CSE 4/587
Data Intensive Computing

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Dr. Shamshad Parvin
shamsadp@buffalo.edu
313 Davis Hall

Hadoop Distributed File System (HDFS)

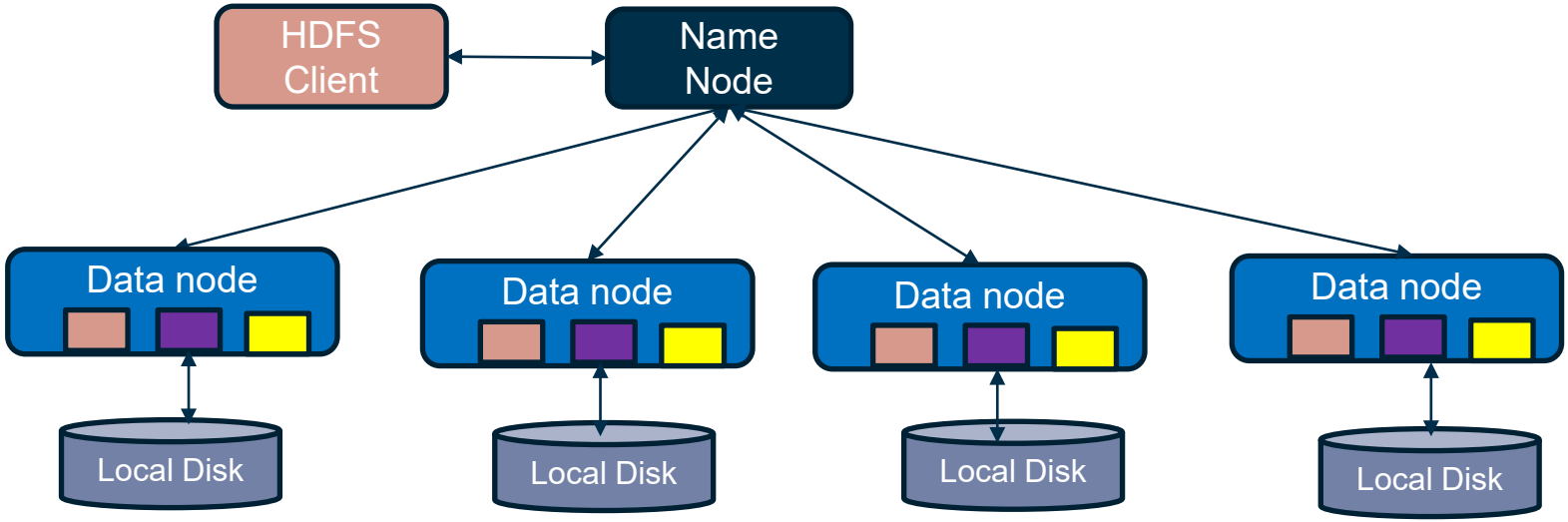
Recap from Last Class

- Hadoop Distributed File System (HDFS) is the open source version of the Google File System (GFS)
  - Allows reliable and efficient storage and access of large write-once, read-many (WORM) files
  - NameNode acts as a server that manages the filesystem
  - DataNodes store data blocks and serve read/write requests
  - Blocks are replicated to allow for fault tolerance and fast reads

Key Aspects

- The key aspects of Hadoop we will be discussed are:
  - Architecture
  - Robustness
  - Data Organization
  - Communications Protocol
  - API to access services
  - Evaluation of Hadoop
  - Hadoop Ecosystem
  - MapReduce

HDFS Architecture



HDFS Architecture

The NameNode does not directly call DataNodes. It uses replies to heartbeats to send instructions to the DataNodes. The instructions include commands to:

- replicate blocks to other nodes;
- remove local block replicas;
- re-register or to shut down the node;
- send an immediate block report

Robustness

Possible Failures

- The primary objective of HDFS is to store data reliably in the presence of failures
- Three common failures that it must handle are:
  - DataNode failure
  - NameNode failure
  - Network partition

