WHITEPAPER

Improve Hadoop Performance with Memblaze® PBlaze SSD

Exclusive Summary

We live in the data age. It’s not easy to measure the total volume of data stored electronically, but an IDC estimate put the size of the “digital universe” at 4.4 zettabytes in 2013 and is forecasting a tenfold growth by 2020 to 44 zettabytes. So how to organize those sheer volume of data being generated every year effectively? Processing massive amounts of data requires a parallel compute and storage infrastructure, thus the parallel processing, scalable and reliable ability which Hadoop provides.

This whitepaper demonstrates the performance of Hadoop is improved with Memblaze PBlaze SSD by up to 4.5x over HDD.

Introduction

Hadoop utilizes the powerful Hadoop Distributed File System (HDFS) which is a highly scalable, parallel file system optimized for very large sequential data sets running on clusters of commodity hardware. CPU and memory can process hundred GB data per second, but disk IO can only support 100MB to several GB read/write per second with HDD. Disk IO is the bottleneck of the system. Large data generated every day, as amount of data grows, processing the data needs longer and longer time. How to save time? Solve disk IO bottleneck issue, improve HDFS performance. Add more disks to each node? May be the node don’t have more space for disk. Even there are several hard drive slot, the performance is not good enough. Add more nodes, may be data center don’t have more space. Add more nodes speed too much. Don’t worry. Memblaze PCIe SSD help you solve it. It offers preferable performance with less nodes. Our test show 3 data nodes, each with one SSD performance is better than 7 data nodes, each with 6 HDDs.

Test Cluster Configuration

The Hadoop distributed file system (HDFS) is a distributed, scalable, and portable file-system written in Java for the Hadoop framework. A Hadoop cluster has nominally a single name node plus a cluster of data nodes. Each data node serves up blocks of data over the network using a block protocol specific to HDFS. The file system uses TCP/IP sockets for communication. Clients use remote procedure call (RPC) to communicate between each other.

The test cluster consists of 1 master node running NameNode, ResourceManager and client function, and 3 data nodes configured as following Figure 1 illustrated:

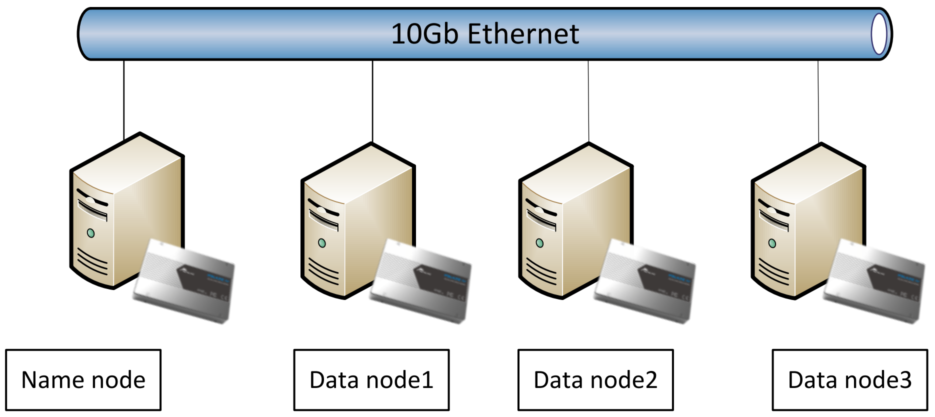

Figure 1 Test Cluster Topology

CPU:	Dell PowerEdge R730x 2 socket Intel XeonE5-2630(8 cores) v3
Memory:	128GB
SSD:	1 x Memblaze 3.2T PBlaze4
Network:	Intel 82599ES 10-Gigabit
Linux:	CentOS 7.0
File system:	xt4
Java:	1.7.0_75
Hadoop:	hadoop-2.6.3
Benchmark Tool:	hadoop-mapreduce-client-jobclient-2.6.3-tests.jar TestDFSIO

Following is the list of configuration parameters used in benchmark:

General parameters

Map tasks: 12

Reduce tasks: 12

HDFS

Block size: 128 MB

Replication factor: 3

Mapreduce

Java child options: -Xmx 2048m

Io sort buffer size: 512MB

Benchmark procedure

The test process aims at evaluate HDFS performance with PCIe SSD.

TestDFSIO is used to measure performance of HDFS and stress both network and IO subsystems. The command read and write files in HDFS which is useful in measuring system-wide performance and exposing network bottlenecks on the NameNode and DataNodes. A majority of MapReduce workloads are IO bound more than compute and hence TestDFSIO can provide an accurate initial picture of such scenarios.

