SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2010

Apache Hadoop FileSystem Internals

Dhruba Borthakur
Project Lead, Apache Hadoop Distributed File System
dhruba@apache.org
Presented at Storage Developer Conference, San Jose
September 22, 2010
http://www.facebook.com/hadoopfs

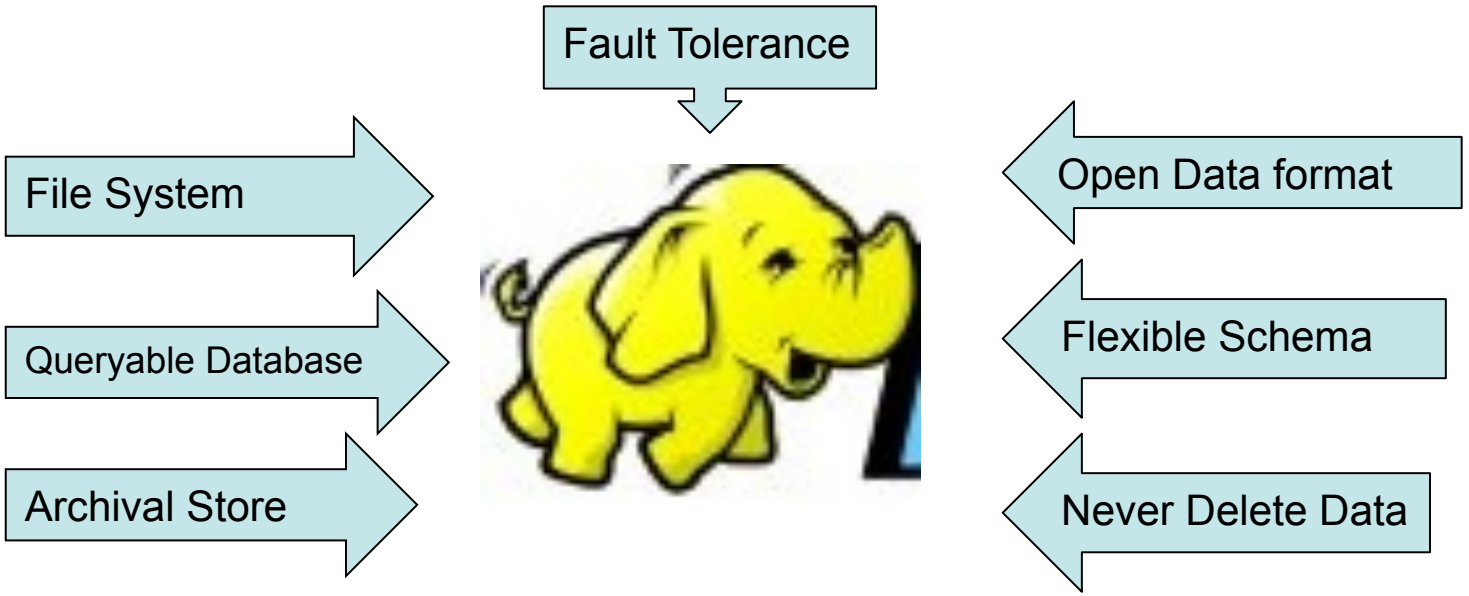
Outline

- Introduction
- Architecture of Hadoop Distributed File System (HDFS)
- HDFS High Availability
- HDFS RAID

Who Am I?

- **Apache Hadoop FileSystem (HDFS)**
  - Project Lead
  - Core contributor since Hadoop's infancy
- **Facebook** (Hadoop, Hive, Scribe)
- **Yahoo!** (Hadoop in Yahoo Search)
- **Veritas** (San Point Direct, Veritas File System)
- **IBM Transarc** (Andrew File System)
- **Univ of Wisconsin Computer Science Alumni** (Condor Project)

A Confluence of Trends



HADOOP: A Massively Scalable Queryable Store and Archive

hadoop

Hadoop, Why?

- **Need to process Multi Petabyte Datasets**
- **Data may not have strict schema**
- **Expensive to build reliability in each application.**
- **Nodes fail every day**
  - Failure is expected, rather than exceptional.
  - The number of nodes in a cluster is not constant.
- **Need common infrastructure**
  - Efficient, reliable, Open Source Apache License

Is Hadoop a Database?

