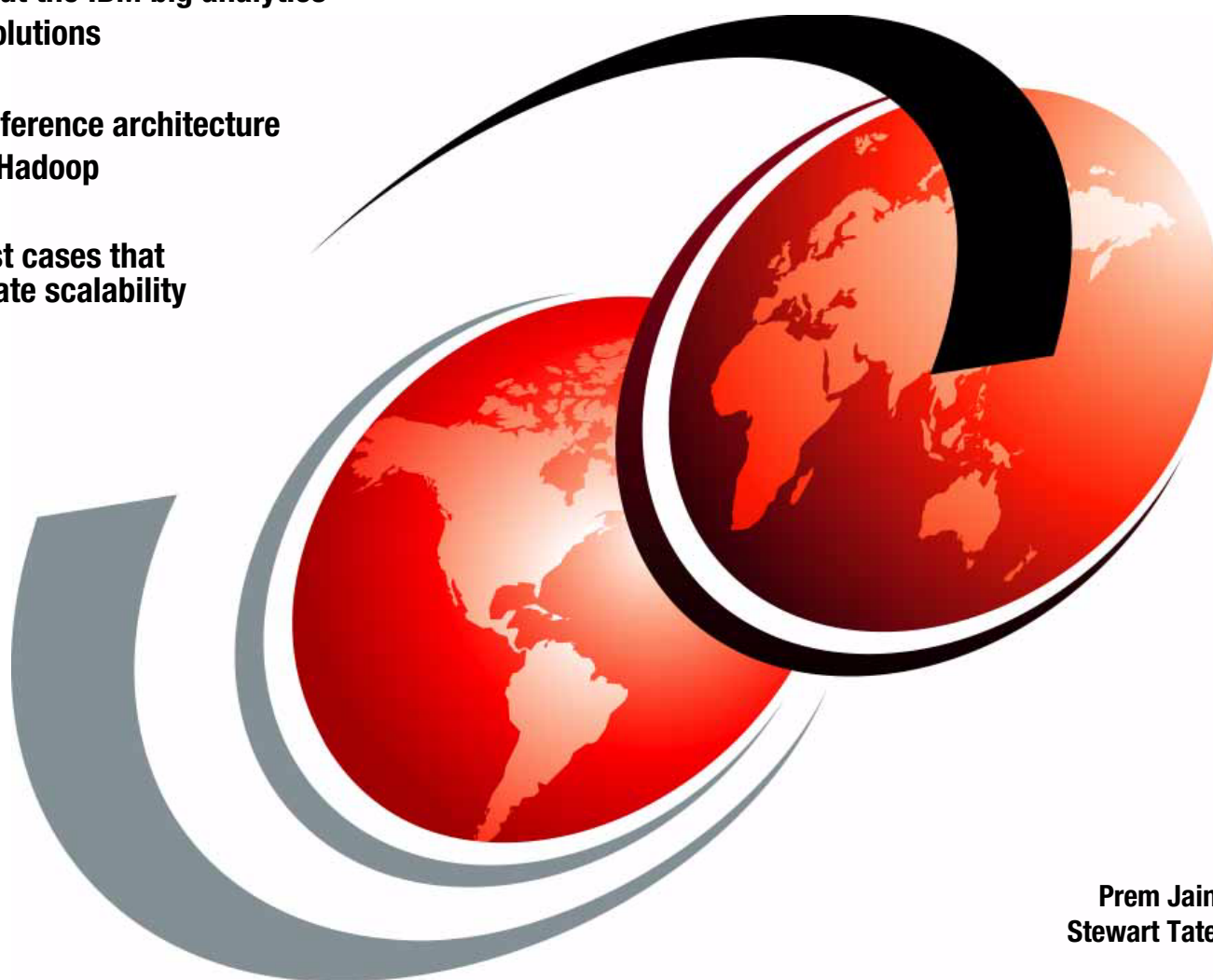


# Big Data Networked Storage Solution for Hadoop

Learn about the IBM big analytics storage solutions

Explore reference architecture based on Hadoop

Follow test cases that demonstrate scalability



Prem Jain  
Stewart Tate





International Technical Support Organization

**Big Data Networked Storage Solution for Hadoop**

June 2013

**Note:** Before using this information and the product it supports, read the information in “Notices” on page v.

### **First Edition (June 2013)**

This edition applies to IBM InfoSphere BigInsights version 2.0, IBM System Storage DCS3700 with controller firmware level 7.8, DS Storage Manager version 10.8 and IBM N Series N3220 with Data ONTAP Operating System version 8.0.2 (operating in 7-mode).

