High-level Architecture

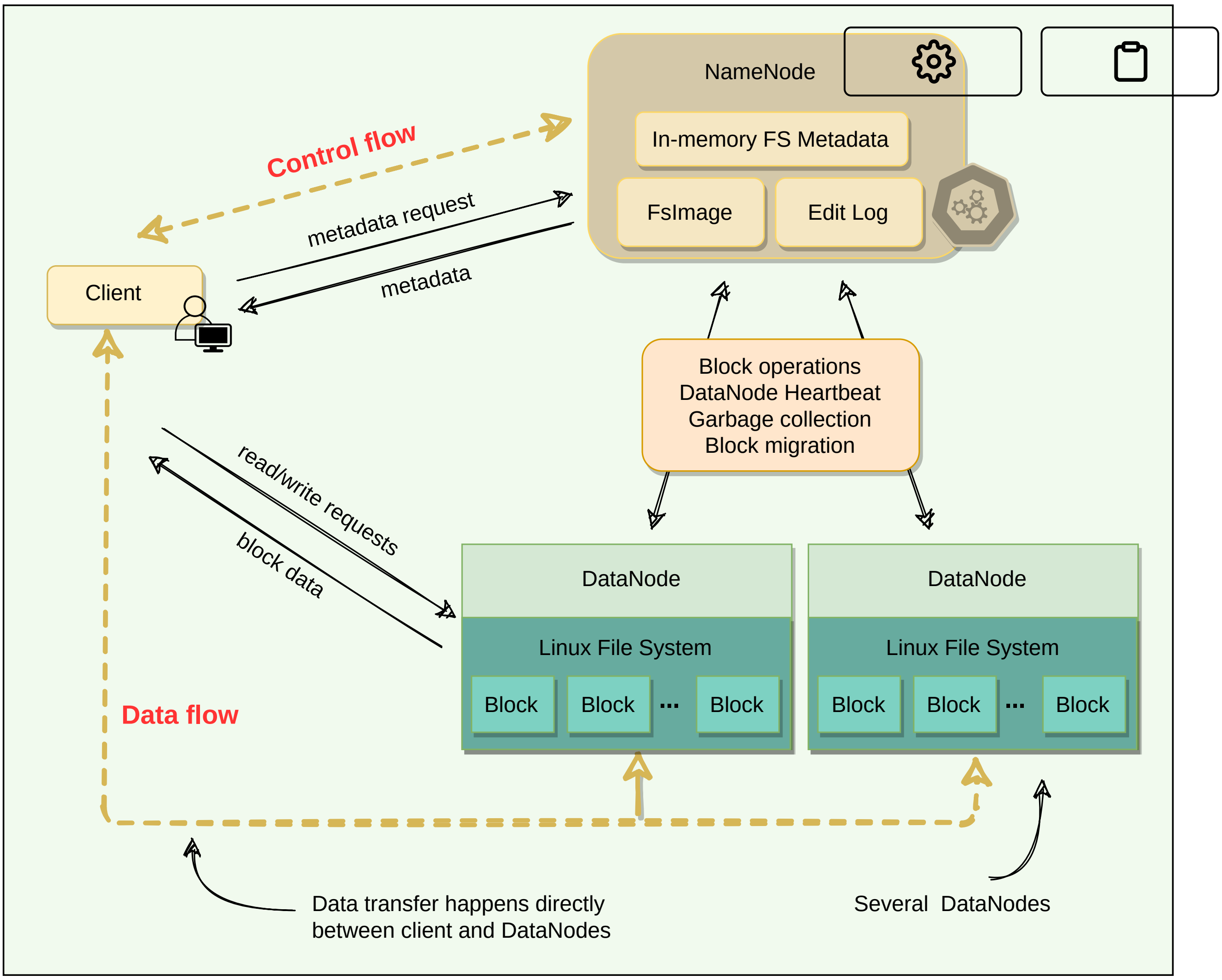
This lesson gives a brief overview of HDFS's architecture.

We'll cover the following

- HDFS architecture
- Comparison between GFS and HDFS

HDFS architecture#

All files stored in HDFS are broken into multiple fixed-size blocks, where each block is 128 megabytes in size by default (configurable on a per-file basis). Each file stored in HDFS consists of two parts: the **actual file data** and the **metadata**, i.e., how many block parts the file has, their locations and the total file size, etc. HDFS cluster primarily consists of a **NameNode** that manages the file system metadata and **DataNodes** that store the actual data.

