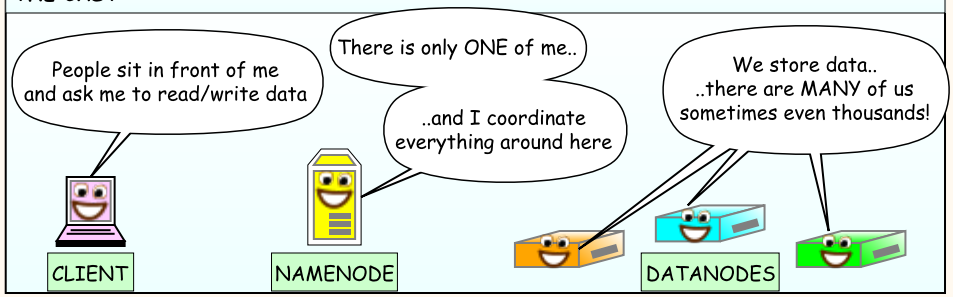
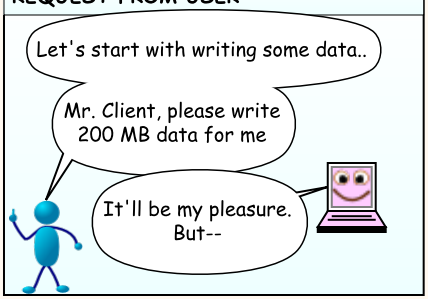
HADOOP DISTRIBUTED FILE SYSTEM (HDFS)

