ApacheCon Europe

Sinsheim, Germany 5th–8th November 2012

cloudera

HBASE STATUS QUO

The State of Affairs in HBase Land

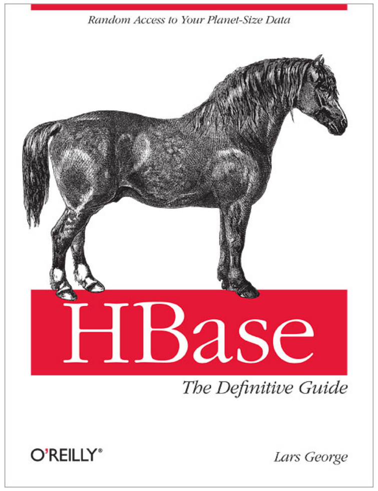
ApacheCon Europe, November 2012

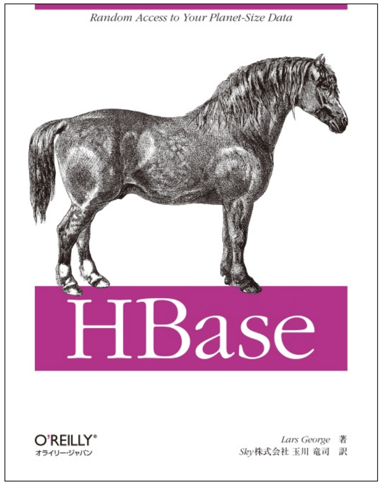
Lars George

Director EMEA Services

About Me

- Director EMEA Services @ Cloudera
  - Consulting on Hadoop projects (everywhere)
- Apache Committer
  - HBase and Whirr
- O'Reilly Author
  - HBase – The Definitive Guide
    - Now in Japanese!
- Contact
  - lars@cloudera.com
  - @larsgeorge





日本語版も出ました!

Agenda

- HDFS and HBase
- HBase Project Status

cloudera

HDFS AND HBASE

Past, Presence, Future

Framework for Discussion

- Time Periods
  - Past (Hadoop pre-1.0)
  - Present (Hadoop 1.x, 2.0)
  - Future (Hadoop 2.x and later)

- Categories
  - Reliability/Availability
  - Performance
  - Feature Set

HDFS and HBase History – 2006

```
Author: Douglass Cutting <cutting@apache.org>
Date: Fri Jan 27 22:19:42 2006 +0000

   Create hadoop sub-project.
```

HDFS and HBase History – 2007

```
Author: Douglass Cutting <cutting@apache.org>
Date: Tue Apr 3 20:34:28 2007 +0000

   HADOOP-1045. Add contrib/hbase, a
   BigTable-like online database.
```

HDFS and HBase History – 2008

```
Author: Jim Kellerman <jimk@apache.org>
Date: Tue Feb 5 02:36:26 2008 +0000

    2008/02/04 HBase is now a subproject of
    Hadoop. The first HBase release as a
    subproject will be release 0.1.0 which will
    be equivalent to the version of HBase
    included in Hadoop 0.16.0...
```

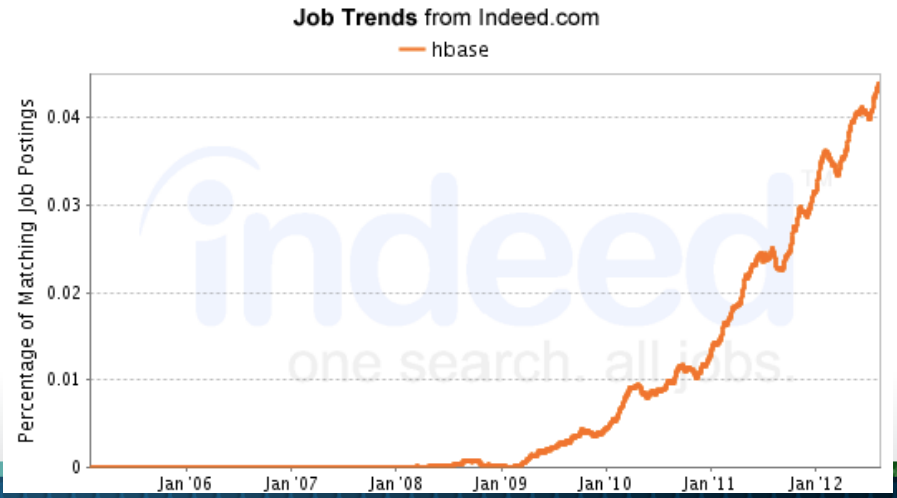
HDFS and HBase History – Early 2010

HBase has been around for 3 years. But HDFS still acts like MapReduce is the only important client!