DataNode Failure and Heartbeat

- A crashed DataNode or a network partition can cause a subset of DataNodes to lose connectivity with the NameNode
- NameNode detects this by the absence of a heartbeat
  - NameNode marks these DataNodes, and does not send requests to them
  - Data registered to the failed DataNode is not available to the HDFS
  - Death of a DataNode may cause some blocks to require more replication

Re-Replication

- Sometimes Blocks in the system may fall below the required replication factor
- This can occur for a number of reasons
  - A DataNode has become unavailable
  - A replica may become corrupted
  - A hard disk on a DataNode may fail
  - The replication factor may have been increased

Data Integrity

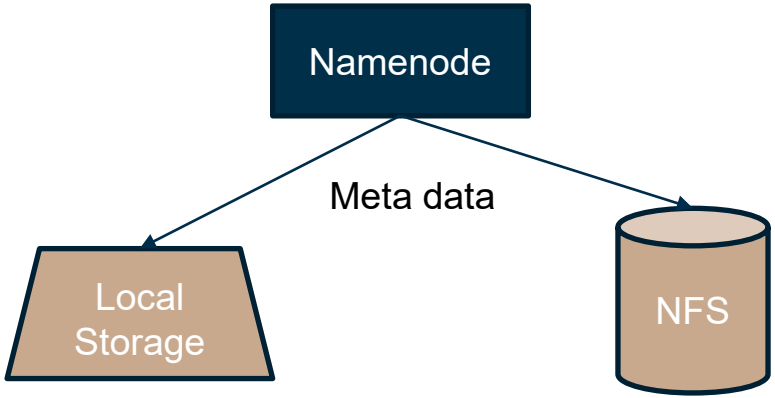
- What if a block of data fetched from a DataNode arrives corrupted
  - Fault in a storage device
  - Network faults
  - Buggy software
- An HDFS client creates a checksum for every block of its file and stores it in the HDFS namespace
- When the client retrieves the contents of a file it verifies that they match…if not it must retrieve the block from another replica

MetaData Disk Failure

- FsImage and EditLog are central data structures of HDFS
  - Corruption of these files can cause an entire HDFS instance to become non-functional.
- A NameNode can be configured to maintain multiple copies of these files
  - These copies are updated synchronously
  - MetaData is not data intensive
- The NameNode is a potential single point of failure
  - This currently requires manual intervention