THE CAST

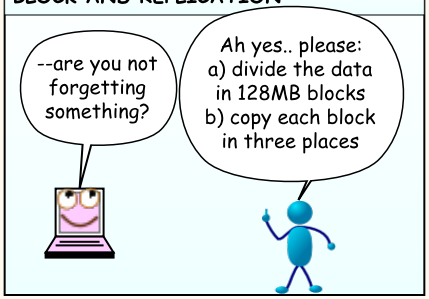


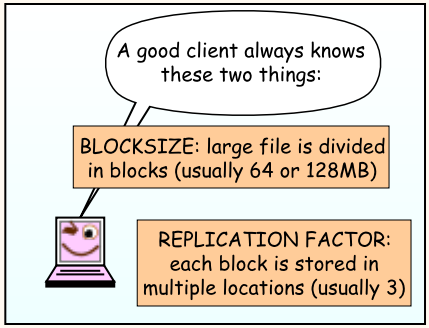
WRITING DATA IN HDFS CLUSTER

REQUEST FROM USER

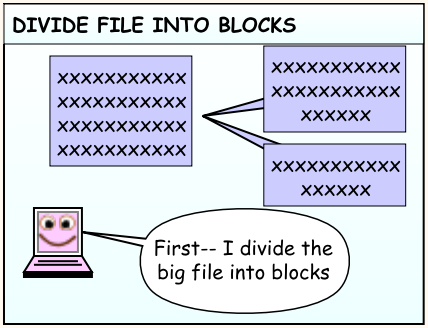


BLOCK AND REPLICATION

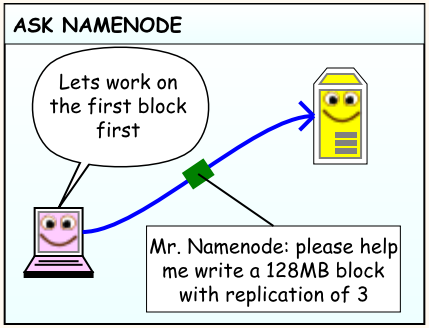




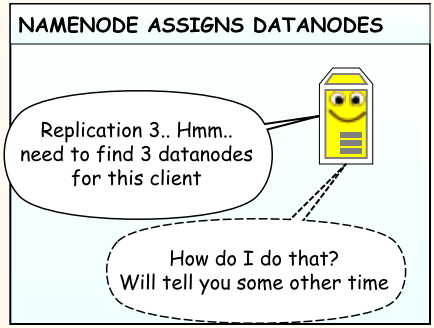
DIVIDE FILE INTO BLOCKS

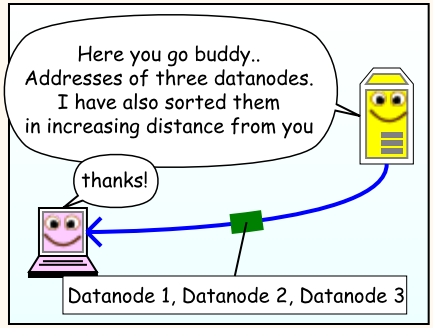


ASK NAMENODE

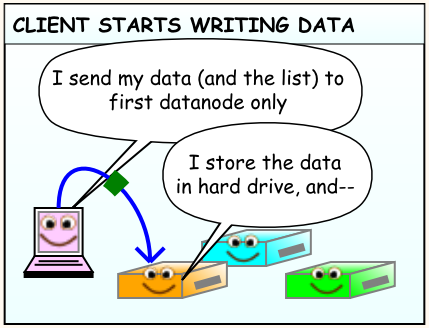


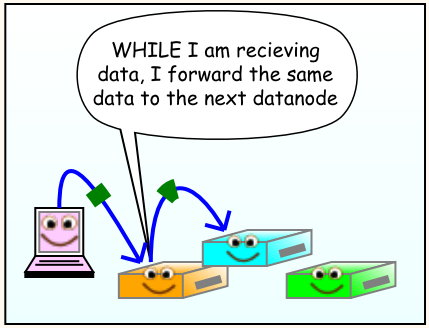
NAMENODE ASSIGNS DATANODES

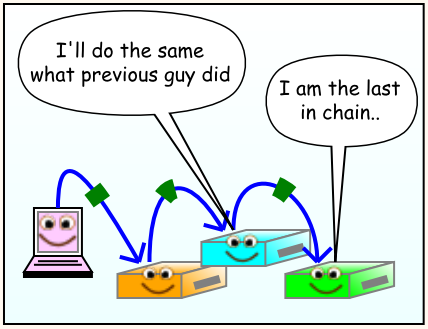


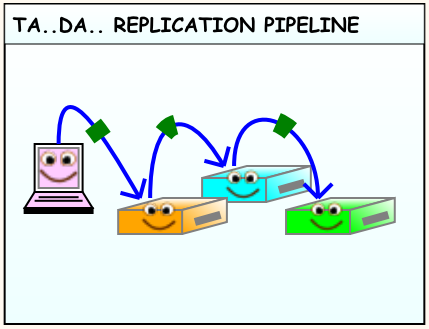


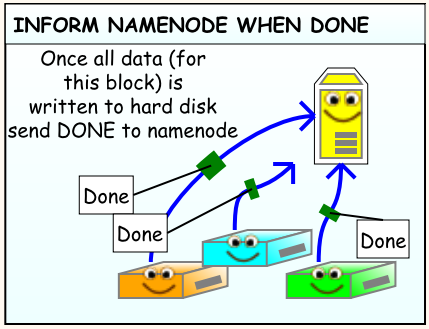
CLIENT STARTS WRITING DATA

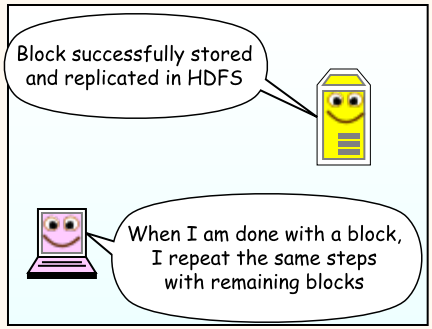


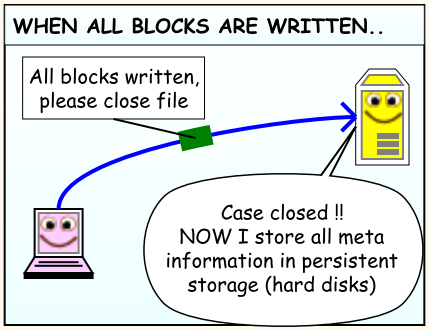


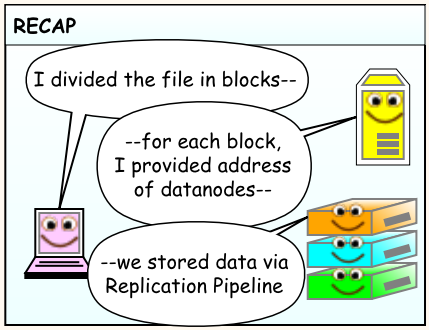




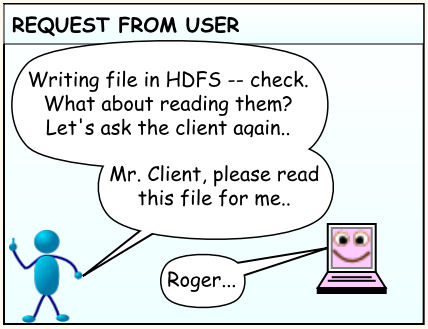


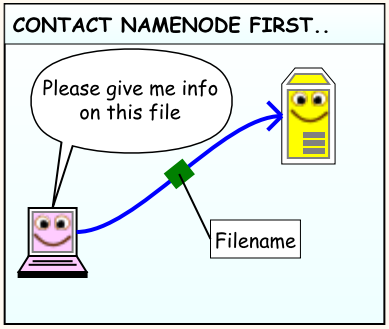


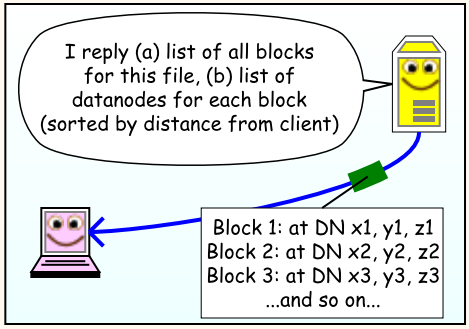