People have accused HDFS of being like a molasses train:
High throughput but not so fast

HDFS and HBase History – 2010

- HBase becomes a top-level project
- Facebook chooses HBase for Messages product
- Jump from HBase 0.20 to HBase 0.89 and 0.90
- First CDH3 betas include HBase
- HDFS community starts to work on features for HBase.
  - Infamous `hadoop-0.20-append` branch



WHAT DID GET DONE?

And where is it going?

Reliability in the Past: Hadoop 1.0

- Pre-1.0, if the DN crashed, HBase would lose its WALs (and your beloved data).
  - 1.0 integrated `hadoop-0.20-append` branch into a main-line release
  - **True durability support** for HBase
  - We have a fighting chance at metadata reliability!
- Numerous bug fixes for write pipeline recovery and other error paths
  - HBase is not nearly so forgiving as MapReduce!
  - “Single-writer” fault tolerance vs. “job-level” fault tolerance

Reliability in the Past: Hadoop 1.0

- Pre-1.0: if any disk failed, entire DN would go offline
  - Problematic for HBase: local RS would lose all locality!
  - 1.0: per-disk failure detection in DN (HDFS-457)
  - Allows HBase to lose a disk without losing all locality

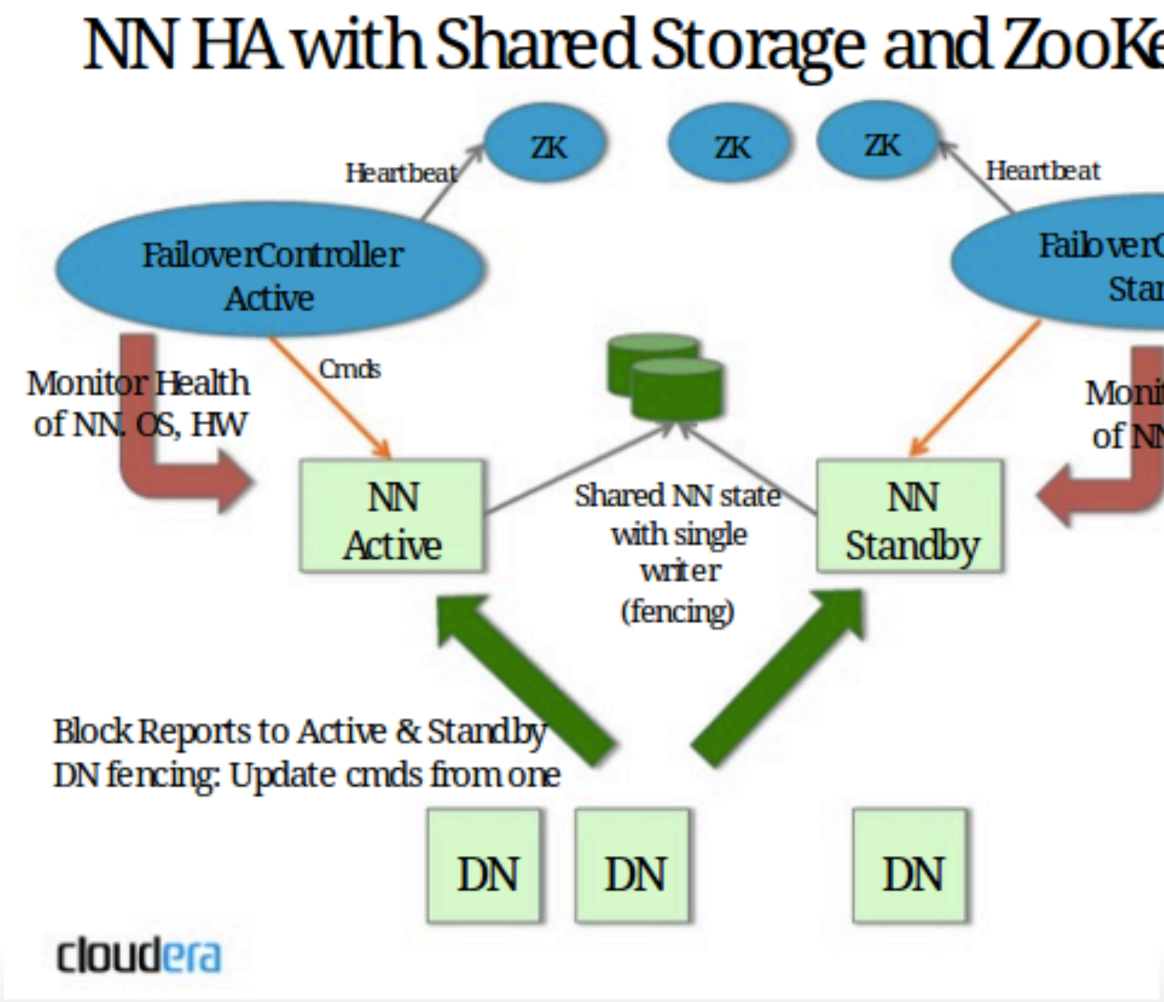
- **Tip**: Configure

```
dfs.datanode.failed.volumes.tolerated = 1
```