- Hadoop triggered upheaval in Database Research
  - "A giant step backward in the programming paradigm", Dewitt et el
  - "DBMS performance outshines Hadoop" – Stonebraker, Dewitt, SIGMOD 2009
- Parallel Databases
  - A few scales to 200 nodes and about 5 PB
  - Primary design goal is "performance"
  - Requires homogeneous hardware
  - Anomalous behavior is not well tolerated:
    - A slow network can cause serious performance degradation
    - Most queries fail when one node fails
- Scalability and Fault Tolerance: Hadoop to the rescue!

Hadoop History

- Dec 2004 – Google GFS paper published
- July 2005 – Nutch uses MapReduce
- Feb 2006 – Starts as a Lucene subproject
- Apr 2007 – Yahoo! on 1000-node cluster
- Jan 2008 – An Apache Top Level Project
- May 2009 – Hadoop sorts Petabyte in 17 hours
- Aug 2010 – World's Largest Hadoop cluster at Facebook
  - 2900 nodes, 30+ PetaByte

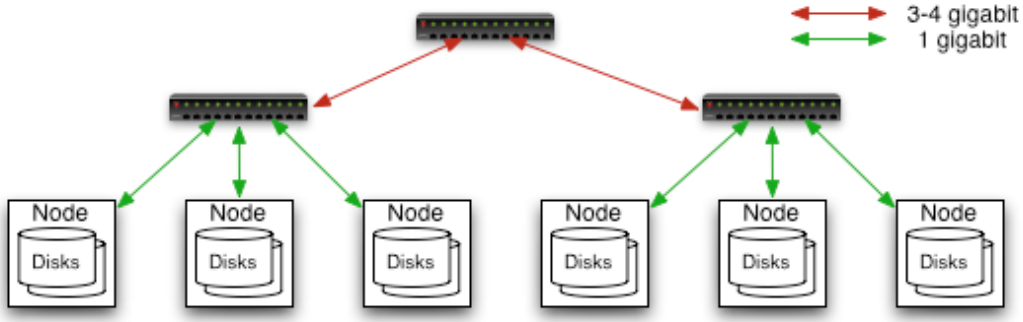
Who uses Hadoop?

- Amazon/A9
- Facebook
- Google
- IBM
- Joost
- Last.fm
- New York Times
- PowerSet
- Veoh
- Yahoo!

What is Hadoop used for?

- Search
  - Yahoo, Amazon, Zvents
- Log processing
  - Facebook, Yahoo, ContextWeb. Joost, Last.fm
- Recommendation Systems
  - Facebook
- Data Warehouse
  - Facebook, AOL
- Video and Image Analysis
  - New York Times, Eyealike

