Get rid of the extra ChunkInfo metadata in PutBlock

Problem:

(1) PutBlock sends the chunk list of the entire block.
(2) If hsync is called, it wraps the latest

Symptoms

(1) Hsync performance severely crippled by the extra metadata.
(2) Ratis messages are too long and the Ratis log rotates very quickly. Several logs rotates every second.
Impact: HDDS-8047 and HDDS-8769.

Proposal

Data Structure

BlockDataTable<String, BlockData>
BlockData = {
  DatanodeBlockID BlockID,
  Int64 flag (optional),
  Map<String, String> metadata,
  List<ChunkInfo> chunks,
  Int64 Size (optional)
}

ChunkInfo = {
  String chunkName,
  Uint64 offset,
  Uint64 len,
  KeyValue metadata,
  ChecksumData checksumData,
  Bytes stripeChecksum,
}

Flush: Send only the last chunk info

Client BlockOutputStream maintains a lastChunkBuffer. It's going to occupy 4MB of buffer for each block and is used to calculate checksum.

WriteChunk sends the buffer from where it last had written a chunk. (this doesn't change)
PutBlock sends from the last full chunk boundary using the lastChunkBuffer. (it creates a fake chunkinfo).

If not flush:
PutBlock

Client: if server supports incremental chunk list and the container is in schema v3 (or v4).

Flush: Server side: Solution 1

PutBlock
Get the block chunk list from rocksdb; if the last chunk is partial, drop it and append the incremental chunk list from PutBlock message.

Flush: Server side: Solution 2 (selected for implementation)

A separate rocksdb table for last chunk

WriteChunk
Append the chunk. This is the same as before.

PutBlock
(If flush)
If partial chunk:
Update the LastChunk rocksdb table.

If end of chunk or if end of block:
Append the last chunk into the block table.

Flush: Server side: Solution 3

Use a file to store BlockData metadata. (similar to HDFS)

If partial chunk (meaning hsync),

Server side test cases (TestBlockManagerImpl):

1. Continuous partial chunk flush (maybe 2 or 3)
2. Write one full chunk, write one partial chunk and flush, and write/flush another partial chunk
3. write /flush a partial chunk, and then write continuously until 4 chunks are written.

Client side test cases: (happy path)

- BlockOutputStream write 1 byte, hsync, close
- BlockOutputStream write 1 full chunk + 1 byte, hsync, write 1 full chunk + 1 byte, close.
- BlockOutputStream write 4 full chunks -1 byte, hsync, write 2 bytes, hsync, close

PutBlock cases

- Case 1: Old client.
- Case 2: End of Block, the block does not have any full chunks yet, or if the client's incremental chunk list's last chunk is full chunk.
    - Case 2.1: the block does not have any full chunks yet.
        - the block's chunk is what received from client this time.
    - Case 2.2: the block already has some full chunks.
        - Append the incremental chunk lists to the block's chunk list.
- Case 3: (client's incremental chunk list has partial chunks)
    - Case 3.1 client's incremental chunk list has only one chunk, which is partial
        - Replace the existing last partial chunk table.
    - Case 3.2: the client's incremental chunk list has more than one chunks, and if the block does not exist in the block data table.
        - Add the full chunks to block data table;
        - Add the last chunk to last chunk table.
    - Case 3.3: the client's incremental chunk list has more than one chunks, and if the block exists in the block data table,
        -
-

—

Append to a rocksdb key is not efficient.

(1) Read the key and deserialize.
(2) Append the in-memory structure.
(3) Write back and serialize.

Create a “LastChunk” rocksdb table.
<chunk id (block id + “_chunk_” + chunk id)> → ChunkInfo.

- LastChunk table is updated for every flush.
- When the chunk is written full (i.e. 4MB in size), update the BlockDataTable: read the list of ChunkInfo, append the list, and write back.

Extend GetBlock, ListBlock, BlockDeletingService
(Anything that touches getBlockDataTable())
Concerns: there will be discrepancies. Blocks that are deleted, but last chunk table is not cleaned up.

GetBlock

4 cases

(1) Block table entry is empty and last chunk table entry is empty
    (a) Throw exception because the block doesn’t exist
(2) Block table entry is empty and last chunk table entry exists
    (a) The block is composed of a single partial chunk; return the entry in the last chunk table.
(3) Block table entry exists and last chunk table entry is empty
    (a) The block is composed of multiple full chunks. Return it.
(4) Block table entry exists and last chunk table entry exists
    (a) Reconcile: Append the last partial chunk to the block table entry.

Tasks

(1) Test case
(2) DataNode change
(3) Client change
(4) Fix tools
(5) Increase default Ratis log segment size HDDS-8040
(6) Review PutBlock/WriteChunk transaction serialized size