Reliability Today: Hadoop 2.0

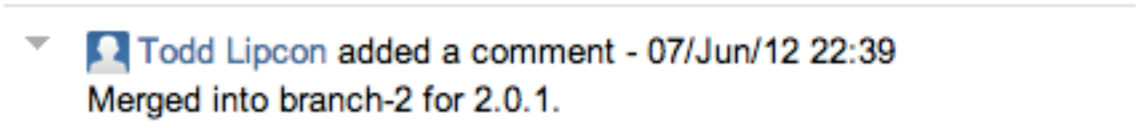
- Integrates **Highly Available HDFS**
- Active-standby hot failover removes SPOF
- Transparent to clients: no HBase changes necessary
- Tested extensively under HBase read/write workloads
- Coupled with HBase master failover, **no more HBase SPOF!**

HDFS HA


NN HA with Shared Storage and ZooKeeper
ZK
ZK
ZK
Heartbeat
Heartbeat
FailoverController Active
FailoverController Standby
Monitor Health of NN. OS, HW
Cmds
Monitor Health of NN. OS, HW
NN Active
Shared NN state with single writer (fencing)
NN Standby
Block Reports to Active & Standby
DN fencing: Update cmds from one
DN
DN
DN
cloudera

Reliability in the Future: HA in 2.x

- Remove dependency on NFS (HDFS-3077)
  - Quorum-commit protocol for NameNode edit logs
  - Similar to ZAB/Multi-Paxos
- Automatic failover for HA NameNodes (HDFS-3042)
  - ZooKeeper-based master election, just like HBase
  - Merged to trunk

