Topic 3: Hadoop Ecosystem

1. What is Hadoop?

Official Definition

- **Hadoop**: A project that develops **open-source software** for reliable, scalable, distributed computing
- Not a product - it's a **framework and ecosystem**

Core Philosophy

- Run on **commodity hardware** (not high-performance computers)
- **Inexpensive** to deploy and scale
- Can be deployed **on premises or in the cloud**

2. Key Features of Hadoop

Storage and Processing Capabilities

1. **Large data set handling**
    - File sizes: Gigabytes to terabytes
    - Can store **millions of files**
    - Simple scalability and coordination
2. **Fault Tolerance**
    - High fault tolerance through **data replication**
    - Automatic recovery from failures
    - No single point of failure
3. **Data Access Patterns**
    - **Streaming access** to data
    - **Write-once-read-many (WORM)** consistency model
    - Optimized for batch processing
    - Supports interactive, iterative, and stream processing
4. **Cost and Deployment**
    - Runs on commodity hardware
    - Inexpensive compared to traditional solutions
    - Flexible deployment options (on-premises, cloud)

3. Core Components of Hadoop

The Two Pillars

1. **HDFS (Hadoop Distributed File System)**

    - **Storage layer** of Hadoop
    - Distributed file system across cluster
    - Handles data replication and fault tolerance

2. **YARN (Yet Another Resource Negotiator)**

    - **Resource management layer**
    - Coordinates computing resources in cluster
    - Schedules and manages applications

4. Hadoop Ecosystem Components

Storage and Management

- **HDFS**: Distributed file system
- **HBase**: NoSQL database on HDFS
- **ZooKeeper**: Coordination service

Processing Frameworks

- **MapReduce**: Original batch processing framework
- **Spark**: Fast, in-memory processing engine
- **Tez**: DAG-based processing framework

Data Access and Query

- **Hive**: SQL-like query language (HQL)
- **Pig**: High-level data flow language
- **Impala**: Real-time SQL queries

Data Integration

- **Sqoop**: Transfer data between Hadoop and relational databases
- **Flume**: Collect and aggregate streaming data
- **Kafka**: Distributed streaming platform

Workflow and Coordination

- **Oozie**: Workflow scheduler
- **ZooKeeper**: Distributed coordination

5. Commercial Hadoop Landscape

Major Distributions

- **Cloudera**: Cloudera Data Platform (CDP)
- **Hortonworks**: Hortonworks Data Platform (HDP) - now merged with Cloudera
- **MapR**: MapR Data Platform
- **Amazon EMR**: Elastic MapReduce (cloud-based)
- **Microsoft HDInsight**: Azure-based Hadoop

- **IBM BigInsights**: IBM's Hadoop distribution

6. Hadoop Cluster Architecture

Master-Slave Architecture

Master Nodes (control plane):

- **NameNode** (HDFS master)
    - Manages filesystem metadata
    - Tracks file locations and block information
    - Does NOT store actual file data
- **ResourceManager** (YARN master)
    - Allocates cluster resources
    - Schedules applications
    - Monitors cluster health

Slave/Worker Nodes (data plane):

- **DataNode** (HDFS slave)
    - Stores actual data blocks
    - Handles read/write operations
    - Reports to NameNode via heartbeats
- **NodeManager** (YARN slave)
    - Launches and monitors containers
    - Manages resources on each node
    - Reports to ResourceManager

Additional Components:

- **SecondaryNameNode**: Checkpoints NameNode metadata (backup)
- **Standby NameNode**: High availability (optional)
- **Edge Servers**: Access points for launching applications
- **Relational Databases**: Store metadata (e.g., MySQL for Hive metastore)

7. Hadoop Cluster Specifications

Scale

- Clusters can support **up to 10,000 servers**
- Near-linear scalability in computing power
- Production clusters typically 100-1000 nodes

Typical Cluster Composition

1. **Master nodes**: Run key Hadoop framework daemons

2. **Worker nodes**: Host storage (HDFS) and computing (YARN) work
3. **Edge servers**: Used for accessing cluster and launching applications
4. **Database servers**: Store metadata repositories
5. **Dedicated framework servers**: Special frameworks (e.g., Kafka)

8. Hadoop Deployment Modes

1. Standalone Mode

- Single JVM, no distributed processing
- Used for debugging

2. Pseudo-Distributed Mode

- **All daemons run on a single node**
- Highly simulates full cluster
- Easy for beginner's practice
- Easy for testing and debugging
- **Lab setting uses this mode**: Ubuntu 14.04 Virtual Machine

3. Fully Distributed Mode

- **Production mode**
- Daemons distributed across multiple nodes
- True cluster configuration

Key Points for Exam

Critical Concepts

1. **Hadoop = HDFS + YARN + Processing Framework**
2. **Master-Slave Architecture**: Core organizational principle
3. **Commodity Hardware**: Cost-effective approach
4. **Write-Once-Read-Many (WORM)**: Data consistency model
5. **Data Locality**: Move code to data, not data to code

Core Components Must Know

- **HDFS**: Storage layer
- **YARN**: Resource management
- **MapReduce**: Original processing framework
- **Hive**: SQL on Hadoop
- **HBase**: NoSQL database
- **Spark**: Fast in-memory processing

Important Daemons

HDFS:

- NameNode (master)
- DataNode (slave)
- SecondaryNameNode (backup)

YARN:

- ResourceManager (master)
- NodeManager (slave)
- ApplicationMaster (per-application)

Key Features to Remember

1. **High fault tolerance**: Data replication
2. **Scalability**: Linear scaling up to 10,000 nodes
3. **Cost-effective**: Commodity hardware
4. **Flexible**: Supports multiple processing paradigms
5. **Data replication**: Default 3 copies

Deployment Modes

- **Standalone**: Single JVM, debugging
- **Pseudo-distributed**: Single machine, all daemons (labs)
- **Fully distributed**: Multiple machines, production

Why Hadoop for Big Data?

1. Cost-effective fault-tolerant storage (HDFS)
2. Scalability (horizontal)
3. Schema-on-read (interpret at runtime)
4. Low cost for unstructured/semi-structured data
5. Fast data ingestion
6. Separation of programming logic and scheduling
7. Multiple abstraction levels (MapReduce, Hive, Pig, Spark)
8. Multi-language support (Java, SQL, Scala, Python)