as a single file system unlike creating directories in each node separately.

**Hadoop Architecture in High Level**



**HDFS Architecture**

****

In the above diagram, there is one Name Node and multiple Data Nodes (servers) with data blocks.

When you dump a file (or data) into the HDFS, it stores them in blocks on the various nodes in Hadoop Clusters. HDFS creates several replications of the data blocks and distribute them accordingly in the cluster is a way that will be reliable and retrieved faster.

Hadoop will internally make sure that any node failure will never result in data loss. There will be only one machine that manages the file system meta-data. There will be multiple data nodes (These are the real cheap commodity servers that will store data blocks). When we execute a query from a client, it will reach out to Name Node to get the file system meta-data information, and then it will reach out to the Data Node to get the real data blocks.

**Major Components of HDFS:**

* Name Node (Master)
* Secondary Name Node
* Data Node (Slave)

1. **Name Node:**

HDFS works by breaking large files into smaller pieces called blocks. The blocks are stored on data nodes, and it is the responsibility of the NameNode to know what blocks on which data nodes make up the complete file.

The complete collection of all the files in the cluster is referred to as the file system namespace. The NameNode manages the filesystem namespace. It maintains the filesystem tree and the metadata for all the files and directories in the tree. So, it contains the information of all the files, directories and their hierarchy in the cluster. Along with the filesystem information it also knows about the DataNode on which all the blocks of a file kept.

It is the Name Node’s job to oversee the health of Data Nodes and to coordinate access to data. The Name Node is the central controller of HDFS.

A client accesses the filesystem on behalf of the user by communicating with the NameNode and DataNode. The client presents a filesystem interface like a Portable Operating System Interface (POSIX), so the user code does not need to know about the NameNode and DataNode to function.

* 1. **Roles of Name Node:**

1. Name Node holds the **Metadata** (data about data) for HDFS in-memory and in-disk.
2. In-memory metadata help to achieve instance/speed access to the data.
3. Controls read/write access to files, check the existence of the directory/files before read/write.
4. Manages blocks, replications and re-replication process.
5. Metadata files (fsimage & editlog) serves only when the cluster is restarted due to failure which will be referring these files
   1. **Metadata Components:**

**Fsimage:** Stores the inode details like modification time, access time, access permission, replication.

**Editlog:** This keeps tracking of each change that is being done on HDFS. (Like adding a new file, deleting an existing file, moving/coping files between folders. Etc)