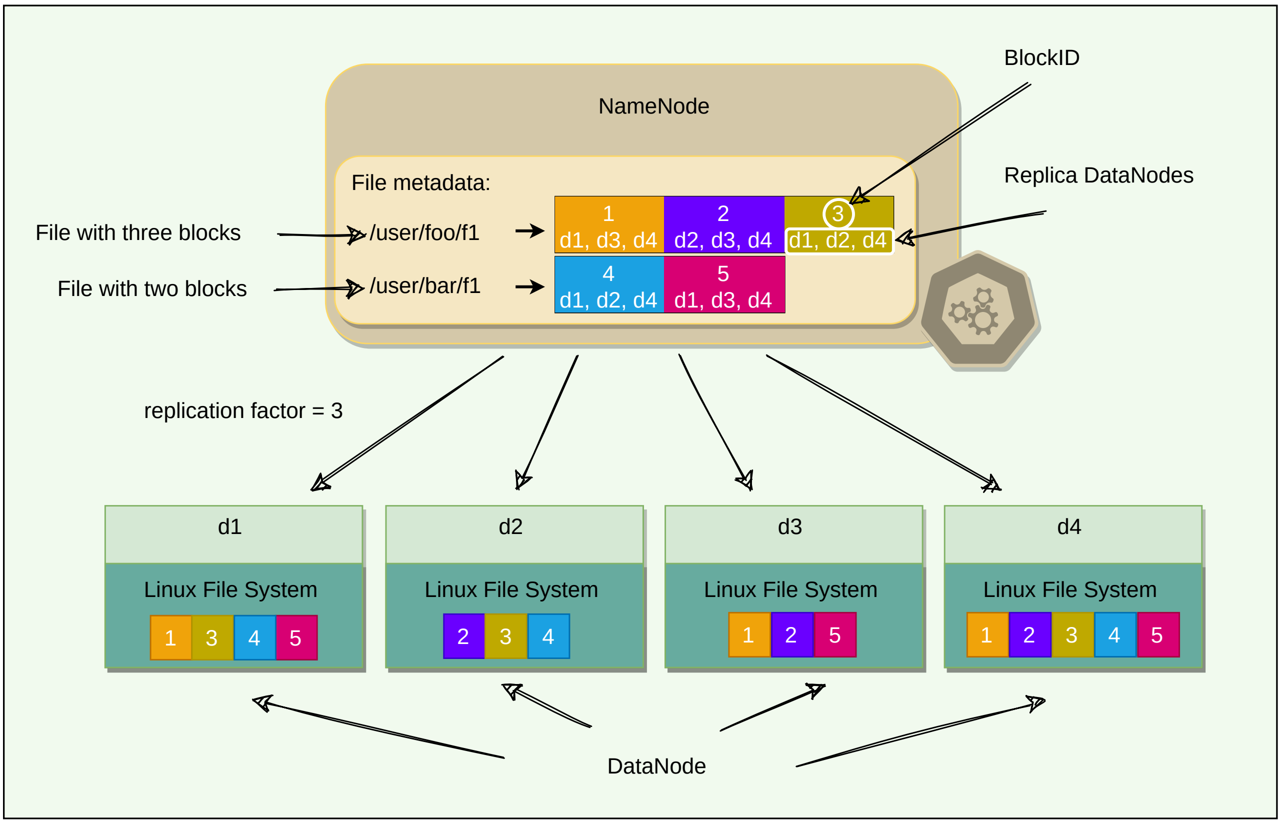


HDFS high-level architecture

- All blocks of a file are of the same size except the last one.
- HDFS uses **large block sizes** because it is designed to store extremely large files to enable MapReduce jobs to process them efficiently.
- Each block is identified by a unique 64-bit ID called **BlockID**.
- All read/write operations in HDFS operate at the block level.
- DataNodes store each block in a separate file on the local file system and provide read/write access.
- When a DataNode starts up, it scans through its local file system and sends the list of hosted data blocks (called BlockReport) to the NameNode.
- The NameNode maintains two on-disk data structures to store the file system's state: an **FsImage** file and an **EditLog**. FsImage is a checkpoint of the file system metadata at some point in time, while

the EditLog is a log of all of the file system metadata transactions since the image file was last created. These two files help NameNode to recover from failure.

- User applications interact with HDFS through its client. HDFS Client interacts with NameNode for metadata, but all data transfers happen directly between the client and DataNodes.
- To achieve high-availability, HDFS creates multiple copies of the data and distributes them on nodes throughout the cluster.



HDFS block replication

Comparison between GFS and HDFS#

HDFS architecture is similar to GFS, although there are differences in the terminology. Here is the comparison between the two file systems:

	GFS	H
Storage node	ChunkServer	Dat
File part	Chunk	B
File part size	Default chunk size is 64MB (adjustable)	Default block size i
Metadata Checkpoint	Checkpoint image	FsI
Write ahead log	Operation log	Ec
Platform	Linux	Cross
Language	Developed in C++	Develop
Available Implementation	Only used internally by Google	Oper
Monitoring	Master receives HeartBeat from ChunkServers	NameNode recei Data
Concurrency	Follow multiple writers and multiple readers model	Does not support m follows the write-c m
File Operations	Append and random writes are possible	Only apper
Garbage Collection	Any deleted file is renamed into a particular folder to be garbage collected later	Any deleted file is name to be garb
Communication	RPC over TCP is used for communication with the master To minimize latency, pipelining and streaming are used over TCP for data transfer.	RPC over TCP communication witl For data transf streaming are
Cache Management	Client cache metadata Client or ChunkServer does not cache file data ChunkServers rely on the buffer cache in Linux to maintain frequently accessed data in memory	HDFS uses d User-specified explicitly in the Data off-heap The cache could be one user) or public users of the

Replication Strategy	Chunk replicas are spread across the racks. Master automatically replicates the chunks. By default, three copies of each chunk are stored. User can specify a different replication factor. The master re-replicates a chunk replica as soon as the number of available replicas falls below a user-specified number.	The HDFS has an a replication system. By default, two copi stored at two differe same rack, and a th a Data Node in a di reliability). User can specify a fa
File system Namespace	Files are organized hierarchically in directories and identified by pathnames.	HDFS supports a tr file organization. Us can create directori inside. HDFS also supp systems such a Storage Service (S
Database	Bigtable uses GFS as its storage engine.	HBase uses HDFS

← Back

Hadoop Distributed File System: Intro...

Next →

Deep Dive

☑ Mark as Completed

Report an Issue