Big Data : Evolution, Concepts, and Core Technologies

Pratheek Patel B M - 4NICS22156

Rakshith Gowda H - 4NI22CS170

Sanketh N R - 4NI22CS192

Sagar J R - 4NI22CS184

The Big Data Explosion: Why Traditional Systems Failed

The digital age has brought an unprecedented surge in data creation. By 2025, global data is projected to exceed a staggering 180 zettabytes annually, a volume that traditional relational database management systems were simply not built to handle.

This explosion of data, often referred to as "Big Data," is characterized by the **5 Vs:**

- **Volume:** The sheer quantity of data generated.
- **Velocity:** The speed at which data is generated and processed.
- **Variety:** The diverse types of data (structured, unstructured, semi-structured).
- **Veracity:** The quality and accuracy of the data.
- **Value:** The potential insights and benefits derived from the data.

The need for scalable, fault-tolerant, and distributed systems to process these massive and diverse datasets became critically apparent.

Hadoop Origins: From Nutch to a Big Data Pioneer

The story of Hadoop begins in **2002** with **Doug Cutting** and **Mike Cafarella**, who embarked on the **Nutch** project, an ambitious open-source web search engine.

1

2002: Nutch Genesis

Doug Cutting and Mike Cafarella initiate Nutch, an open-source web search engine, laying the groundwork for distributed data processing.

2

2005: Hadoop Emerges

Inspired by Google's groundbreaking papers on Google File System and MapReduce, Hadoop is spun off from Nutch to handle large-scale data processing.

3

Name & Adoption

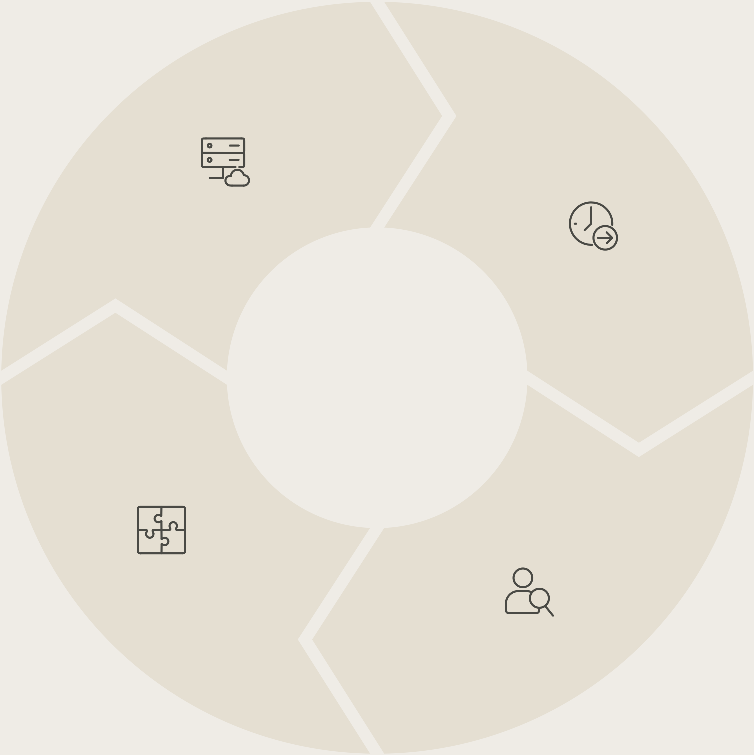
Named after Cutting's son's toy elephant, Hadoop quickly gains traction, becoming an Apache project in 2008. Yahoo plays a crucial role in its early development and widespread adoption.

4

A New Era

Hadoop establishes itself as the leading platform for processing and storing Big Data, changing the landscape of data analytics.

Hadoop’s Core Architecture: Distributed Storage and Processing



HDFS

Stores huge datasets by splitting files into blocks across commodity hardware nodes, ensuring reliability and throughput.

MapReduce

A programming model that processes data in parallel, mapping tasks and reducing results for efficient batch processing.