Commodity Hardware



Typically in 2 level architecture

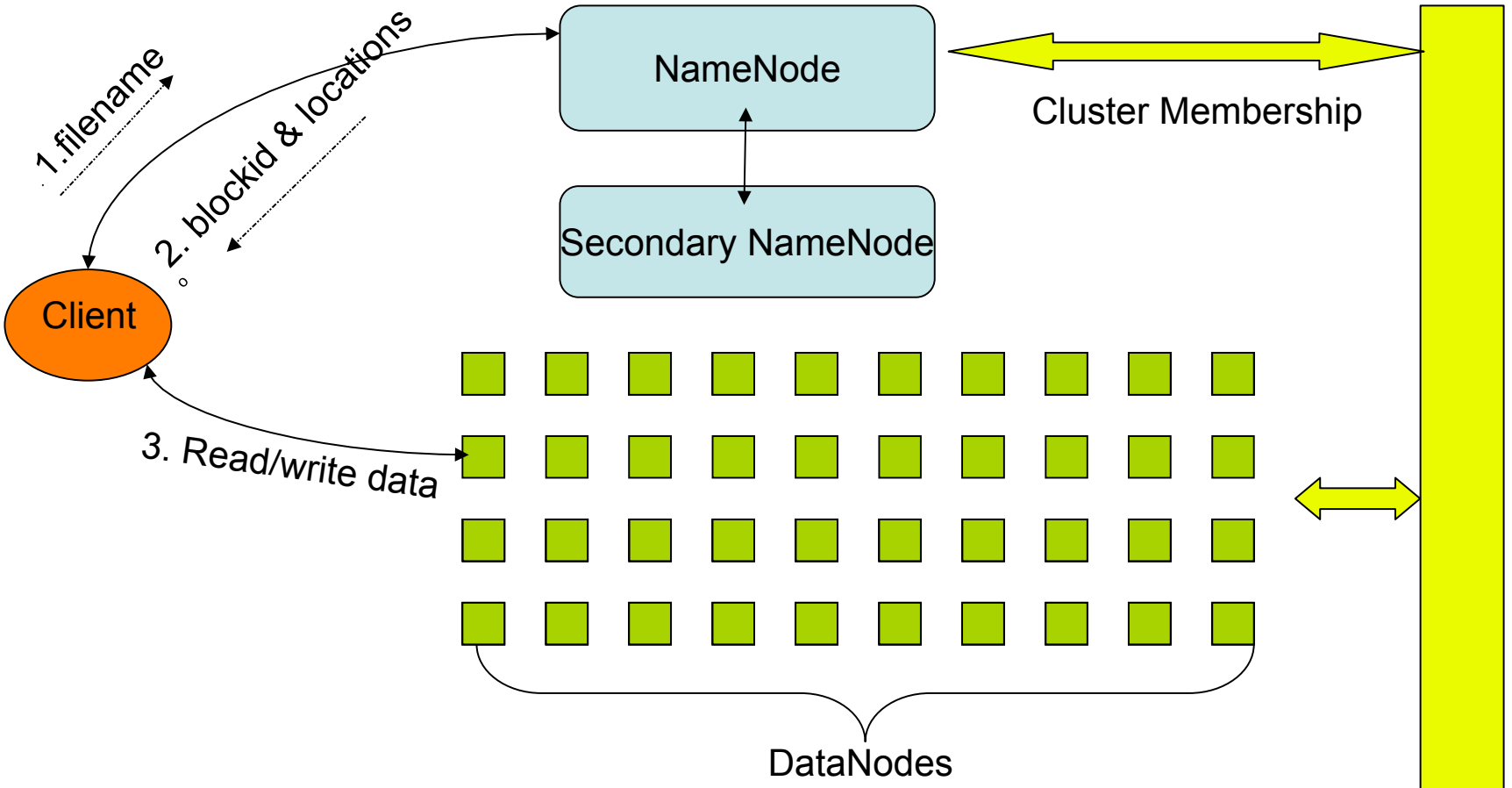
– Nodes are commodity PCs
– 20-40 nodes/rack
– Uplink from rack is 4 gigabit
– Rack-internal is 1 gigabit

hadoop

Goals of HDFS

- **Very Large Distributed File System**
  - 10K nodes, 1 billion files, 100 PB
- **Assumes Commodity Hardware**
  - Files are replicated to handle hardware failure
  - Detect failures and recovers from them
- **Optimized for Batch Processing**
  - Data locations exposed so that computations can move to where data resides
  - Provides very high aggregate bandwidth
- **User Space, runs on heterogeneous OS**

