SDC 18
September 24-27, 2018
Santa Clara, CA

www.storagedeveloper.org

Unlock BDaaS efficiency with storage disaggregation and in-memory acceleration on All Flash systems

Jian Zhang (jian.zhang@intel.com), Senior Software Engineer Manager
Yuan Zhou (yuan.zhou@intel.com), Senior Software Engineer

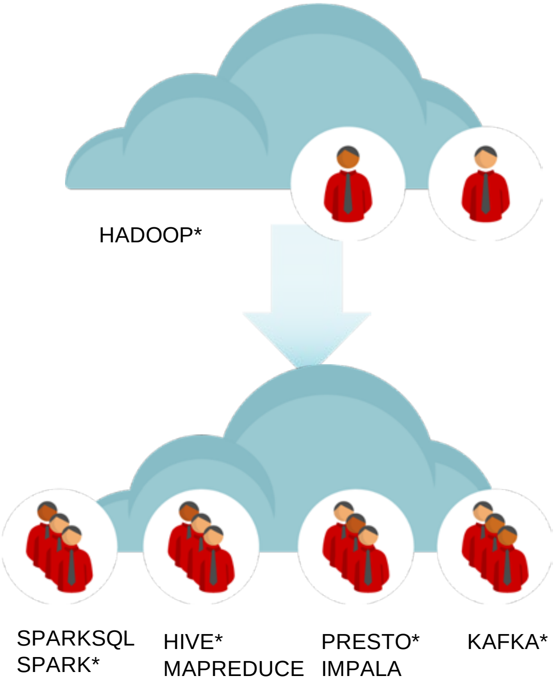
Intel APAC R&D

Agenda

- Background and Motivations
- Disaggregated Analytics Architecture
- Performance of Disaggregated Analytics Solutions
- In-memory Acceleration for Disaggregated Analytics Architecture
- Summary

Background and Motivation

Discontinuity in bigdata infrastructure – why?



CONGESTION
in busy analytic clusters
causing missed SLAs.

MULTIPLE TEAMS COMPETING
and sharing the same
big data resources.

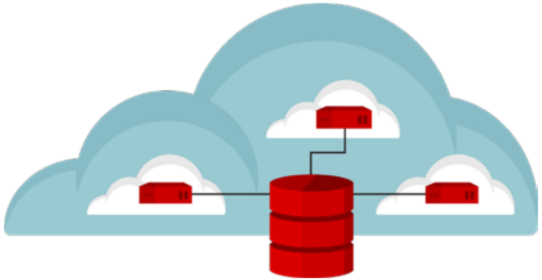
Causing customers to pick a solution

SINGLE LARGE CLUSTER

Get a bigger cluster
for many teams to share.

MULTIPLE SMALL CLUSTERS

Give each team their
own dedicated cluster,
each with a copy of
PBs of data.



ON DEMAND ANALYTIC CLUSTERS

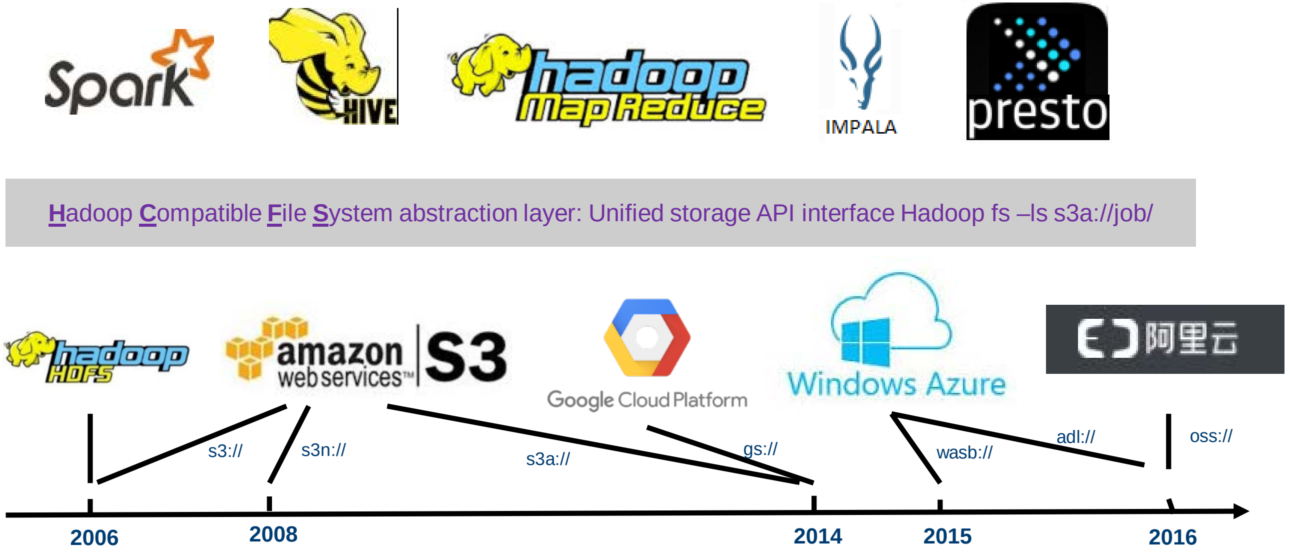
Give teams ability to
spin-up/spin-down
clusters which can
share data sets.