The benchmark can be run for writing, using the –write switch, and using –read for the read test. The command line accepts a number of files and sizes of each file in HDFS. As an example, the command for a read test may look like:

```
hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-client-jobclient-2.6.3-tests.jar  TestDFSIO
-write -nrFiles 1  -size 240GB
```

TestDFSIO generates 1 map task per file and splits are defined such that each map gets only one file name. After every run, the command generates a log file indicating performance in terms of 4 metrics: Throughput in MBytes/s, Average IO rate in MBytes/s, IO rate standard deviation and execution time. The most notable metrics are throughput and average IO, both of which are based on file size read or written by the individual map task and the elapsed time in performing the task. The throughput for N map tasks are defined as:

$$Throughput\ (N) = \frac{\sum_{i=0}^{N} size\ of\ file_i}{\sum_{i=0}^{N} execution\ time_i}$$

And average IO rate is calculated as:

$$Average\ IO\ Rate\ (N) = \frac{\sum_{i=0}^{N} rate_i}{N} = \frac{\sum_{i=0}^{N} \frac{size\ of\ file_i}{time_i}}{N}$$

If the cluster has 50 map slots and TestDFSIO creates 1000 files, the throughput can be calculated as:

$$Concurrent\ throughput = Reported\ throughput * Number\ of\ Map\ Slots$$

The IO rate can be calculated in similar fashion. While measuring cluster performance using TestDFSIO may be considered sufficient, the HDFS replication factor (value of dfs.replication in hdfs-site.xml) also plays important role. A lower replication factor leads to higher throughput performance due to reduced background traffic [1].

Test Results

Our test function run on the Namenode. Illustrated in Figure 2, total write 240GB data, replicate 3, file number N with 1, 3, 6, 12, 48, file size M, each file with one map. 1 map can get best Throughput 144MB/s, 12 and 24 maps can get best concurrent throughput 576 MB/s.

Because replicate 3, HDFS write 576 * 3 = 1728MB/s to 3 data node, that is 576 MB/s each data node.

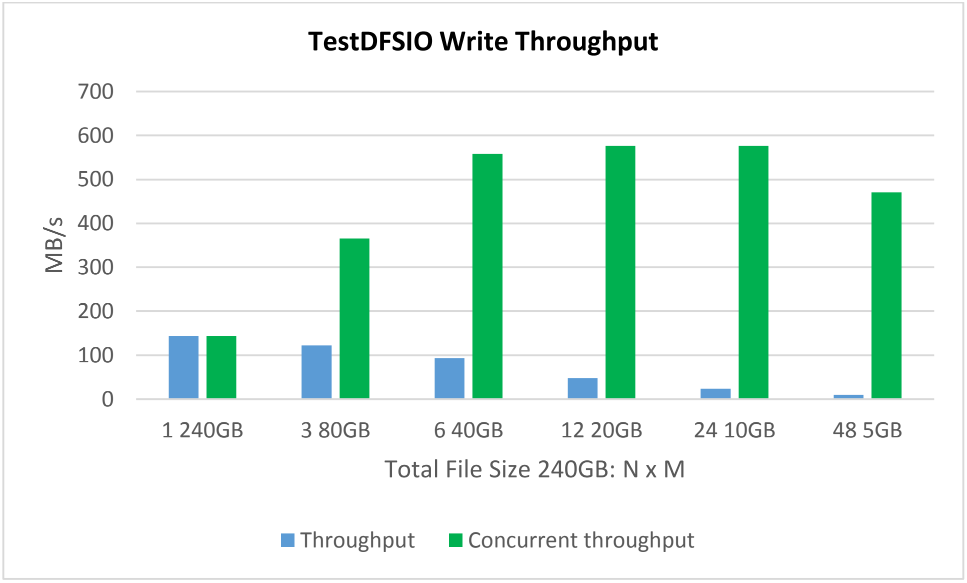

Figure 2 TestDFSIO Write Throughput

Illustrated in Figure 3, total read 240GB data, replicate 3, file number N with 1, 3, 6, 12, 48, file size M, each file with one map. 1 map can get best throughput 357MB/s, 48 maps can get best concurrent throughput 8208MB/s.

