cloudera

HBase Tales From the Trenches

Wellington Chevreuil

Agenda

- Common types of problems
- Most affected features
- Common reasons
- Case stories review
- General best practices

Common types of problems

- RegionServers/Master crashes
- Performance
- Regions stuck in transition (RIT)
- FileSystem usage
- Corruption / Data loss / Replication data consistency

Most affected features or sub-systems

- AssignmentManager
- Replication
- Snapshot
- Region Server Memory Management
- Master Initialisation
- WAL/StoreFile
- Custom applications / filters / co-processors

Common reasons

- Performance
  - Overload/under-dimensioned cluster
  - Non optimal configurations
- Crashes
  - Memory exhaustion
  - File system issues
  - Known bugs
- RIT
  - Bugs
  - Self induced (wrong hbck commands triggered)

Common reasons

- RIT (cont.)
  - Can also happen as side effect of performance, crash or corruption issues
- File system usage exhaustion
  - Replication related issues
  - Too many snapshots
- Corruption / Data loss / Replication data consistency
  - Bugs
  - Faulty peers / custom or third party components
  - FileSystem problems

Case Stories Review

Case story - RegionServers slow/crashing randomly

Type: Process crash | Service outage | Performance

Feature: Region Server Memory Management

Reason: Long GC pauses, due to mismatching heap sizes and workloads

Diagnosing:

- Frequent JvmPauseMonitor alerts on RSes logs
- Occasionally OOME on stdout
- Too many regions per RS (more than 200 regions)
- JVM Heap usage charts show wide heap usage (JVisualVM/Jconsole)

Resolution:

- Initially increase the heap size, but CMS may experience slowness with large heaps
- For heaps larger than 20GB, general G1 recommendations from Cloudera engineering blog post had provided good results
- Horizontal scaling by adding more RSes

Case story - Slow scans, compactions delayed

Type: Performance

Feature: Store File

Reason: PrefixTree HFile encoding issues (HBASE-17375)

Diagnosing:

- Compaction queue piling up
- jstacks from RSes show below trace over several frames:

```
"regionserver/hadoop30-r5.phx.impactradius.net/10.16.20.138:60020-longCompactions-1550194360449" #111 prio=5 os_prio=0
tid=0x00007fcb429b3800 nid=0x243a8 runnable [0x00007fc3481c7000]
java.lang.Thread.State: RUNNABLE
at org.apache.hadoop.hbase.codec.prefixtree.decode.PrefixTreeArrayScanner.advance(PrefixTreeArrayScanner.java:214)
at org.apache.hadoop.hbase.codec.prefixtree.PrefixTreeSeeker.next(PrefixTreeSeeker.java:127)
at org.apache.hadoop.hbase.io.hfile.HFileReaderV2$EncodedScannerV2.next(HFileReaderV2.java:1278)
at org.apache.hadoop.hbase.regionserver.StoreFileScanner.next(StoreFileScanner.java:181)
at org.apache.hadoop.hbase.regionserver.KeyValueHeap.next(KeyValueHeap.java:108)
at org.apache.hadoop.hbase.regionserver.StoreScanner.next(StoreScanner.java:628)
```

Resolution: Disable table, manual compact it with CompactionTool, disable PrefixTree encoding

Mitigation: Disable PrefixTree encoding (Not supported anymore from 2.0 onwards)

Case story - My HBase is slow

Type: Performance

Feature: Client Application Read/Write operations

Reason:

- Dependency services underperforming
- Poor client implementation not reusing connections
- Faulty CPs or custom filters

Diagnosing:

- Client application/RSes jstacks
- General HBase stats such as: compaction queue size, data locality, cache hit ratio
- HDFS/ZK logs

Resolution: Usually require tunings on dependency services or redesign of client application/custom CPs/Filters

Case story - Client scans failing | HBCK reports inconsistencies

Type: RIT

Feature: AssignmentManager

Reason: Various

- Misuse of hbck
- Snapshots cold backups out of hdfs
- Busy/overloaded clusters where regions keep moving constantly

Diagnosing:

- Evident from hbck reports/Master Web UI
- Master logs would show RS opening/hosting region
- RS holding region should have relevant error message logs