Benefits of compute and storage disaggregation

Independent scale of CPU and storage capacity	Single copy of data	Enable Agile application development	Hybrid cloud deployment	Simple and flexible software management
• Rightsize HW for each layer • Reduce resource wastage • Cost saving	• Multiple compute cluster share common data repo/lake • Simplified data management • Reduced provisioning overhead • Improve security	• In-memory cloning • Snapshot service • Quick & efficient copies	• Mix and match resources depending on workload nature and life cycle	• Avoid software version management • Upgrade compute software only

Disaggregated Analytics Architecture

Unified Hadoop File System and API for cloud storage



Evolution of AI/Analytics Solutions

GENERATION 1
MONOLITHIC HADOOP STACKS

cloudera
HORTONWORKS
POWERING THE FUTURE OF DATA
MAPR

ANALYTICS + INFRASTRUCTURE

GENERATION 2
DECOUPLED STACK WITH PRIVATE CLOUD INFRASTRUCTURE

Batch | Streaming | Interactive | Graph Analytics | Machine Leaning

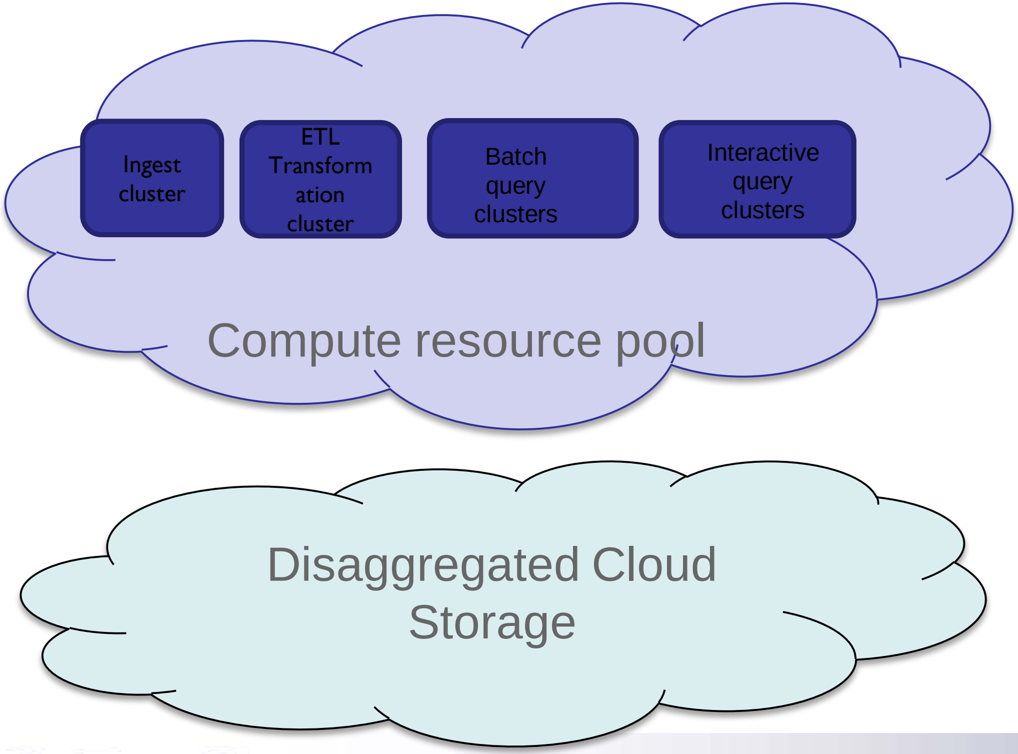
cloudera, HORTONWORKS, MAPR, HIVE, Apache Spark, presto