**© Copyright International Business Machines Corporation 2013. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	v
Trademarks .....	vi
<b>Preface</b> .....	vii
Authors .....	vii
Now you can become a published author, too! .....	viii
Comments welcome .....	viii
Stay connected to IBM Redbooks .....	ix
<b>Chapter 1. Introduction and solution summary</b> .....	1
1.1 Meeting the challenge of big data analytics with IBM InfoSphere BigInsights .....	2
1.1.1 The value of the IBM InfoSphere BigInsights analytics .....	2
1.1.2 Analysis and discovery .....	2
1.1.3 Enterprise software integration .....	2
1.2 Hadoop Distributed File System .....	3
1.3 Triple-mirror trade-offs .....	5
1.4 When drives fail .....	6
1.5 Reference architecture for big data analytics .....	7
1.6 Benefits of the Big Data Networked Storage Solution for Hadoop .....	8
1.7 Solution summary .....	9
1.7.1 Data protection and job completion .....	9
<b>Chapter 2. Solution architecture</b> .....	11
2.1 Hardware requirements .....	11
2.1.1 Servers .....	11
2.1.2 Storage .....	11
2.2 Software requirements .....	12
2.3 Solution architecture details .....	12
2.3.1 Network architecture .....	12
2.3.2 BigInsights cluster network .....	13
2.3.3 Network switch .....	13
2.3.4 Cabling topology .....	14
2.3.5 Storage architecture .....	15
2.3.6 Storage sizing and performance considerations .....	16
2.3.7 NameNode metadata protection .....	16
2.3.8 Rack-awareness implementation .....	17
<b>Chapter 3. Big Data Networked Storage Solution for Hadoop reference architecture tests</b> .....	19
3.1 Solution objectives .....	21
3.2 Solution testing .....	21
3.2.1 Test case 1 .....	22
3.2.2 Test case 2 .....	22
3.2.3 Test case 3 .....	22
3.3 Test environment configurations .....	22
3.3.1 DCS3700 storage arrays .....	22
3.3.2 Hadoop servers .....	24
3.3.3 Local file system configuration .....	24
3.3.4 XFS file system mount options (/etc/fstab entries) .....	25

3.3.5 IBM N series N3220 storage . . . . .	25
3.3.6 Mount options to be used on NameNode and secondary NameNode servers . . .	25
3.3.7 Linux kernel and device tuning . . . . .	26
3.3.8 Linux SAS control depth settings with procedure . . . . .	28
3.3.9 Hadoop parameter settings . . . . .	30
3.4 Test software inventory . . . . .	33
3.5 Solution test results . . . . .	33
<b>Chapter 4. Conclusions and details of test cases . . . . .</b>	<b>35</b>
4.1 Test case details . . . . .	36
4.1.1 Test case 1-A: Initial tuning . . . . .	36
4.1.2 Test case 1-B: Full functionality as baseline . . . . .	36
4.1.3 Test case 2: Full functionality with fault injection . . . . .	37
4.1.4 Test case 3: NameNode metadata recovery . . . . .	38
<b>Related publications . . . . .</b>	<b>41</b>
IBM Redbooks . . . . .	41
Other publications . . . . .	41
Online resources . . . . .	41
Help from IBM . . . . .	42

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

## COPYRIGHT LICENSE:


This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

BigInsights™  
BNT®  
DB2®  
IBM®  
InfoSphere®

RackSwitch™  
Redbooks®  
Redpaper™  
Redbooks (logo) ®  
System Storage®

System Storage DS®  
Tivoli®  
WebSphere®

The following terms are trademarks of other companies:

Netezza, and N logo are trademarks or registered trademarks of IBM International Group B.V., an IBM Company.

Intel, Intel Xeon, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.



# Preface

Today, businesses need to manage and control the unprecedented complexity, speed and volume of data. To succeed, they must find ways to affordably manage this big data, while extracting greater business insights from it. They need to empower their employees, innovate, serve their customers, and increase revenue. They must also find a way to manage their rapidly expanding data growth sustainably, while finding new ways to get the most possible value from their data.

IBM® and NetApp have extended their partnership to develop a reference architecture, based on Apache Hadoop, to help businesses gain control over their data, meet tight service level agreements (SLAs) around their data applications, and turn data-driven insight into effective action. Big analytics storage solutions from IBM deliver the capabilities for ingesting, storing, and managing large data sets with high reliability. IBM InfoSphere® BigInsights™ provides an innovative analytics platform that processes and analyzes all types of data to turn large complex data into insight.

This IBM Redpaper™ publication provides basic guidelines and best practices for how to size and configure the IBM and NetApp reference architecture based on IBM InfoSphere BigInsights.

## Authors

This paper was produced by a team of specialists from around the world working with the International Technical Support Organization, Tucson Center.