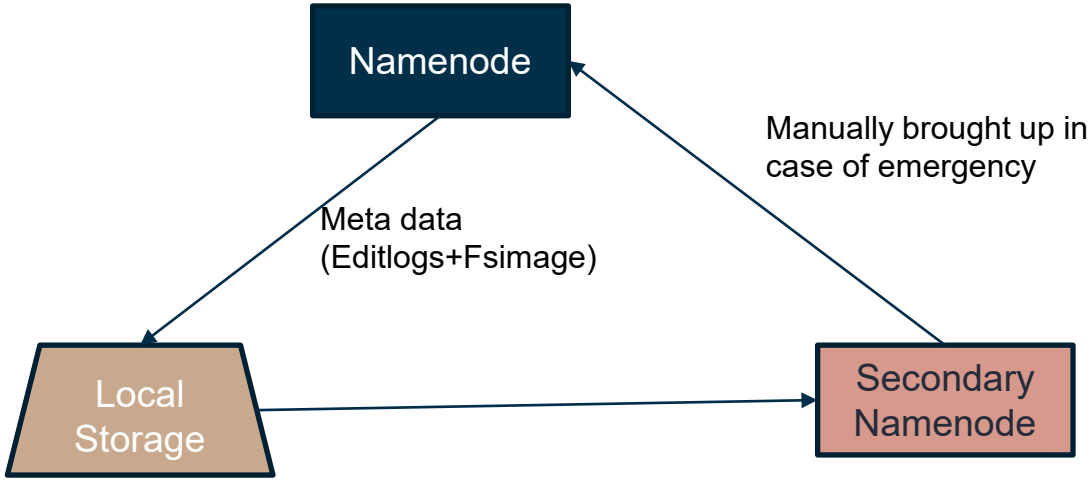
Name Node-Single Point of Failure

- 1. Keeping a backup of the name node image & Editlogs on two locations



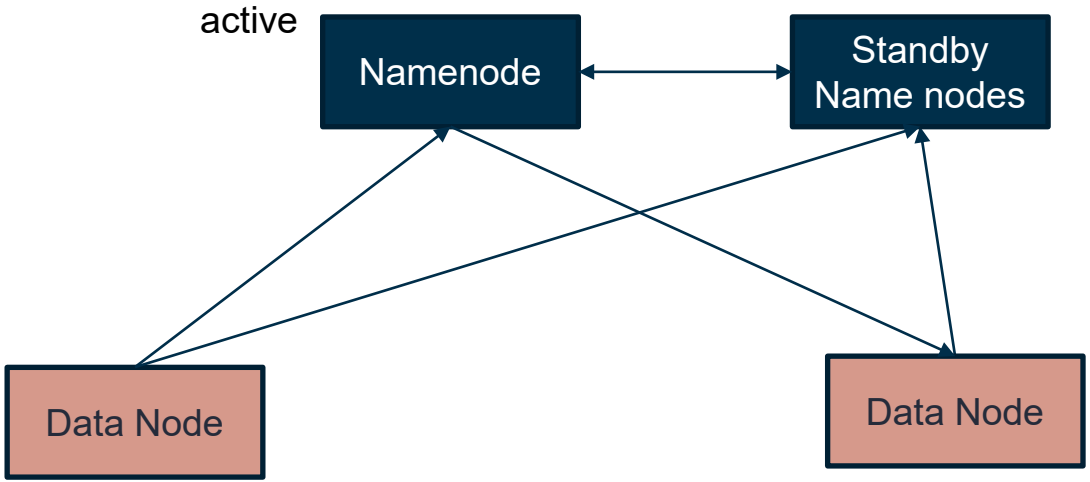
Name Node-Single Point of Failure (contd)

- 2. Keeping a secondary Name node



Name Node-Single Point of Failure (contd)

- 3. Keeping a standby namenode



Cluster Rebalancing

- HDFS architecture is compatible with data rebalancing schemes
- A scheme may autometically move data from one DataNode to another if the free space on a DataNode is falling below a certain threshold
- A scheme may dynamically create and place additional replicas and rebalance other data if there is sudden high demand for a particular file
- These types of rebalancing are not yet implemented

Data Organization

HDFS-Data Block

- HDFS support write-once-read-many with reads at streaming speeds.
- A typical block size is 64MB (or even 128 MB).
- A file is chopped into 64MB chunks and stored.
- Advantages