Provisioned Compute Pool

Shared Data Lake

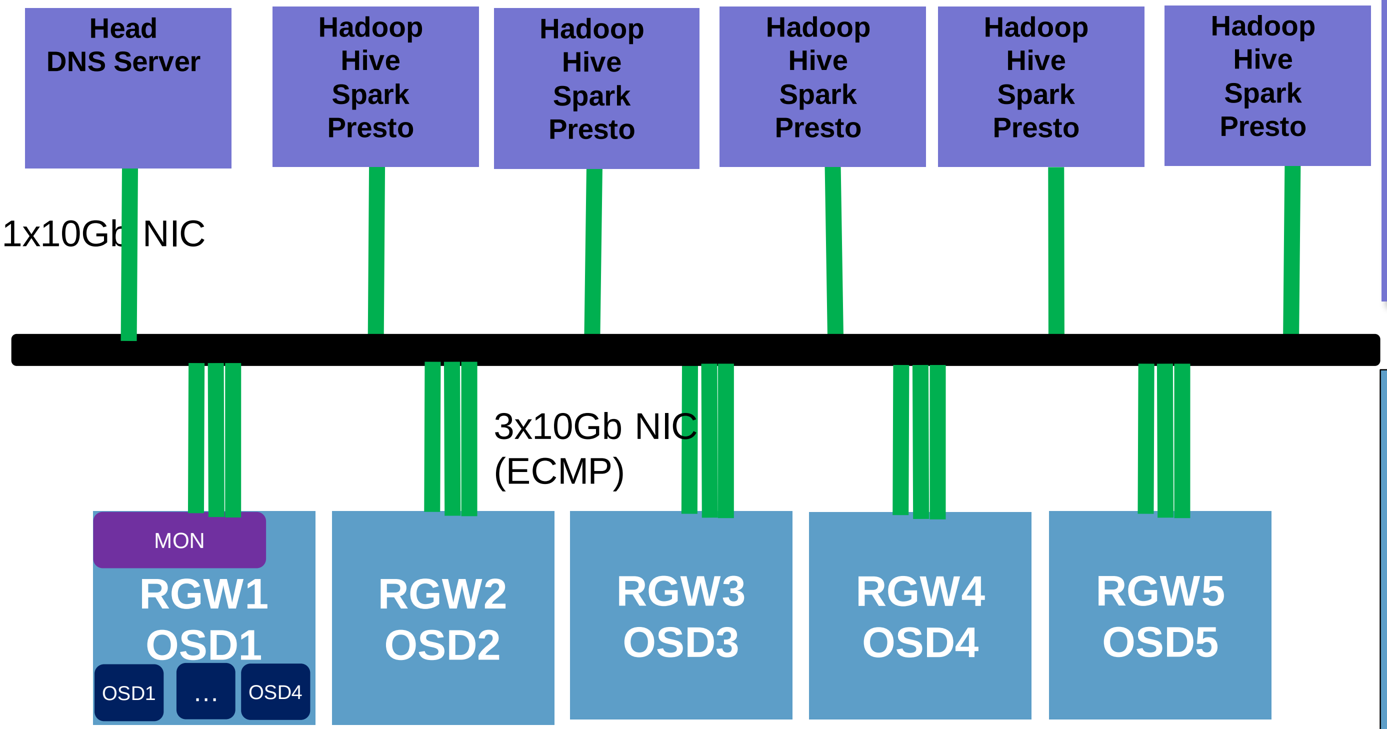
Performance of Disaggregated Analytics Solutions

Workloads



- **Simple Read/Write**
  - DFSIO: TestDFSIO is the canonical example of a benchmark that attempts to measure the Storage's capacity for reading and writing bulk data.
  - Terasort: a popular benchmark that measures the amount of time to sort one terabyte of randomly distributed data on a given computer system.
- **Batch ingestion**
  - Support collection of data from a variety of data sources in a consistent and repeatable manner designed to reduce data loss, improve traceability, increase availability, and increase timeliness.
- **Data Transformation**
  - ETL: Taking data as it is originally generated and transforming it to a format (Parquet, ORC) that more tuned for analytical workloads.
- **Batch Analytics**
  - To consistently executing analytical process to process large set of data.
  - Leveraging 54 derived from TPC-DS* queries with intensive reads across objects in different buckets
- **Interactive Query**
  - This is very similar to the batch analytics workload, with the key distinction being required response time.
- **Streaming**
  - streaming data collection is the landing and aggregation of streaming data from messaging queues.