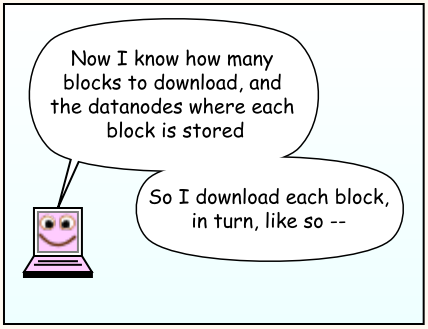


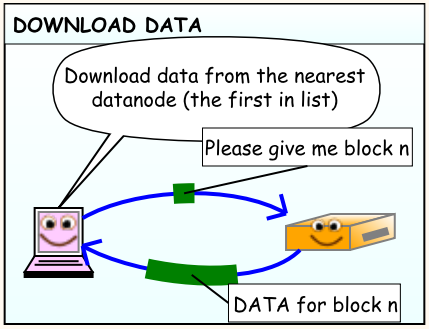
READING DATA IN HDFS CLUSTER

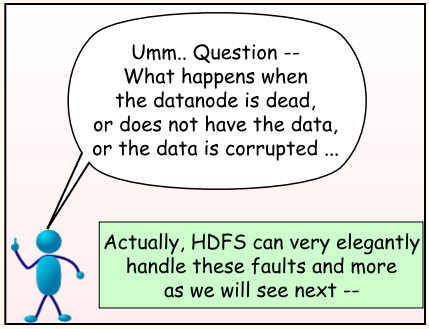






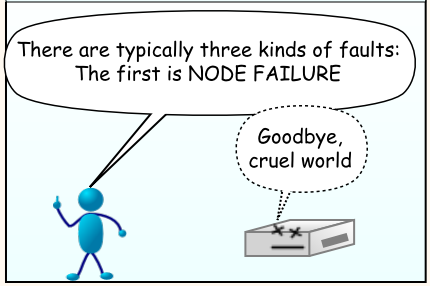




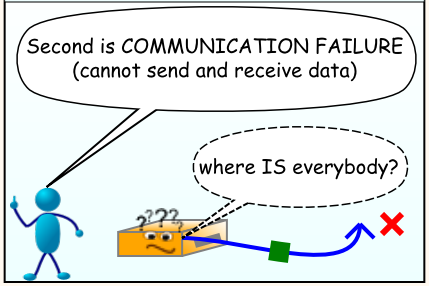


FAULT TOLERANCE IN HDFS. PART I: TYPES OF FAULTS AND THEIR DETECTION

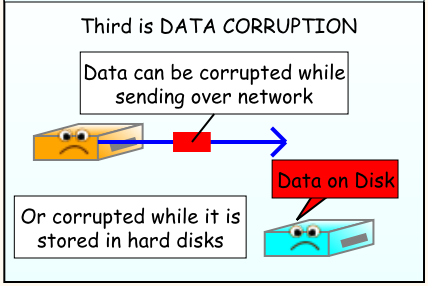
FAULT I: NODE FAILURE



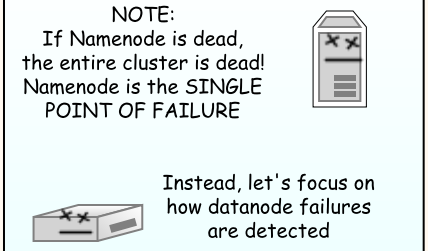
FAULT II: COMMUNICATION FAILURE



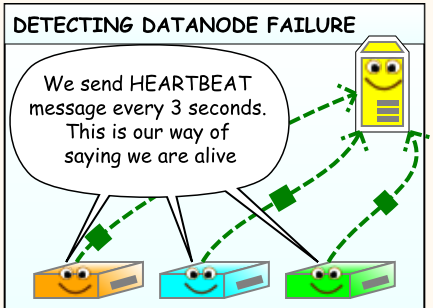
FAULT III: DATA CORRUPTION

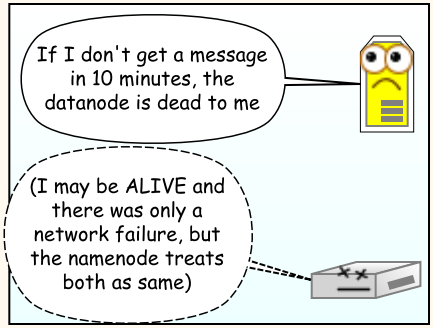


DETECTION #1: NODE FAILURES

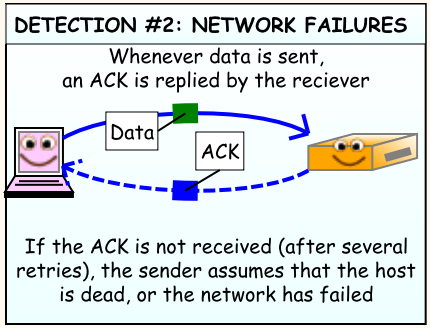


DETECTING DATANODE FAILURE

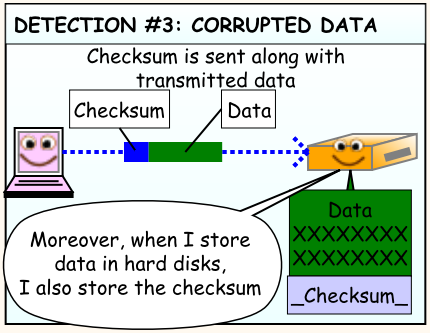




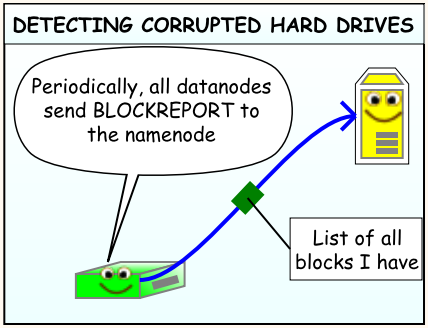
DETECTION #2: NETWORK FAILURES

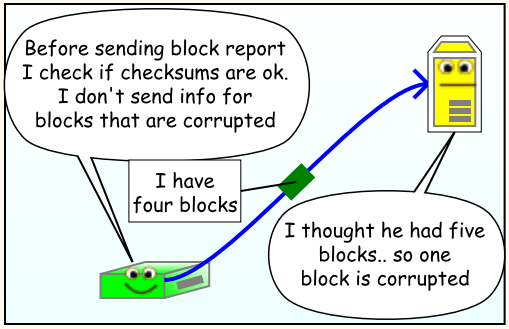


DETECTION #3: CORRUPTED DATA