Resolution:

- There's no single recipe
- Each case may require a combination of hbck/hbck2 commands

Case story - Master timing out during initialisation

Type: Master Crash | Service Outage

Feature: Master initialisation

Reason: Different bugs can cause procedures to pile up:

- HBASE-22263, HBASE-16488, HBASE-18109

Diagnosing:

- Listing "/hbase/MasterProcWALs" shows hundreds or more files
- Master times out and crashes before assigning namespace region

```
ERROR org.apache.hadoop.hbase.master.HMaster: Master failed to complete initialization after
```

Resolution:

- Stop Master and clean "/hbase/MasterProcWALs" folder
- Caution when on hbase > 2.x releases

Mitigation:

- Increase init timeout, number of open region threads

Case story - Replication lags

Type: Replication stuck

Feature: Replication

Reason: Single WAL entries with too many OPs, leading to RPCs larger than "hbase.ipc.server.max.callqueue.size"

Diagnosing: Destination peer RSes showing type of log messages below

```
2018-09-07 10:40:59,506 WARN org.apache.hadoop.hbase.client.AsyncProcess: #690, table=MY_TABLE, attempt=4/4 failed=2ops,
last exception: org.apache.hadoop.hbase.ipc.RemoteWithExtrasException(org.apache.hadoop.hbase.CallQueueTooBigException): Call
queue is full on /0.0.0.0:60020, is hbase.ipc.server.max.callqueue.size too small? on
region-server-1.example.com,,60020,1524334173359, tracking started Fri Sep 07 10:35:53 IST 2018; not retrying 2 - final failure
2018-09-07 10:40:59,506 ERROR org.apache.hadoop.hbase.replication.regionserver.ReplicationSink: Unable to accept edit because:
```

Resolution: requires wal copy and replay from source to destination, plus manual znode cleanup

Mitigation: releases including HBASE-18027 would prevent this situation

Case story - Client scan failing on specific regions

Type: HFile Corruption

Feature: Store File

Reason: Unknown

Diagnosing: Following errors when scanning specific regions

```
java.lang.InternalError: Could not decompress data. Input is invalid.
    at org.apache.hadoop.io.compress.snappy.SnappyDecompressor.decompressBytesDirect(Native Method)
    at org.apache.hadoop.io.compress.snappy.SnappyDecompressor.decompress(SnappyDecompressor.java:239)
```

Or

```
org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:163) Caused by: java.lang.NegativeArraySizeException at
org.apache.hadoop.hbase.io.hfile.HFileBlock$FSReaderImpl.readBlockDataInternal(HFileBlock.java:1718) at
org.apache.hadoop.hbase.io.hfile.HFileBlock$FSReaderImpl.readBlockData(HFileBlock.java:1542) at
```

Resolution: Requires sideline of affecting files and re-ingestion of row keys stored on this file. Potential data loss

Case story - HBase is eating HDFS space

Type: FileSystem usage

Feature: Replication | Snapshot | Compaction | Cleaners

Reason: Various

- Replication: Slow, faulty or disabled peer, missing tables on remote peer
- Snapshot: Too many snapshots being retained
- Compaction and Cleaners threads stuck or not running

Diagnosing:

- Check usage for "archive" and "oldWALS"
- Master logs would show if cleaner threads are running
- Is replication stuck or lagging?
- How about snapshot retention policy?

Resolution:

- If cleaner threads are not running, restart master
- For disabled peers, only enabling it again, or remove it, if no replication is wanted
- Too many snapshots would require some cleaning or cold backup
- Replication lags reason may vary

General best practices

- Heap usage monitoring
- Keep regions/RS on low hundreds
- Consider G1 GC for heaps > 20GB
- Data locality
- Adjust caching according to workload
- Compaction Policy (Consider offline compactions using CompactionTool)
- Consider an exclusive Zookeeper for HBase

General best practices

- Adjust Master initialization timeout accordingly
- Consider increase number of "open region" handlers
- Define reasonable snapshot retention policy
- Caution with experimental/non stable features (snappy/prefixtree)
- Define deployment policy for custom applications/CPs/Filters
- Define patch/bug fix upgrades schedule

Q&A