System Configuration

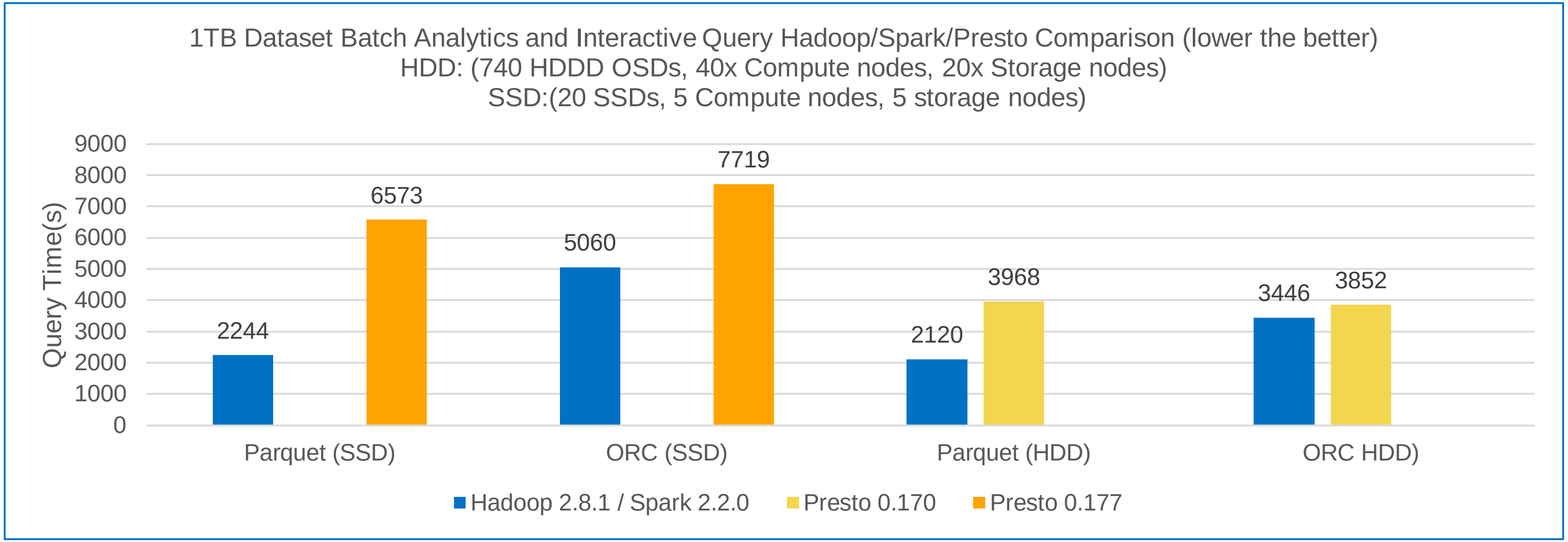


5x Compute Node
- Intel® Xeon™ processor E5-2699 v4 @ 2.2GHz, 128GB mem
- 2x10G 82599 10Gb NIC
- 2x SSDs
- 3x Data storage (can be emliminated)

Software:
- Hadoop 2.7.3
- Spark 2.1.1
- Hive 2.2.1
- Presto 0.177
- RHEL7.3

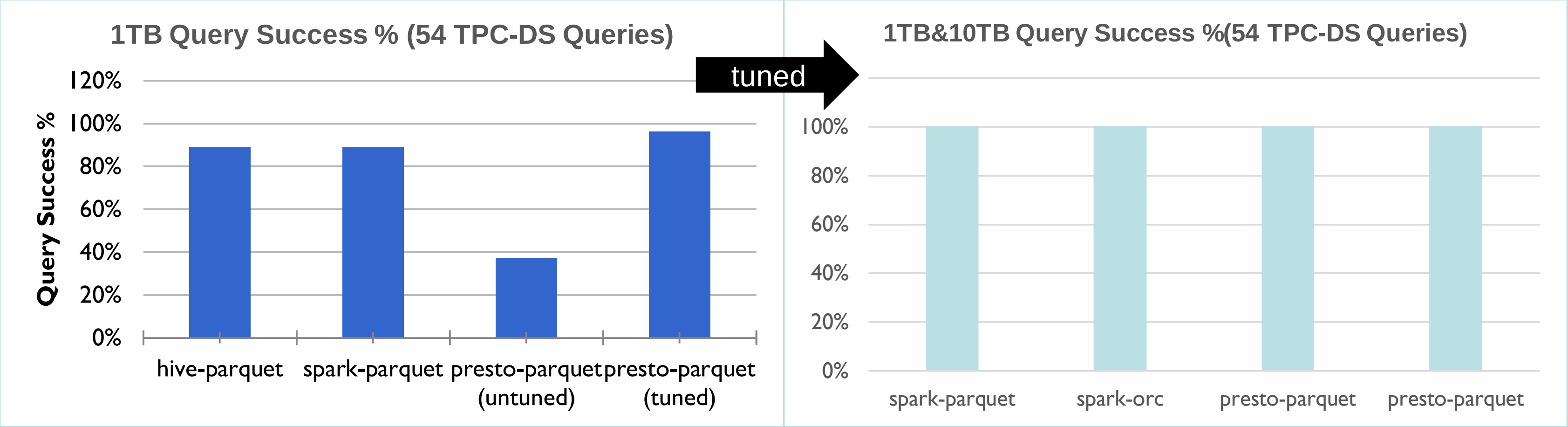
5x Storage Node, 2 RGW nodes, 1 LB nodes
- Intel(R) Xeon(R) CPU E5-2699v4 2.20GHz
- 128GB Memory
- 2x 82599 10Gb NIC
- 1x Intel® P3700 1.0TB SSD as WAL and rocksdb
- 4x 1.6TB Intel® SSD DC S3510 as data drive
- 2x 400G S3700 SSDs
- 1 OSD instances one each S3510 SSD
- RHEl7.3
- RHCS 2.3