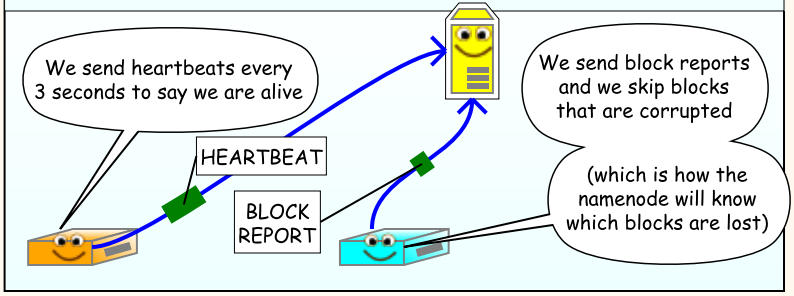


DETECTING CORRUPTED HARD DRIVES





RECAP: HEARTBEAT MESSAGES AND BLOCK REPORTS



FAULT TOLERANCE IN HDFS. PART II: HANDLING READING AND WRITING FAILURES

HANDLING WRITE FAILURES

One thing I should have said earlier.. I write the block in smaller data units (usually 64KB) called "packets"

Packet

Remember replication pipeline?

Moreover, each datanode replies back an ACK for each packet to confirm that they got it

ACK

So, if I don't get ACKs from some datanode, I know it is dead. I adjust the pipeline to skip him

Here's the adjusted pipeline. Note that the block will be "under replicated", but the namenode will take care of that later on

HANDLING READ FAILURES

Remember, when I asked for location of a block, the namenode gave me locations of all datanodes

DN 1, DN 2, DN 3

If one datanode is dead, I read from the others in the list

Got Data? No?