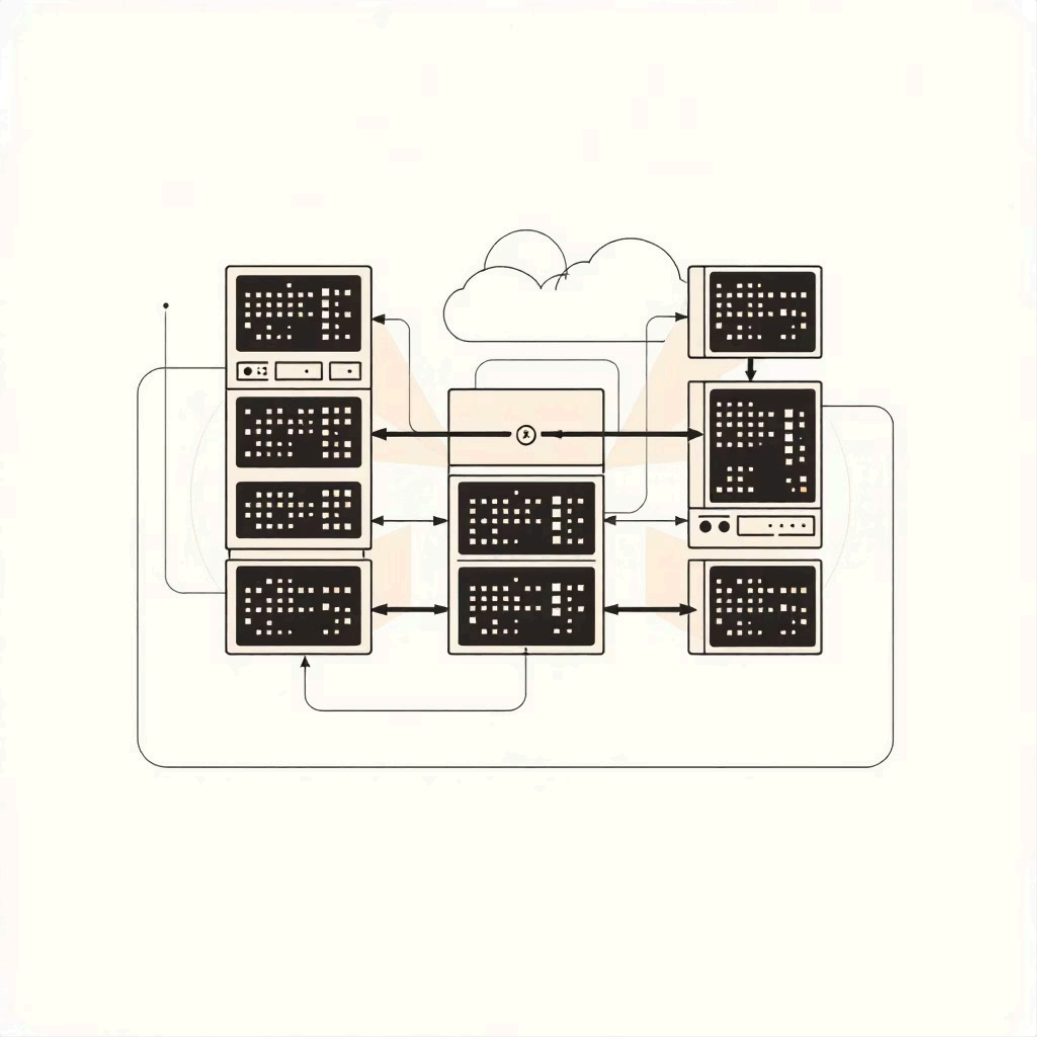
YARN

Manages cluster resources and schedules jobs, enabling multiple processing engines to coexist and utilize resources efficiently.

Hadoop Common

Provides shared utilities and libraries that support the other core components, forming the foundational layer of the Hadoop ecosystem.

HDFS: The Backbone of Hadoop Storage



The **Hadoop Distributed File System (HDFS)** is the primary storage component of Hadoop, fundamentally altering how large files are stored and accessed. Its design principles are rooted in two key concepts:

- **Fault Tolerance:** Data is automatically replicated across multiple machines, ensuring that the system can recover from hardware failures without data loss.
- **High Throughput:** Optimized for batch processing over large datasets, allowing for high data transfer rates, crucial for big data analytics.

HDFS operates with two main types of nodes:

- **NameNode:** The master server that manages the file system metadata, such as file names, directories, and block locations.
- **DataNodes:** The worker nodes that store the actual data blocks, typically in sizes of 128MB or more, and handle read/write requests from clients.

This architecture enables scalable, cost-effective storage solutions leveraging commodity hardware, making Big Data accessible for many organizations.

MapReduce: Parallel Data Processing Engine

MapReduce is a powerful programming model and an associated implementation for processing large data sets with a parallel, distributed algorithm on a cluster. It was introduced by Google in 2004 and quickly adopted by Hadoop to become its primary batch processing engine.