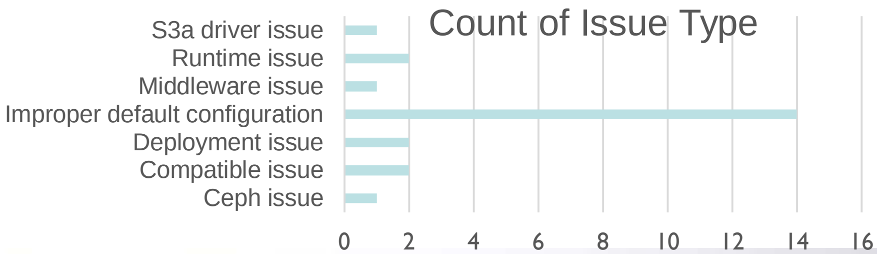
Bigdata on Object Storage Performance Overview
-- batch analytics



- Significant performance improvement from Hadoop 2.7.3/Spark 2.1.1 to Hadoop 2.8.1/Spark 2.2.0 (improvement in s3a)
- Batch analytics performance of 10-node Intel AFA is almost on-par with 60-node HDD cluster

Improve Query Success Ratio
-- Functional Trouble-shooting



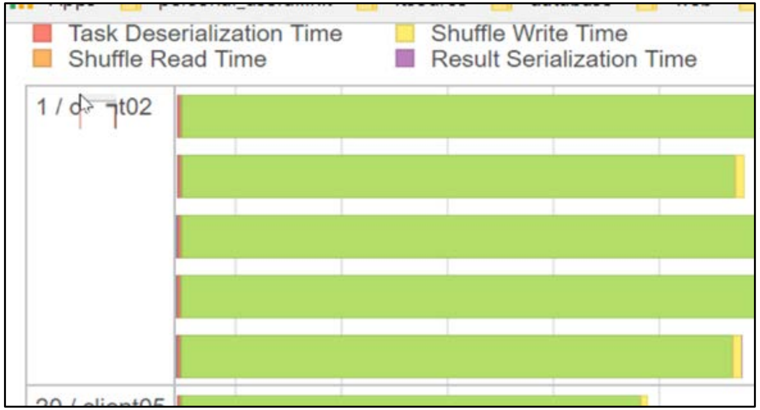


- 100% selected TPC-DS* query passed with tunings
- Improper Default configuration
  - small capacity size,
  - wrong middleware configuration
  - improper Hadoop/Spark configuration for different size and format data issues

Optimizing HTTP Requests
-- The bottlenecks

```
2017-07-18 14:53:52.259976 7fddd67fc700  1 ====== starting new request req=0x7fddd67f6710 =====
2017-07-18 14:53:52.271829 7fddd5ffb700  1 ====== starting new request req=0x7fddd5ff5710 =====
2017-07-18 14:53:52.273940 7fddd7fff700  0 ERROR: flush_read_list(): d->client_c->handle_data() returned -
5
2017-07-18 14:53:52.274223 7fddd7fff700  0 WARNING: set_req_state_err err_no=5 resorting to 500
2017-07-18 14:53:52.274253 7fddd7fff700  0 ERROR: s->cio->send_content_length() returned err=-5
2017-07-18 14:53:52.274257 7fddd7fff700  0 ERROR: s->cio->print() returned err=-5
2017-07-18 14:53:52.274258 7fddd7fff700  0 ERROR: STREAM_IO(s)->print() returned err=-5
2017-07-18 14:53:52.274267 7fddd7fff700  0 ERROR: STREAM_IO(s)->complete_header() returned err=-5
```

Http 500 errors in RGW log

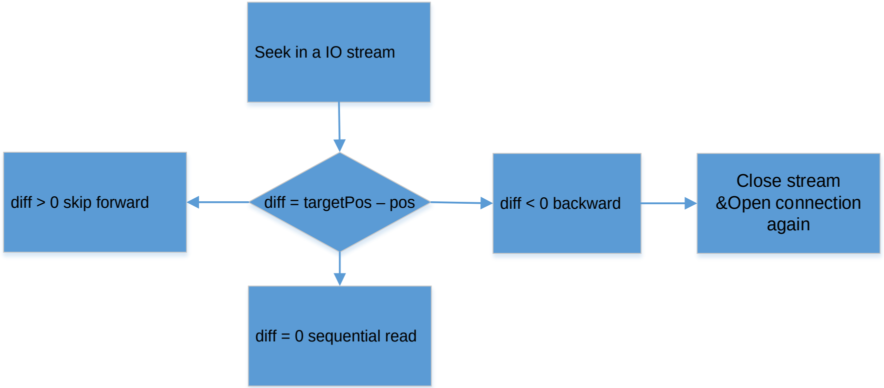


Compute time take the big part.
(compute time = read data +sort)