HDFS Architecture

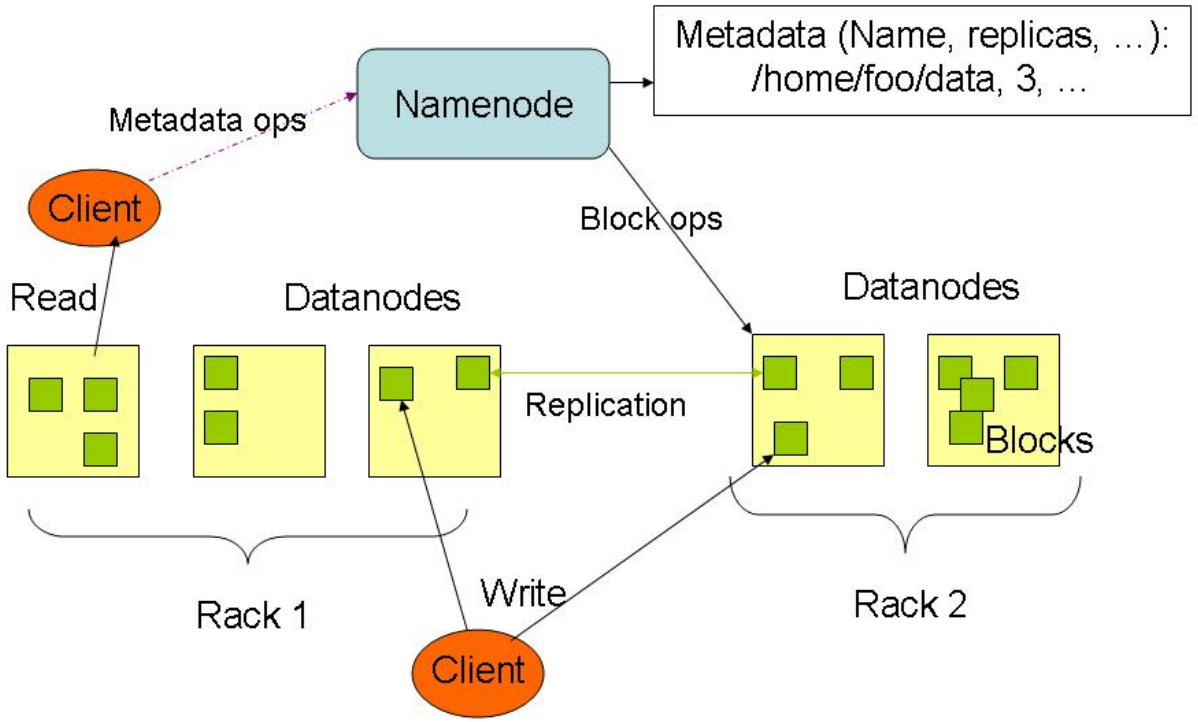


NameNode : Maps a file to a file-id and list of DataNodes
DataNode : Maps a block-id to a physical location on disk
SecondaryNameNode: Periodic merge of Transaction log

Distributed File System

- **Single Namespace for entire cluster**
- **Data Coherency**
  - Write-once-read-many access model
  - Client can only append to existing files
- **Files are broken up into blocks**
  - Typically 128 - 256 MB block size
  - Each block replicated on multiple DataNodes
- **Intelligent Client**
  - Client can find location of blocks
  - Client accesses data directly from DataNode

HDFS Architecture



NameNode Metadata

- **Meta-data in Memory**
  - The entire metadata is in main memory
  - No demand paging of meta-data
- **Types of Metadata**
  - List of files
  - List of Blocks for each file
  - List of DataNodes for each block
  - File attributes, e.g creation time, replication factor
- **A Transaction Log**
  - Records file creations, file deletions. etc

DataNode

- **A Block Server**
  - Stores data in the local file system (e.g. ext3)
  - Stores meta-data of a block (e.g. CRC32)
  - Serves data and meta-data to Clients
  - Periodic validation of checksums
- **Block Report**
  - Periodically sends a report of all existing blocks to the NameNode
- **Facilitates Pipelining of Data**
  - Forwards data to other specified DataNodes

Block Placement

- **Current Strategy**
  - -- One replica on local node
  - -- Second replica on a remote rack
  - -- Third replica on same remote rack
  - -- Additional replicas are randomly placed
- **Clients read from nearest replica**
- **Pluggable policy for placing block replicas**
  - Co-locate datasets that are often used together
  - http://hadoopblog.blogspot.com/2009/09/hdfs-block-replica-placement-in-your.html

Data Pipelining

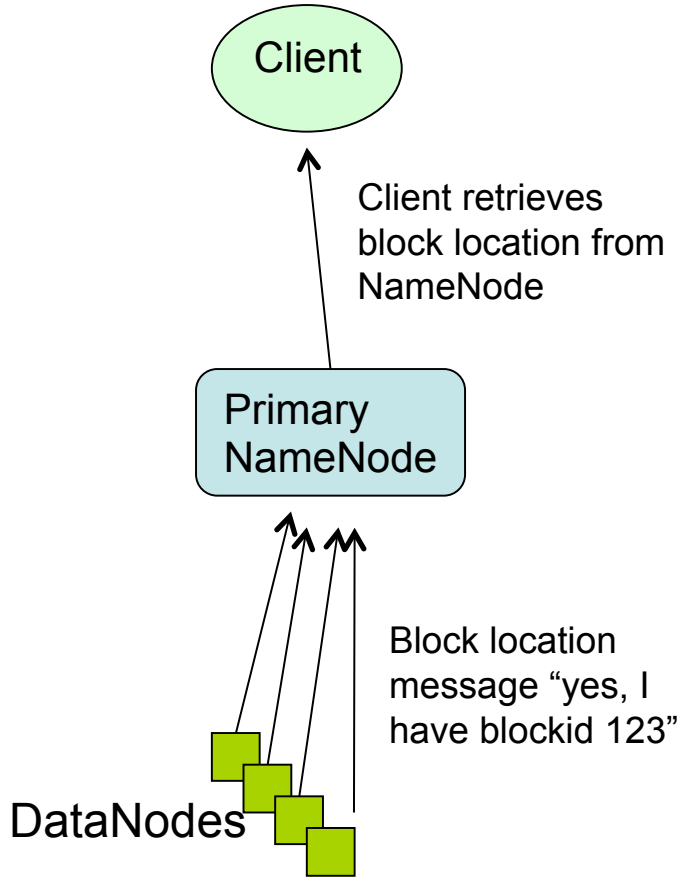
- Client writes block to the first DataNode
- The first DataNode forwards the data to the next DataNode in the Pipeline, and so on
- When all replicas are written, the Client moves on to write the next block in file