Got Data?

FAULT TOLERANCE IN HDFS. PART III: HANDLING DATANODE FAILURES

First-- I must tell you about the two tables I keep..

List of Blocks
Block 1 - stored at DN1, DN2, DN3
Block 2 - stored at DN1, DN4, DN5

List of Datanodes
Datanode 1 - has block 1, 2, ..
Datanode 2 - has block 1, 5, ..

I continuously update these two tables--

If I find a block on a datanode is corrupted, I update first table (by removing bad DN from block's list)

And if I find that a datanode has died, I update both tables

UNDER REPLICATED BLOCKS

I scan the first list (list of blocks) periodically, and see if there are blocks that are not replicated properly

These are called "under replicated" blocks

For all under-replicated blocks, I ask other datanodes to copy them from datanodes that have the replica

like so --

Could you copy the block from that datanode

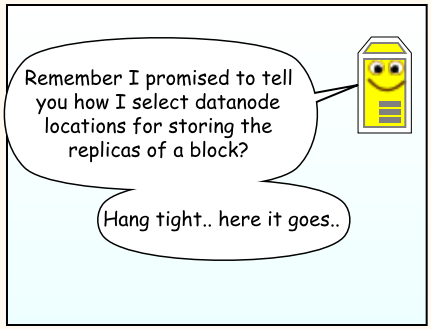
Hey, I need to copy a block from you

Here you go..

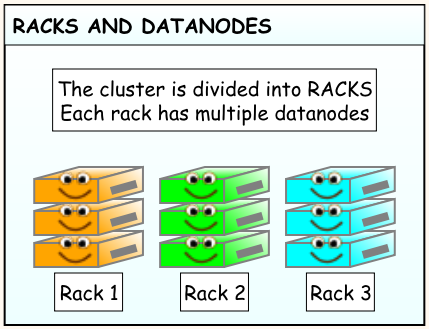
Umm.. one more question: All of this works if there is atleast one valid copy of the block somewhere.. right?

That's correct. HDFS cannot guarantee that atleast one replica will always survive. But it tries it best by smartly selecting replica locations, as we will see next --

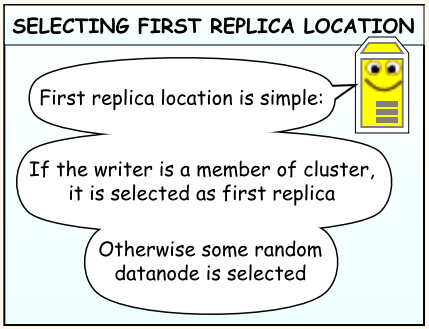
REPLICA PLACEMENT STRATEGY



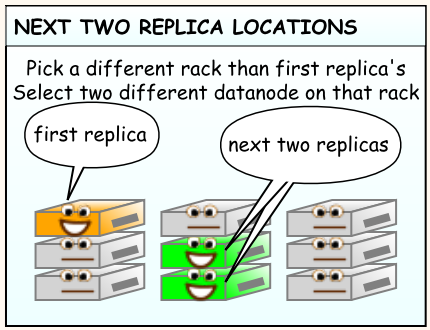
RACKS AND DATANODES



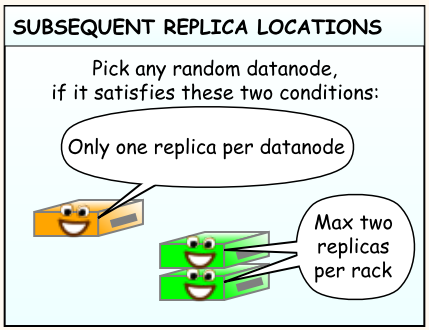
SELECTING FIRST REPLICA LOCATION

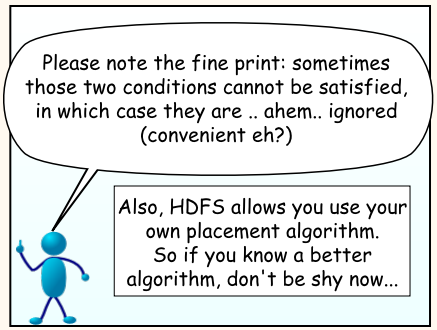


NEXT TWO REPLICA LOCATIONS



SUBSEQUENT REPLICA LOCATIONS





WHERE TO GO FROM HERE?