New connections out every time,
Connection not reused

```
ESTAB   0      0       ::ffff:10.0.2.36:44446    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44454    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44374    ::ffff:10.0.2.254:80
ESTAB   159724 0       ::ffff:10.0.2.36:44436    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44448    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44338    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44438    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44414    ::ffff:10.0.2.254:80
ESTAB   0      480     ::ffff:10.0.2.36:44450    ::ffff:10.0.2.254:80
timer:(on,170ms,0)
ESTAB   0      0       ::ffff:10.0.2.36:44442    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44390    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44326    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44452    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44394    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44444    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44456    ::ffff:10.0.2.254:80
2 seconds interval ======================
ESTAB   0      0       ::ffff:10.0.2.36:44508    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44476    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44524    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44374    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44500    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44504    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44512    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44506    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44464    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44518    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44510    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44442    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44526    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44472    ::ffff:10.0.2.254:80
ESTAB   0      0       ::ffff:10.0.2.36:44466    ::ffff:10.0.2.254:80
```

Optimizing HTTP Requests
-- S3a input policy



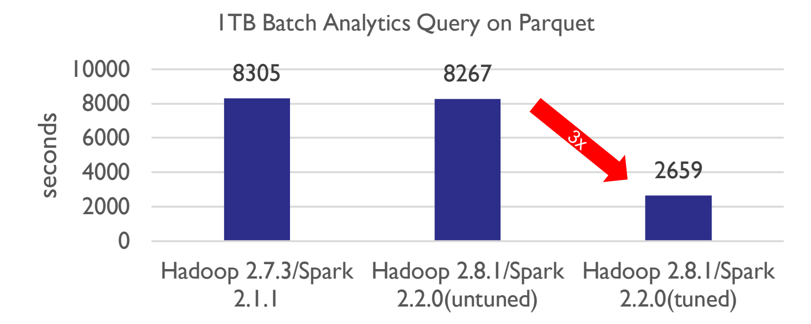
- Background
  - The S3A filesystem client supports the notion of input policies, similar to that of the POSIX fadvise() API call. This tunes the behavior of the S3A client to optimize HTTP GET requests for various use cases. To optimize HTTP GET requests, you can take advantage of the S3A experimental input policy fs.s3a.experimental.input.fadvise.
  - Ticket: https://issues.apache.org/jira/browse/HADOOP-13203

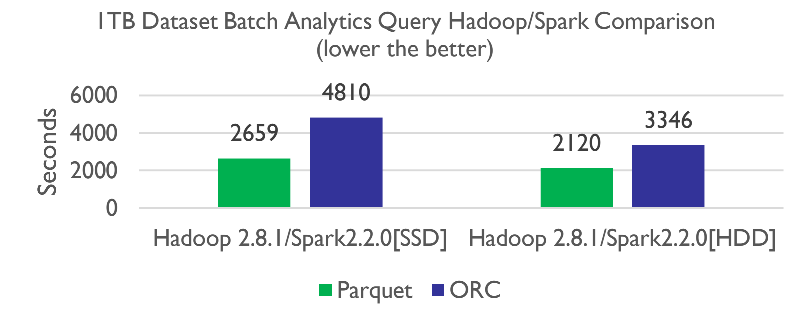
- Enable random read policy hadoop:

```
<property>
   <name>fs.s3a.experimental.input.fadvise</name>
   <value>random</value>
</property>
<property>
   <name>fs.s3a.readahead.range</name>
   <value>64K</value>
</property>
```

- By reducing the cost of closing existing HTTP requests, this is highly efficient for file IO accessing a binary file through a series of `PositionedReadable.read()` and `PositionedReadable.readFully()` calls.