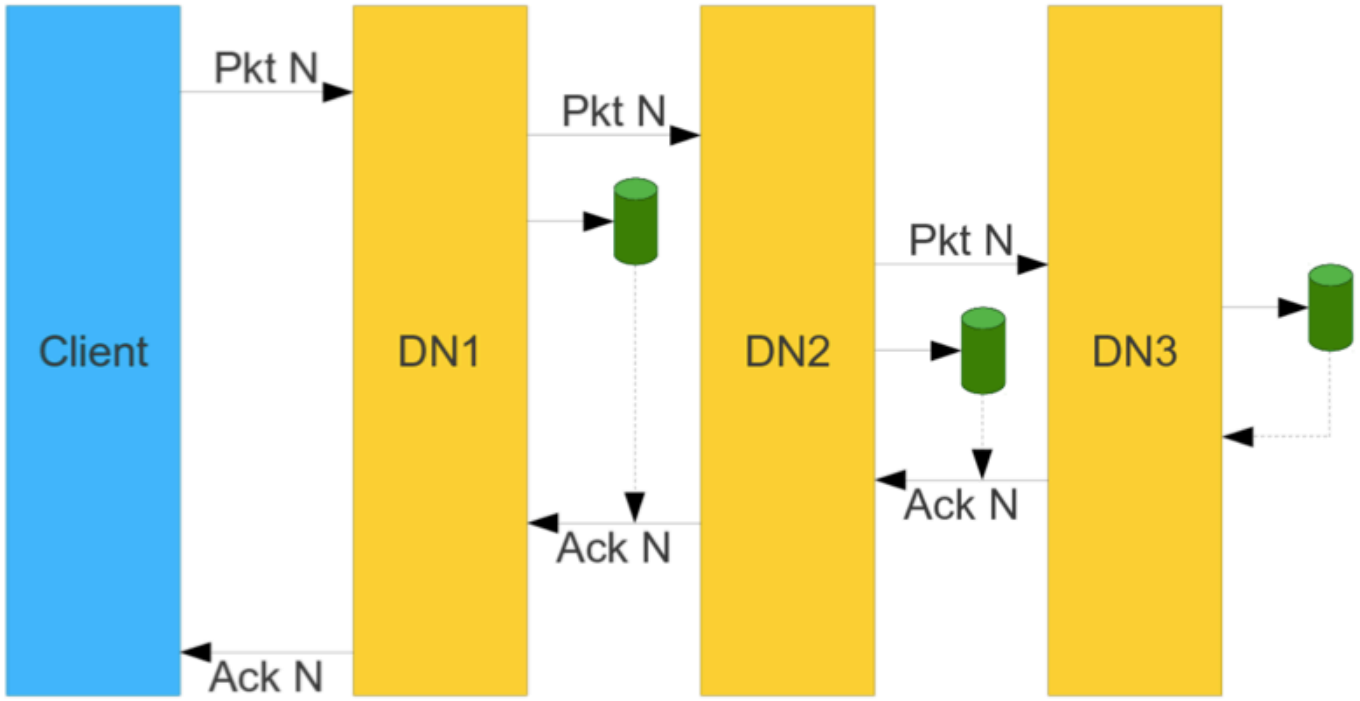


Todd Lipcon added a comment - 07/Jun/12 22:39
Merged into branch-2 for 2.0.1.

Other Reliability Work for HDFS 2.x

- 2.0: current `hflush()` API only guarantees data is replicated to three machines – not fully on disk.
- A cluster-wide power outage can lose data.
    - Upcoming in 2.x: Support for `hsync()` (HDFS-744, HBASE-5954)
    - Calls `fsync()` for all replicas of the WAL
    - **Full durability of edits**, even with full cluster power outages

hflush() and hsync()



hflush()/hsync()

- send all queued data, note seq num
- block until corresponding ACK is received

hflush(): flushes to OS buffer

hsync(): fsync() to disk

HDFS Wire Compatibility in Hadoop 2.0

- In 1.0: HDFS client version must match server version closely.
- How many of you have manually copied HDFS client jars?
- Client-server compatibility in 2.0:
  - Protobuf-based RPC
  - Easier HBase installs: **no more futzing with jars**
  - **Separate HBase upgrades from HDFS upgrades**
- Intra-cluster server compatibility in the works
  - Allow for **rolling upgrade without downtime**

Performance: Hadoop 1.0

- Pre-1.0: even for reads from local machine, client connects to DN via TCP
- 1.0: Short-circuit local reads
  - Obtains direct access to underlying local block file, then uses regular `FileInputStream` access.
  - **2x speedup for random reads**
- Configure

```
dfs.client.read.shortcircuit = true
dfs.block.local-path-access.user = hbase
dfs.datanode.data.dir.perm = 755
```

- Note: Currently does not support security

Performance: Hadoop 2.0

- Pre-2.0: Up to 50% CPU spent verifying CRC
- 2.0: Native checksums using SSE4.2 `crc32 asm` (HDFS-2080)
    - **2.5x speedup reading from buffer cache**
    - Now only 15% CPU overhead to checksumming
- Pre-2.0: re-establishes TCP connection to DN for each seek
- 2.0: Rewritten `BlockReader`, keep-alive to DN (HDFS-941)
    - 40% improvement on random read for HBase
    - **2-2.5x in micro-benchmarks**
- Total improvement vs. 0.20.2: **3.4x!**