NameNode Failure

- **A Single Point of Failure**
- **Transaction Log stored in multiple directories**
  - A directory on the local file system
  - A directory on a remote file system (NFS/CIFS)
- **This is a problem with 24 x 7 operations, no joke!**
  - AvatarNode comes to the rescue

NameNode High Availability: Challenges

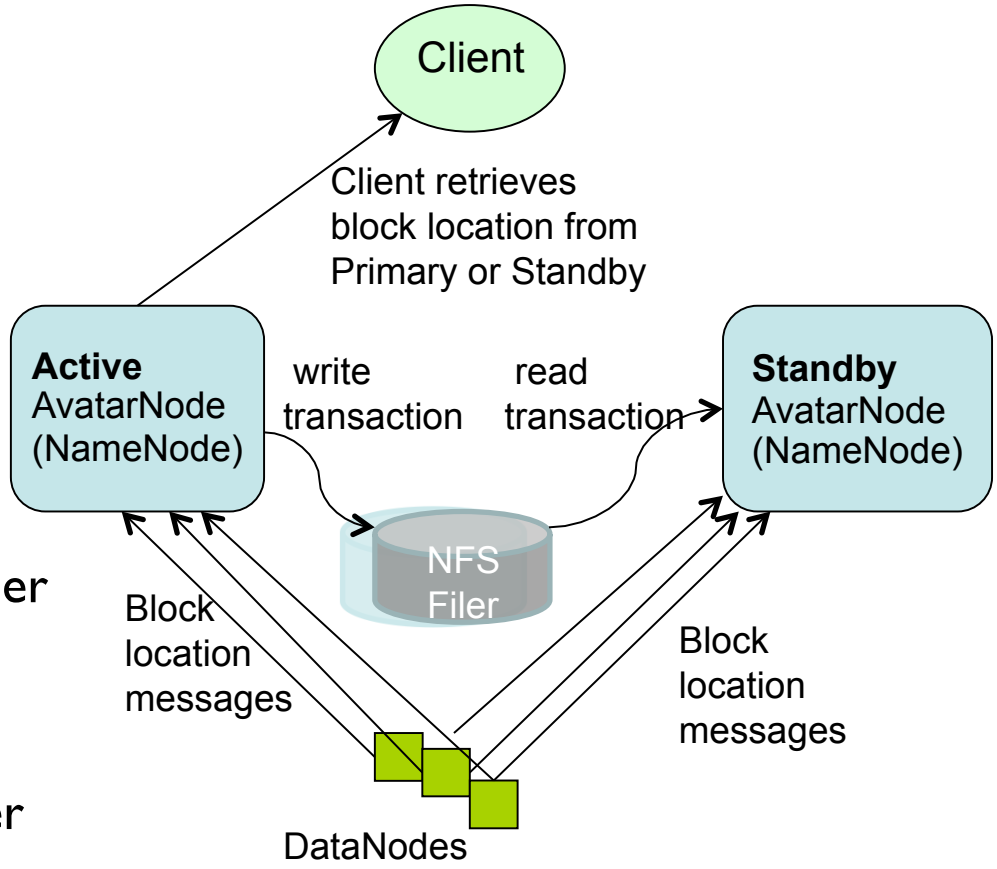
- DataNodes send block location information to only one NameNode
- NameNode needs block locations in memory to serve clients
- The in-memory metadata for 100 million files could be 60 GB, huge!