Optimizing HTTP Requests
-- Performance

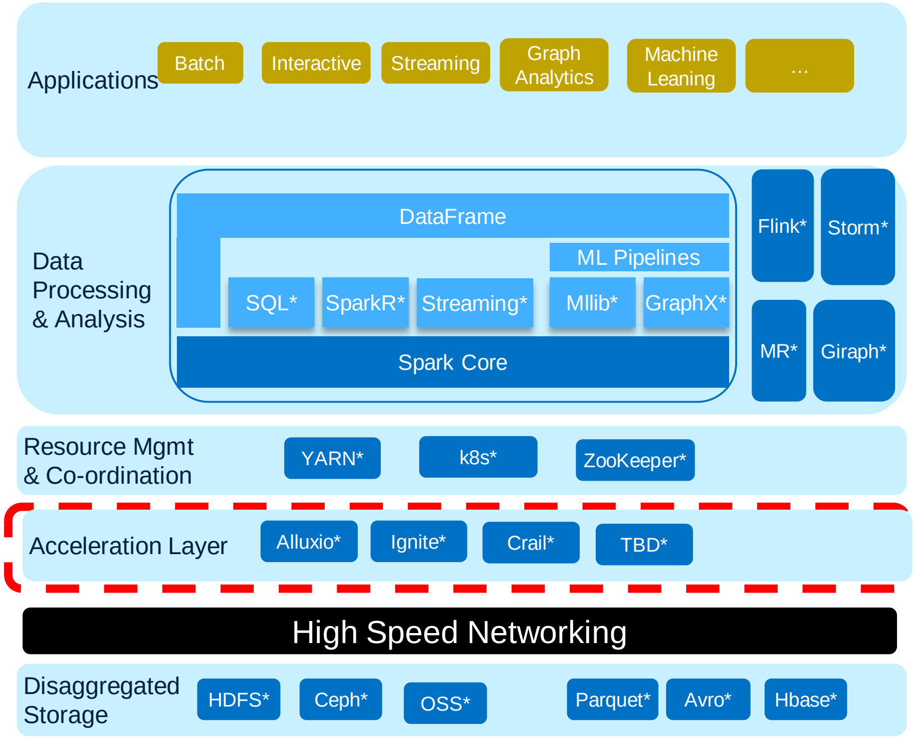




- Readahead feature is supported from Hadoop 2.8.1, but not enabled by default. By applying random read policy, the 500 issue is fixed and performance improved 3x
- All Flash storage architecture also show great performance benefit and low TCO which compared with HDD storage

In-memory Acceleration for Disaggregated Analytics Architecture

Acceleration Layer For Disaggregated Architecture

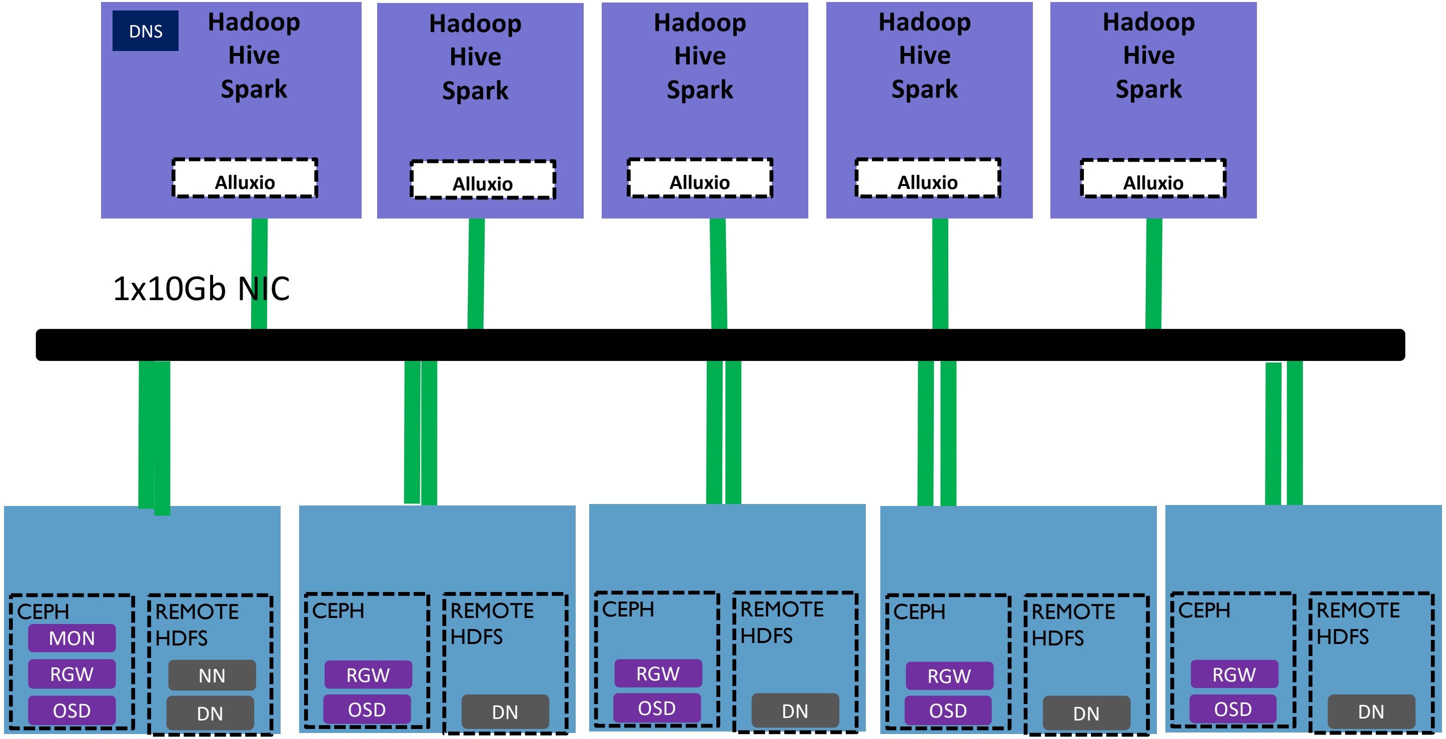


- With disaggregated storage, the networking overhead of I/O will be bigger
  - Network is going to be a bottleneck
- Bigdata I/O stack is still using technology 10+ years ago
  - New technology like SPDK, RDMA has not been used
- We want to implement a new storage layer for intermediate data like cache/shuffle to accelerate the data access

In-memory Accelerator POC with Alluxio*

- The world's first system that unifies disparate storage systems at memory speed
  - HDFS, Blob, S3, GCS, Minio, Ceph, Swift, MapR-FS to name a few
- Accelerates Cloud Deployments for Analytics and Machine Learning
  - Location aware data management; optimized for object storage and every* major CSP; User Space File system enables ML frameworks to access cloud data
- Trusted by the world's leading companies.
  - Alibaba, Baidu, CMU, Google, IBM, Intel, NJU, Red Hat, UC Berkeley and Yahoo, JD.com etc.