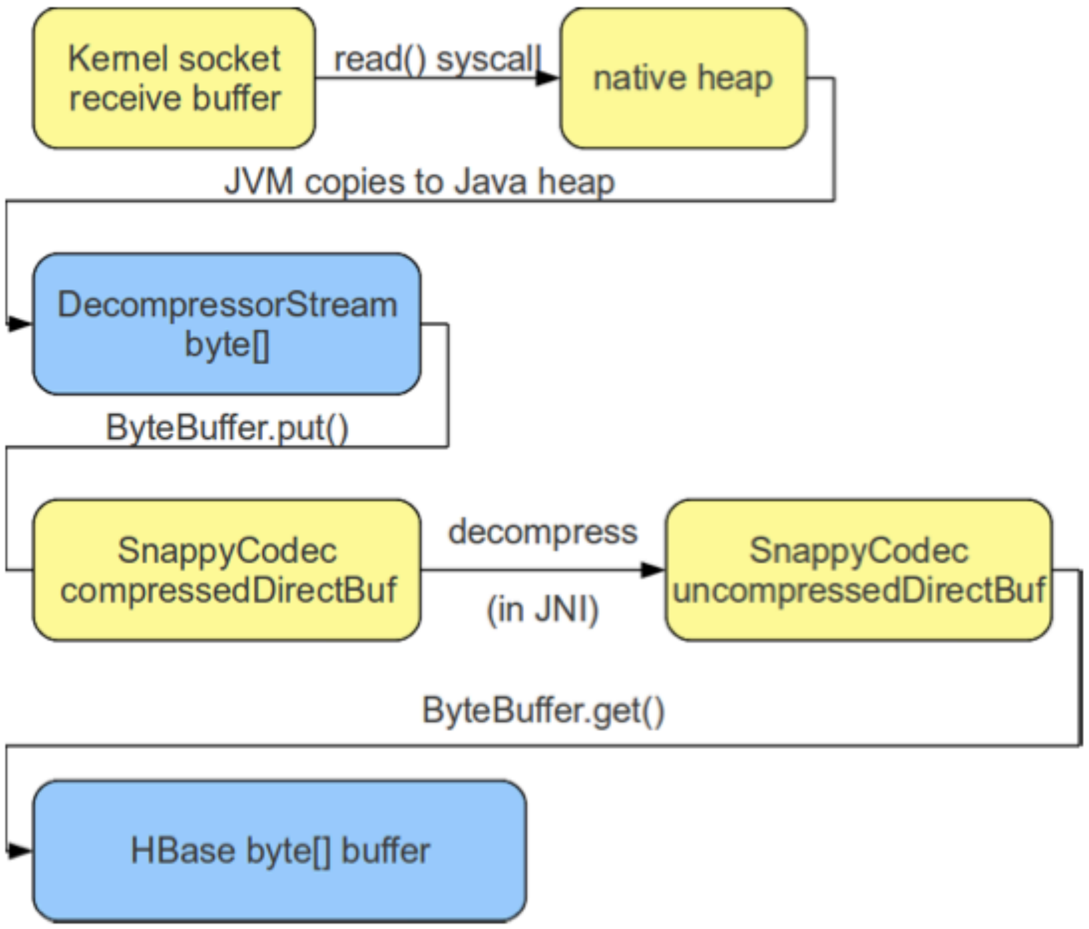
Performance: Hadoop 2.x

- Currently: lots of CPU spent copying data in memory
- "Direct-read" API: read directly into user-provided DirectByteBuffers (HDFS-2834)
  - **Another ~2x improvement** to sequential throughput reading from cache
  - Opportunity to avoid two more buffer copies reading compressed data (HADOOP-8148)
  - Codec APIs still in progress, needs integration into HBase