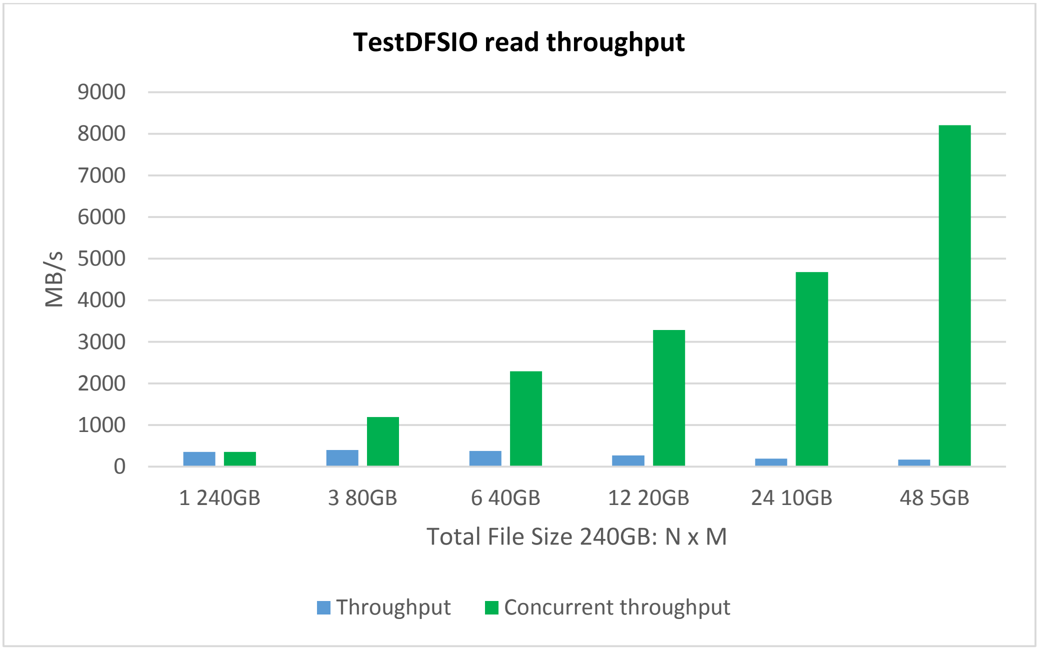

Figure 3 TestDFSIO Read Throughput

Emulex tested Hadoop with 7 Data node each 6 HDD, the performance test result from test report *Emulex, Performance measurement of a Hadoop Cluster.* Our benchmark result shows 3 data nodes performance much better than 7 data nodes HDD cluster as shown in Figure 4.

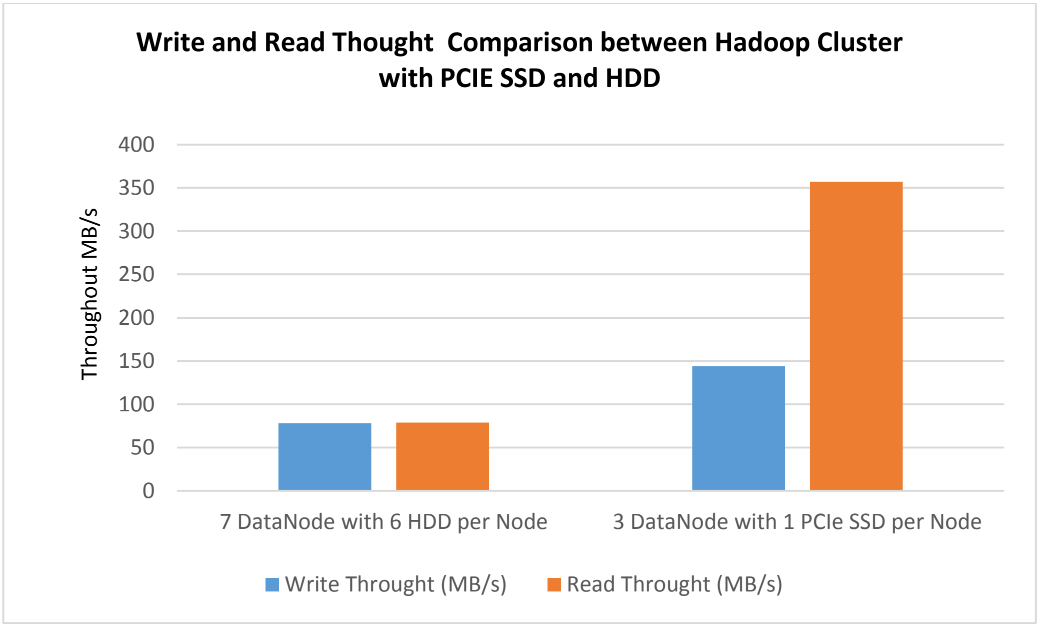

Figure 4 Write and Read Thought Comparison between Hadoop Cluster with PCIE SSD and HDD

Conclusions

As the amount of data grows, the most efficiency solution to improve Hadoop IO performance is utilizes PCIe SSD instead of HDD. From the white paper it shows clearly that Hadoop cluster greatly benefits from innovative Memblaze PCIe SSD and offers optimum performance, in the meantime, save more nodes and with TCO comparable to HDDs.

Reference

[1] Emulex, Performance measurement of a Hadoop Cluster