POC System Configuration



5x Compute Node

Hardware:

- ntel® Xeon™ processor Gold 6140 @ 2.3GHz, 384GB Memory
- 1x 82599 10Gb NIC
- 5x P4500 SSD (2 for spark-shuffle)

Software:

- Hadoop 2.8.1
- Spark 2.2.0
- Hive 2.2.1
- RHEL7.3

Alluxio Acceleration Layer

- 200GB Mem for mem mode
- 1TB SSD(P4500) for SSD mode

Software:

- Alluxio 1.7.0

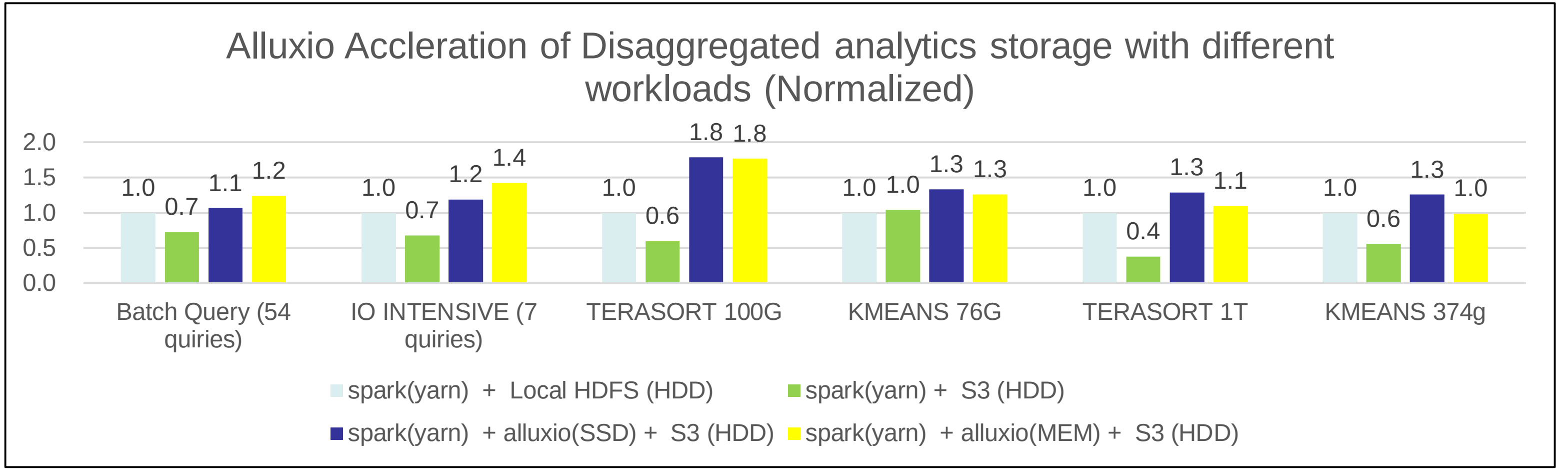
5x Storage Node

- Intel(R) Xeon(R) CPU Gold 6140 @ 2.30GHz, 192GB Memory
- 2x 82599 10Gb NIC
- 7x 1TB HDD for Ceph bluestore or HDFS namenode and datanode

Software:

- Hadoop 2.8.1
- Ceph 12.2.7
- RHEL7.3

POC Performance with Alluxio*



- Alluxio based in memory acceleration layers provides significant performance boost for analytics workloads with disaggregated storage
  - Up to 3.25x for Terasort
  - Up to 1.8x compared with local HDFS

Summary

Summary

- Discontinuity in bigdata infrastructure drives storage disaggregation, Decoupling compute and storage brings cost savings, agility and flexibility and becoming increasing popular in public CSPs
- Cloud based Bigdata and Analytics grows much faster than on-premise solutions, public cloud adoption is No.1 priority for Bigdata investments
- BDA on Cloud performance is much lower compared with on-premise
- In memory acceleration layer helps to deliver higher performance and enables new usage scenario;
  - Little or no interruptions to AI/Analytics processing
  - Memory Locality Optimizations via Compute-Side Caching
  - Optimizes both performance and TCO for AI/Analytical workload
  - Alluxio based POC demonstrates up to 3.25x performance boost for bigdata analytics workloads on s3 cloud storage

Legal Disclaimer & Optimization Notice

- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.
- INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.
-

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Backup

Architecture involvement