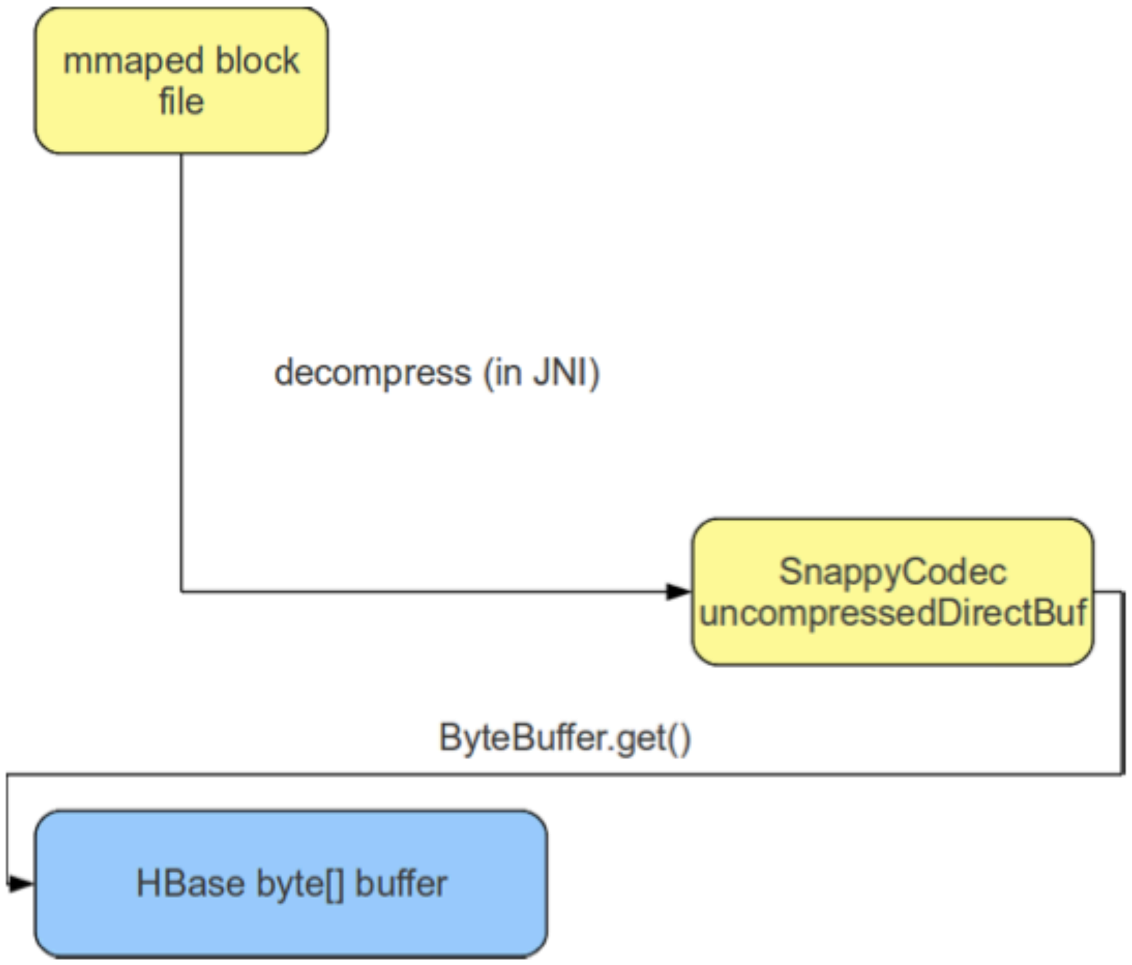
Performance: Hadoop 2.x

- True "zero-copy read" support (HDFS-3051)
  - New API would allow direct access to `mmaped` block files
  - No syscall or JNI overhead for reads
  - Initial benchmarks indicate at least ~30% gain.
  - Some open questions around best safe implementation

Current Read Path


Kernel socket receive buffer
read() syscall
native heap
JVM copies to Java heap
DecompressorStream byte[]
ByteBuffer.put()
SnappyCodec compressedDirectBuf
decompress
(in JNI)
SnappyCodec uncompressedDirectBuf
ByteBuffer.get()
HBase byte[] buffer

Proposed Read Path



Performance: Why Emphasize CPU?