→helps fitting big files into small discs

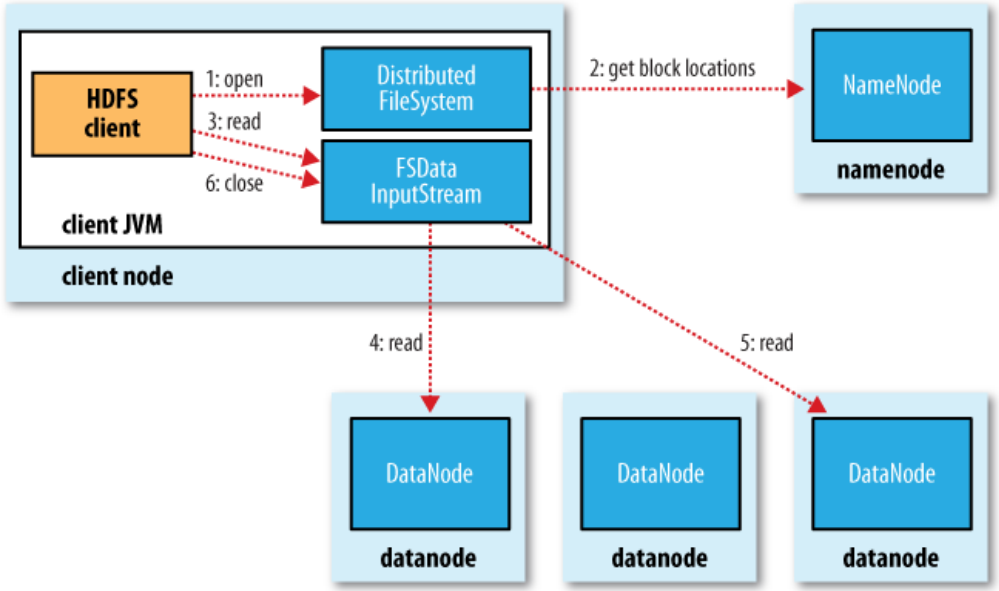
→ leaves less unused space on the disc

→Optimizes the transfer

→ distributes the load to multiple machines

HDFS –Data Flow

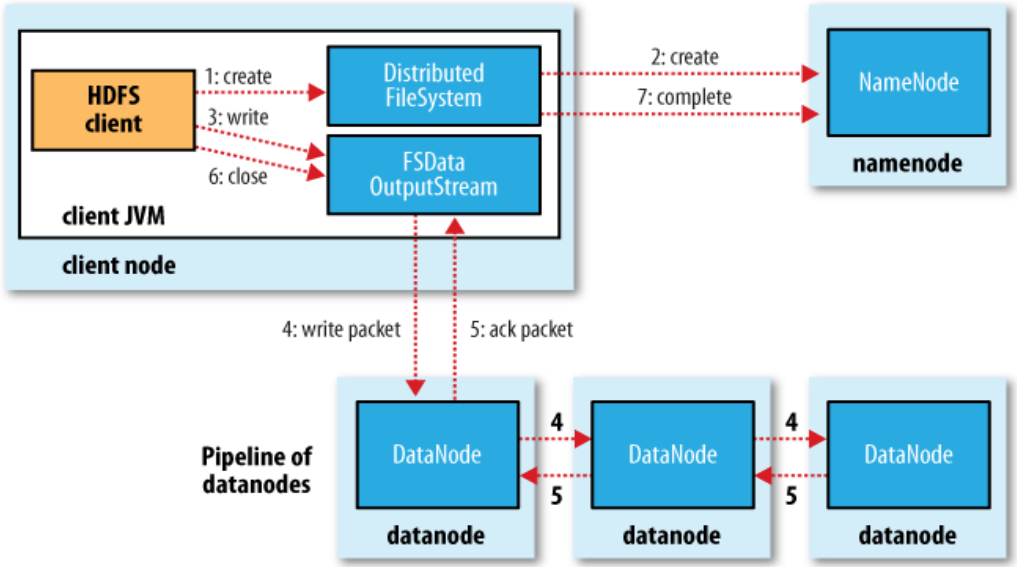
- Client reading data from a HDFS



Ebook :Hadoop the definitive guide by tom white 3rd edition

HDFS –Data Flow

- Client writing data to HDFS



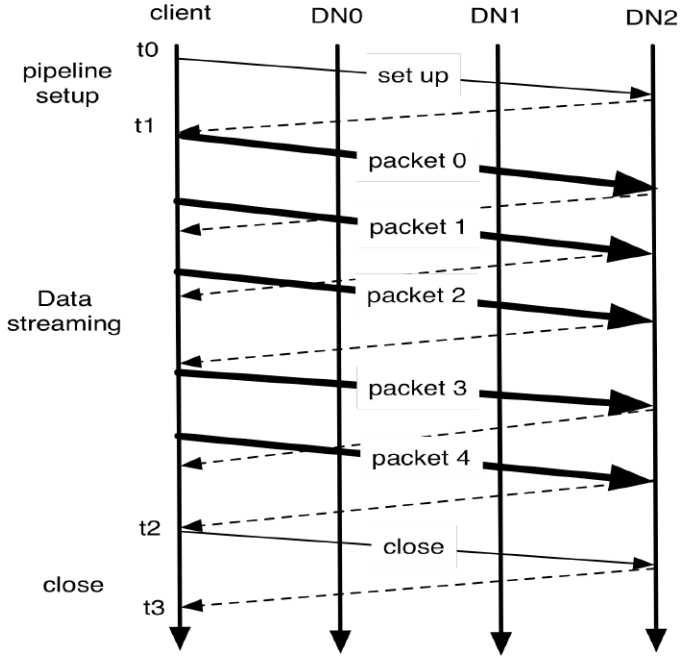
Ebook :Hadoop the definitive guide by tom white 3rd edition

Replication -Pipeline

- When a client is writing data to an HDFS file with a replication factor of three,the NameNode retrieves a list of DataNodes using a replication target choosing algorithm.
- It contains the DataNodes that will host a replica of that block.
- The client then writes to the first DataNode, starts receiving the data in portions, writes each portion to the second DataNode in the list.
- The second DataNode, in turn starts receiving each portion of the data block, writes that portion to its repository and then flushes that portion to the third DataNode.
- Finally, the third DataNode writes the data to its local repository.

- Thus, the data is pipelined from one DataNode to the next.

Data –Pipeline



[1] Hadoop distributed file system Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler Yahoo! Sunnyvale, California USA

Staging

- A client request to create a file does not reach Namenode immediately.
- HDFS client caches the data into a temporary file. When the data reached a HDFS block size the client contacts the Namenode.
- Namenode inserts the filename into its hierarchy and allocates a data block for it.
- The Namenode responds to the client with the identity of the Datanode and the destination of the replicas (Datanodes) for the block.

Staging (Cont.)

- The client sends a message that the file is closed.
- Namenode proceeds to commit the file for creation operation into the persistent store.