hadoop

NameNode High Availability: AvatarNode

- **Active-Standby Pair**
  - Coordinated via zookeeper
  - Failover in few seconds
  - Wrapper over NameNode
- **Active AvatarNode**
  - Writes transaction log to filer
- **Standby AvatarNode**
  - Reads transactions from filer
  - Latest metadata in memory



http://hadoopblog.blogspot.com/2010/02/hadoop-namenode-high-availability.html

hadoop

Rebalancer

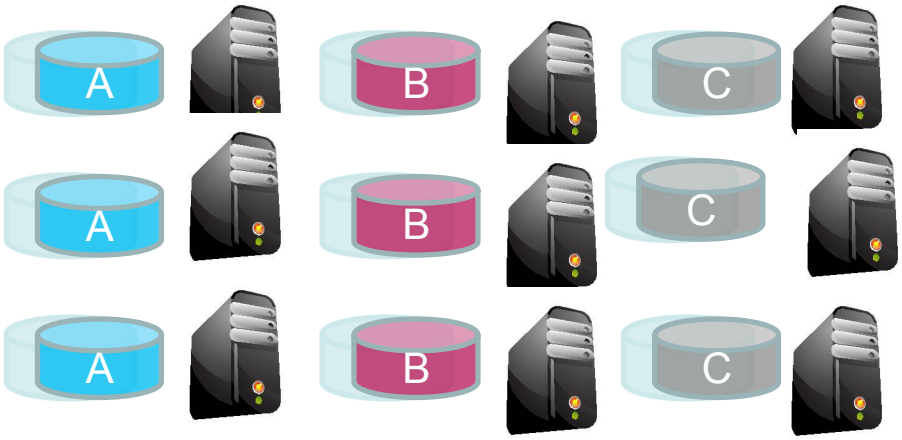
- **Goal: % disk full on DataNodes should be similar**
  - Usually run when new DataNodes are added
  - Cluster is online when Rebalancer is active
  - Rebalancer is throttled to avoid network congestion

- **Disadvantages**
  - Does not rebalance based on access patterns or load
  - No support for automatic handling of hotspots of data

hadoop

Disk is not cheap! - RAID

- A Data Block is stored in triplicate
- File /dir/file.txt
  - three data blocks
  - nine physical blocks on disk
- HDFS RAID to the rescue
  - DiskReduce from CMU
  - Garth Gibson research

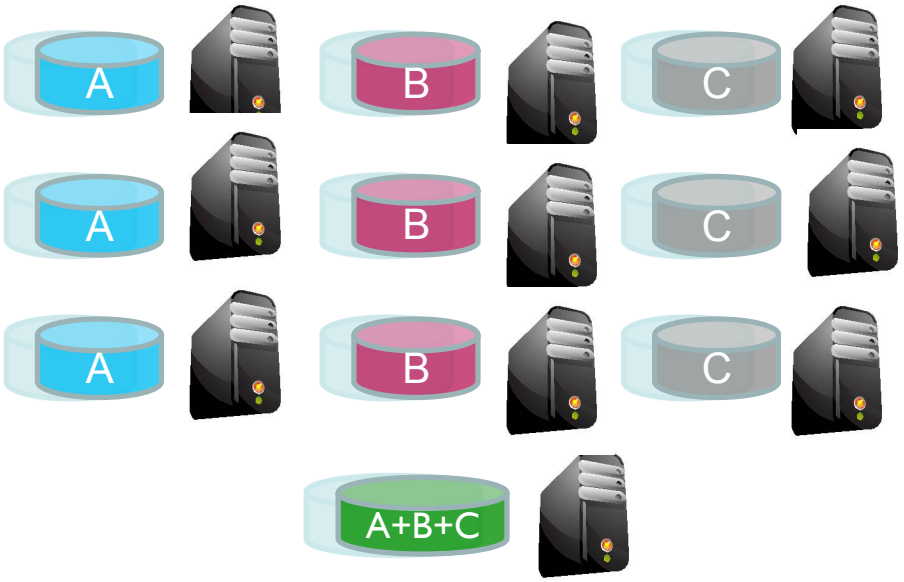


A file with three blocks A, B and C

hadoop

HDFS Raid

- Start the same: triplicate every data block
- Background encoding
  - Combine third replica of blocks from a single file to create parity block
  - Remove third replica
- RaidNode
  - Auto fix of failed replicas



A file with three blocks A, B and C

http://hadoopblog.blogspot.com/2009/08/hdfs-and-erasure-codes-hdfs-raid.html

hadoop

Useful Links

- **HDFS Design:**
  - http://hadoop.apache.org/core/docs/current/hdfs_design.html
- **Hadoop API:**
  - http://hadoop.apache.org/core/docs/current/api/
- **My Hadoop Blog:**
  - http://hadoopblog.blogspot.com/
  - http://www.facebook.com/hadoopfs