- Machines with lots of RAM now inexpensive (48-96GB common)
- Want to use that to improve cache hit ratios.
- Unfortunately, 50GB+ Java heaps still impractical (GC pauses too long)
- Allocate the extra RAM to the buffer cache
    - OS caches *compressed* data: another win!
- CPU overhead reading from buffer cache becomes limiting factor for read workloads

What's Up Next in 2.x?

- HDFS Hard-links (HDFS-3370)
  - Will allow for HBase to clone/snapshot tables efficiently!
  - Improves HBase table-scoped backup story
- HDFS Snapshots (HDFS-2802)
  - HBase-wide snapshot support for point-in-time recovery
  - Enables consistent backups copied off-site for DR

What’s Up Next in 2.x?

- Improved block placement policies (HDFS-1094)
  - Fundamental tradeoff between probability of data unavailability and the amount of data that becomes unavailable
  - Current scheme: if any 3 nodes not on the same rack die, some very small amount of data is unavailable
  - Proposed scheme: lessen chances of unavailability, but if a certain three nodes die, a larger amount is unavailable
  - For many HBase applications: any single lost block halts whole operation. Prefer to minimize probability.

What's Up Next in 2.x?

- HBase-specific block placement hints (HBASE-4755)
  - Assign each region a set of three RS (primary and two backups)
  - Place underlying data blocks on these three DNs
  - Could then fail-over and load-balance without losing any locality!

Summary

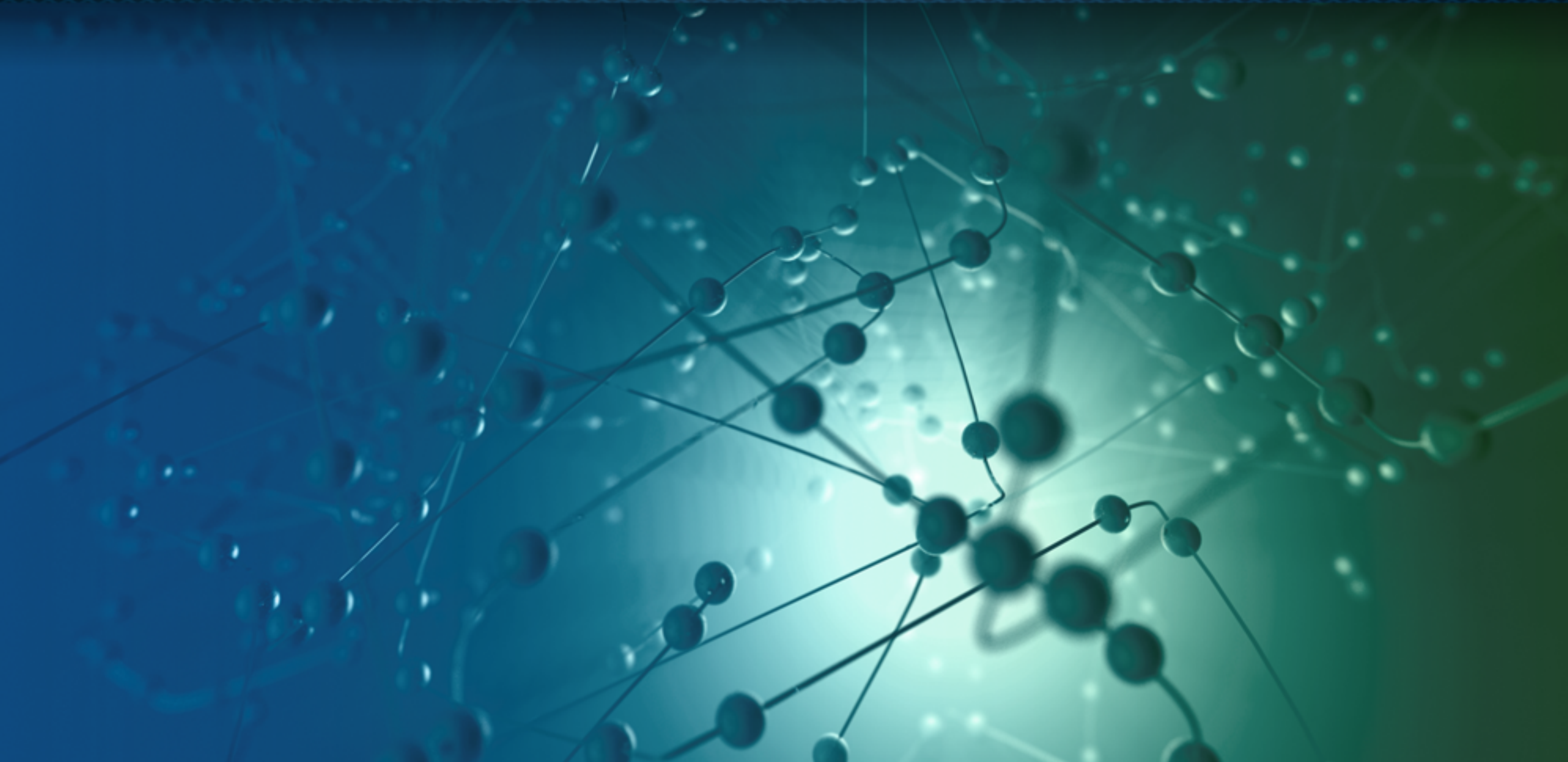
	Hadoop 1.0	Hadoop 2.0	Hadoop 2.x
Availability	• DN volume failure isolation	• NameNode HA • Wire Compatibility	• HA without NAS • Rolling upgrade
Performance	• Short-circuit reads	• Native CRC • DN keep-alive	• Direct-read API • Zero-copy API • Direct codec API
Features	• Durable `hflush()`		• `hsync()` • Snapshots • Hard links • HBase-aware block placement