Input Data

Raw, unstructured or semi-structured data is fed into the MapReduce framework.

Map Phase

Input data is split and processed by 'mapper' functions. Each mapper filters and sorts data, transforming it into key-value pairs.

Shuffle & Sort

Intermediate key-value pairs are grouped by key and sorted, preparing them for the reduce phase.

Reduce Phase

'Reducer' functions aggregate and summarize the mapped data, generating the final output.

Output

The aggregated results are stored back into HDFS or another designated storage system.

This two-phase process allows for the processing of petabytes of data across thousands of nodes simultaneously, making it ideal for tasks like log analysis, web indexing, and data warehousing.

YARN: The Resource Manager Revolution

YARN (Yet Another Resource Negotiator) was introduced in Hadoop 2.0 as a significant architectural shift. It addressed limitations of earlier Hadoop versions, where MapReduce was the sole processing engine, leading to resource contention and inflexibility.

YARN revolutionized Hadoop by separating the cluster's **resource management** capabilities from the **data processing** functionalities. This separation allows Hadoop clusters to support diverse data processing engines beyond just MapReduce, such as Spark, Tez, and Flink, to run on the same cluster.

Key Components of YARN:

- **ResourceManager:** The global arbitrator of resources, responsible for allocating compute resources to applications.
- **NodeManager:** Runs on each node in the cluster, responsible for launching and monitoring containers (resource allocations) and reporting resource usage to the ResourceManager.
- **ApplicationMaster:** Manages individual applications, negotiating resources from the ResourceManager and working with NodeManager to execute tasks.

This architecture greatly improves cluster utilization, supports real-time and interactive workloads, and enhances the overall efficiency and versatility of the Hadoop ecosystem.

Hive: SQL for Big Data

Apache Hive, developed by Facebook and open-sourced in **2008**, provides a data warehousing solution built on top of Hadoop. Its primary goal is to enable analysts and data scientists to query large datasets stored in HDFS using a familiar SQL-like language.

Key Features:

- **HiveQL:** A SQL-like query language that abstracts away the complexities of MapReduce programming, allowing users to write standard SQL queries.
- **Data Abstraction:** Hive imposes a schema on data stored in HDFS, treating them as tables, even if the underlying data is unstructured.
- **Query Translation:** When a HiveQL query is executed, Hive translates it into a series of MapReduce or Tez jobs that run on the Hadoop cluster.
- **Batch Processing:** Primarily designed for batch processing and analytics rather than real-time querying.

Hive is widely used for data summarization, ad-hoc queries, and analysis of large datasets, making Big Data more accessible to a broader range of users.

The Expanding Hadoop Ecosystem

Hadoop evolved into a vibrant ecosystem with numerous complementary projects, each addressing specific Big Data challenges. This expansion enabled Hadoop to support a wide range of diverse workloads, extending beyond simple batch processing.

Apache Pig

A high-level platform for creating MapReduce programs, simplifying scripting for data analysis.

Apache HBase

A NoSQL, column-oriented database providing real-time read/write access to Big Data in Hadoop.

Apache Spark

A lightning-fast unified analytics engine for large-scale data processing, offering in-memory computation for faster performance.

Apache Oozie

A workflow scheduler system to manage and orchestrate Hadoop jobs, defining a sequence of actions.

Apache Kafka

A distributed streaming platform for building real-time data pipelines and streaming applications.

Companies like Netflix, Facebook, Twitter, and numerous financial institutions have adopted various components of the Hadoop ecosystem to build scalable and robust data solutions for everything from batch processing and real-time analytics to machine learning and search indexing.

Conclusion: Hadoop's Legacy and the Future of Big Data

Hadoop undeniably transformed the landscape of data storage and processing by introducing a distributed, scalable, and fault-tolerant architecture. Its core components—HDFS for reliable storage, MapReduce for parallel processing, YARN for efficient resource management, and Hive for SQL-based queries—remain foundational.

While the Big Data ecosystem continues to evolve, with new technologies and cloud-native solutions emerging, the principles pioneered by Hadoop are more relevant than ever. Modern Big Data platforms often build upon Hadoop's concepts, integrating advancements in real-time processing, machine learning, and cloud infrastructure.

Embracing the Hadoop ecosystem, or its spiritual successors, is crucial for organizations looking to unlock valuable insights from the ever-growing volumes of data in our interconnected world.