HDFS Architecture, Goals, Components & Configuration – Detailed Answers

1. Architecture of HDFS

HDFS (Hadoop Distributed File System) follows a master-slave architecture designed to store very large files across multiple machines in a distributed environment. It is optimized for high throughput and fault tolerance.

The master node in HDFS is called the NameNode. It manages the file system namespace, metadata, and controls access to files. It keeps track of where file data blocks are stored across the cluster.

The slave nodes are called DataNodes. They are responsible for storing actual data blocks and serving read/write requests from clients. Each file is split into fixed-size blocks and distributed across multiple DataNodes.

HDFS also includes a Secondary NameNode (or Standby NameNode in HA setups), which periodically merges metadata changes to prevent NameNode failure.

2. Goals of HDFS

HDFS is designed to meet specific goals required for big data processing and storage.

- Fault Tolerance – Automatically replicates data blocks to handle hardware failures.
- High Throughput – Optimized for large data access rather than low-latency access.
- Scalability – Can scale to thousands of nodes and petabytes of data.
- Data Locality – Moves computation closer to data to reduce network congestion.
- Reliability – Ensures data availability even during node failures.
- Cost Effectiveness – Runs on commodity hardware.

3. HDFS Components

- NameNode – Maintains metadata and file system hierarchy.
- DataNode – Stores actual data blocks.
- Secondary NameNode – Performs checkpointing of metadata.
- Standby NameNode – Provides high availability.
- JournalNode – Maintains edit logs in HA configuration.
- Zookeeper – Coordinates failover in HA clusters.

4. Hadoop Configuration Files

Hadoop uses XML configuration files to define cluster settings and behavior.

core-site.xml: Defines basic Hadoop system properties such as file system URI, temporary directories, and I/O settings.

hdfs-site.xml: Contains HDFS-specific configuration like replication factor, block size, NameNode and DataNode directories.

yarn-site.xml: Defines configuration for YARN resource management, including NodeManager and ResourceManager settings.

5. Function of SSH Keygen

SSH keygen is a utility used to generate secure SSH key pairs (public and private keys). In Hadoop clusters, SSH keys are used to enable password-less authentication between master and slave nodes.

Password-less SSH allows Hadoop daemons to start and manage services across the cluster without manual password entry. This improves security, automation, and cluster management efficiency.

The public key is shared with remote nodes, while the private key remains securely stored on the local machine.