- If the Namenode dies before file is closed, the file is lost.
- This client side caching is required to avoid network congestion; also it has precedence is AFS (Andrew file system).

Communication Protocols

The Communications Protocols

- All HDFS communication protocols are layered on top of the TCP/IP protocol
- A client establishes a connection to a configurable TCP port on the Namenode machine. It talks ClientProtocol with the Namenode.
- The Datanodes talk to the Namenode using Datanode protocol.
- A remote procedure call (RPC) abstraction wraps both ClientProtocol and Datanode protocol.
- Namenode is simply a server and never initiates a request; it only responds to RPC requests issued by DataNodes or clients.

API

FS Shell, Admin, and Browser Interface

- HDFS organizes its data in files and directories
- It provides a commandline interface called FS shell that lets a user interact with the data in HDFS.
- The syntax of this command set is similar to other shells (e.g. bash, csh) that users are already familiar with. Here are some sample action/command pairs:

Action	Command
Create a directory named /foodir	bin/hadoop dfs -mkdir /foodir
Remove a directory named /foodir	bin/hadoop fs -rm -R /foodir

FS Shell, Admin, and Browser Interface

- There is also a DFSAdmin interface
- These are commands that are used only by an HDFS administrator. Here are some sample action/command pairs:

Action	Command
Put the cluster in Safemode	bin/hdfs dfsadmin -safemode enter
Generate a list of DataNodes	bin/hdfs dfsadmin -report

- A browser can be used to view the namespace

Space Reclamation

- When a file is deleted, HDFS moves it to a trash directory for a configurable amount of time
- A client can request for the file to be recovered during this time
- After the specified time the file is deleted along with replicas, and all space is reclaimed
- This will also occur automatically if the replication factor is reduced