**HDFS**

The **Hadoop** Distributed File System (**HDFS**) is a distributed file system designed to store and process large amounts of data in a distributed manner, allowing for parallel processing of data across multiple nodes. It is an open-source project under the Apache Foundation and is a key component of the Apache Hadoop ecosystem.

**Purpose:**

One of the main reasons we need HDFS is to handle big data workloads, where data sets are too large or complex to be processed by a single machine. Traditional file systems are not well-suited for handling such large data sets because they can become slow and cumbersome as the amount of data grows. HDFS, on the other hand, is designed to scale horizontally, allowing for the addition of more nodes to the cluster as the amount of data grows. This makes it possible to handle large data sets without the need for expensive, high-performance hardware.

Another reason HDFS is useful is its ability to handle large amounts of data in a fault-tolerant manner. HDFS is designed to be highly fault-tolerant, with features such as data replication and automatic recovery from node failures. This allows it to continue functioning even in the event of hardware failures or other disruptions. HDFS also ensures that data is always available and accessible, even in the event of node failures or other disruptions, through the use of replicas and the ability to automatically recover from failures.

**Main Components:**

The main components of HDFS are the Namenode, the Datanode, and the Client:

**1 - NameNode:** The NameNode is the central component of HDFS, responsible for managing the file system namespace and regulating access to files by client applications. It maintains a record of all the files and directories in the file system, as well as the blocks that make up each file and the locations of those blocks within the cluster. The NameNode is a single point of failure in an HDFS cluster, so it is important to have a backup NameNode in case of a failure.

**2 - DataNode:** DataNodes are the worker nodes in an HDFS cluster, responsible for storing and replicating the actual data blocks. They communicate with the NameNode to report on the blocks they are storing and to receive instructions on replication and block movement. HDFS stores each file as a sequence of blocks, and each block is replicated across multiple DataNodes to ensure fault tolerance.

**3 - Client:** Clients are the applications that interact with HDFS, either to read or write data. They communicate with the NameNode to request access to files and with the DataNodes to read or write the actual data blocks.

Overall, the architecture of HDFS is designed to provide a scalable, fault-tolerant, and highly available system for storing and processing large amounts of data in a distributed manner.

**Properties:**

**1 - Scalability:** HDFS is designed to scale horizontally, allowing for the addition of more nodes to the cluster as the amount of data grows. This allows HDFS to handle large data sets without the need for expensive, high-performance hardware.

**2 - Fault tolerance**: HDFS is designed to be highly fault-tolerant, with features such as data replication and automatic recovery from node failures. This allows it to continue functioning even in the event of hardware failures or other disruptions.

**3 - High availability**: HDFS is designed to ensure that data is always available and accessible, even in the event of node failures or other disruptions. It does this through the use of replicas and the ability to automatically recover from failures.

**4 - Data locality**: HDFS is designed to store data blocks as close as possible to the nodes that will be processing them, to minimize the amount of data transfer across the network. This can improve the performance of big data applications that rely on HDFS.

**5 - Data integrity**: HDFS includes checksumming to ensure the integrity of data blocks, allowing it to detect and repair corrupted data.

**6 - Data coherency**: HDFS ensures that changes to files are atomic and consistent, so that multiple clients can read and write to the same file without interference.

**Operation:**

**To operate on HDFS**, you can use the HDFS command-line interface or a programming API such as the Java API or the Hadoop Streaming API. Some common HDFS operations include creating and deleting files and directories, reading and writing data to files, and setting permissions on files and directories.

**Some examples of HDFS commands:**

**1- To create a new directory**:

This command creates a new directory at the specified path in HDFS. If the directory already exists, this command will fail.

"hdfs dfs -mkdir /newdir"

**2- To copy a file from the local file system to HDFS:**

This command copies a file from the local file system to HDFS.

hdfs dfs -put /local/path/to/file /hdfs/path/to/destination))

**3- To read the contents of a file:**

This command reads the contents of a file in HDFS and prints them to the console.

hdfs dfs -cat /hdfs/path/to/file))

**4- To delete a file:**

This command deletes a file from HDFS.

(hdfs dfs -rm /hdfs/path/to/file)

**5 - To list the contents of a directory:**

This command lists the contents of a directory in HDFS.

hdfs dfs -ls /hdfs/path/to/directory

**There are also some disadvantages to using HDFS**, such as a lack of support for certain types of workloads (such as low-latency, high-concurrency workloads) and the need for specialized knowledge and expertise to set up and maintain an HDFS cluster.

**References:**

Works Cited

*Apache Hadoop*,

<https://hadoop.apache.org>.

Accessed 24 December 2022.

“Hadoop - HDFS Operations.” *Tutorials point*, <https://www.tutorialspoint.com/hadoop/hadoop_hdfs_operations.htm>

Accessed 24 December 2022.

“Learn Big Data Hadoop - HDFS Overview.” *Tutorials point*, <https://www.tutorialspoint.com/hadoop/hadoop_hdfs_overview.htm>

. Accessed 24 December 2022.

Rosencrance, Linda. “What is HDFS? Hadoop Distributed File System overview.” *TechTarget*,

<https://www.techtarget.com/searchdatamanagement/definition/Hadoop-Distributed-File-System-HDFS>.

Accessed 24 December 2022