Summary

- HBase is no longer a second-class citizen.
- We've come a long way since Hadoop 0.20.2 in performance, reliability, and availability.
- New features coming in the 2.x line specifically to benefit HBase use cases
- Hadoop 2.0 features available today via CDH4. Many Cloudera customers already using CDH4 with HBase with great success.

cloudera

PROJECT STATUS



Current Project Status

- HBase 0.90.x “Advanced Concepts”
  - Master Rewrite – More Zookeeper
  - Intra Row Scanning
  - Further optimizations on algorithms and data structures

➡ CDH3

Current Project Status

- HBase 0.92.x “Coprocessors”
  - Multi-DC Replication
  - Discretionary Access Control
  - Coprocessors
    - Endpoints and Observers
    - Can hook into many explicit and implicit operation

➡ CDH4

Current Project Status (cont.)

- HBase 0.94.x “Performance Release”
  - Read CRC Improvements
  - Seek Optimizations
    - Lazy Seeks
  - WAL Compression
  - Prefix Compression (aka Block Encoding)

  - Atomic Append
  - Atomic put+delete
  - Multi Increment and Multi Append

Current Project Status (cont.)

- HBase 0.94.x “Performance Release”
  - Per-region (i.e. local) Multi-Row Transactions

```
RegionMutation rm = new RegionMutation();
Put p = new Put(ROW1);
p.add(FAMILY, QUALIFIER, VALUE);
rm.add(p);
p = new Put(ROW2);
p.add(FAMILY, QUALIFIER, VALUE);
rm.add(p);
t.mutateRegion(rm);
```

Current Project Status (cont.)

- HBase 0.94.x “Performance Release”
  - Uber HBCK
  - Embedded Thrift Server
  - WALPlayer
  - Enable/disable Replication Streams

➡ CDH4.x (soon)

Current Project Status (cont.)

- HBase 0.96.x “The Singularity”
    - Protobuf RPC
        - Rolling Upgrades
        - Multiversion Access
    - Metrics V2
    - Preview Technologies
        - Snapshots
        - PrefixTrie Block Encoding

➡ CDH5 ?

Client/Server Compatibility Matrix

RegionServer, Master	Client 0.96.0	Client 0.96.1	Client 0.98.0	Client 1.0.0
0.96.0	Works	Works*	Works*	No guarantee
0.96.1	Works	Works	Works*	No guarantee
0.98.0	Works	Works	Works	Works*
1.0.0	No guarantee	No guarantee	Works	Works

Notes: * If new features are not used

Questions?