**Prem Jain** is a Senior Solutions Architect and Field Technical Lead with the NetApp Big Data team. Prem's career of more than 20 years in technology consists of solution development for data migration, virtualization, high-performance computing (HPC) and now big data initiatives. He has architected innovative big data and data migration solutions and authored several reference architectures and technical white papers. Prem holds a degree in Electronics Engineering from Nagpur University, India.

**Stewart Tate** is a Senior Technical Staff Member at IBM Silicon Valley Lab in San Jose, California. He has over 30 years of IBM experience and is currently responsible for the design, development, and deployment of one of the largest big data clusters within IBM. Stewart is responsible for improving the performance and scalability of IBM software products and developing business solutions for IBM customers. These solutions use BigInsights (Hadoop), IBM DB2®, IBM WebSphere®, Content Management, and IBM Tivoli® products. He has employed his background in hardware, networking, performance tuning, and also software development to architect and develop large scale, high-performance systems including a number of well known web sites.

Thanks to the following people for their contributions to this project:

Karla Magana Renoud  
Vicente Moranta  
Raul Raudry  
John Sing  
Manuel Vega  
IBM Systems & Technology Group, Storage Platform

Melinda L Harris  
IBM Sales & Distribution, Inside Sales

Larry Coyne  
Megan Gilge  
IBM International Technical Support Organization

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks® publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on Facebook:  
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:  
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>





# Introduction and solution summary

Today, businesses need to manage and control unprecedented complexity, speed, and volume of their data. To succeed, they must find ways to manage this big data affordably, while extracting greater business insights from it. They need to gain competitive advantage, innovate, serve their customers better, and increase revenue. They must also find a way to manage their rapidly expanding data growth sustainably while finding new ways to get the most possible value from their data.

IBM and NetApp have extended their partnership to develop a reference architecture to help businesses get control over their data, meet tight SLAs around their data applications, and turn data-driven insight into effective action. This reference architecture is built on Hadoop, a powerful, open big data platform that offers the flexibility and scale you need for the long-term.

Big analytics storage solutions from IBM deliver the capabilities for ingesting, storing, and managing large data sets with high reliability. IBM BigInsights provides an innovative analytics platform that processes and analyzes all types of data to turn large complex data into insight.

Big Data Networked Storage Solution for Hadoop delivers “big analytics” with the following advantages:

- ▶ Quick insights from unstructured, polystuctured and structured data with ready-to-deploy integrated hardware stack and prebuilt queries for instant analysis
- ▶ Resilient implementation of Hadoop for the enterprise with lower downtime, higher data availability, all of which reduces risk of deploying Hadoop
- ▶ Cost-effective and higher storage efficiencies, lower operational expenses, and full set of services

## 1.1 Meeting the challenge of big data analytics with IBM InfoSphere BigInsights

Every day, we create 2.5 quintillion bytes of data; this is so much that 90% of the data in the world today has been created in the last two years alone. This data is from everywhere: sensors used to gather climate information, posts to social media sites, digital pictures and videos, purchase transaction records, and cell phone GPS signals, to name a few. This data is *big data*.

### 1.1.1 The value of the IBM InfoSphere BigInsights analytics

The IBM InfoSphere BigInsights software platform helps firms discover and analyze business insights hidden in large volumes of a diverse range of data, data that is often ignored or discarded because it is too impractical or difficult to process using traditional means. Examples include log records, click-streams, social media data, news feeds, email, electronic sensor output, and even some transactional data. To help organizations process these and other forms of data that are increasing in volume, variety, and velocity, IBM developed BigInsights, a software platform that includes the open source Apache Hadoop framework.

### 1.1.2 Analysis and discovery

BigInsights helps firms analyze large volumes of document and message content with its built-in text processing engine and library of annotators. Developers can quickly query and identify items of interest (such as persons, email addresses, street addresses, phone numbers, URLs, and business alliances) to understand the context and content of relevant business information hidden in unstructured text data.

In addition, programmers can use the BigInsights Eclipse-based plug-in to create their own text analytic functions. Built-in pattern discovery, expression builders, and a test environment promote rapid prototyping and validation of custom text annotators.

To help business analysts and non-programmers benefit from big data, BigInsights provides a spreadsheet-like discovery and visualization facility. Through the BigInsights web console, users can collect and integrate a variety of data into a spreadsheet structure, explore and manipulate that data using built-in functions and macros, create charts, and export the results, if they want.

The web console also features a catalog of applications that authorized users can launch immediately or schedule for future execution. IBM provides prebuilt applications (apps) for web crawling, importing and exporting data, accessing a public forum search engine, querying BigInsights data, and performing other work. Users can also publish their own applications in this catalog.

### 1.1.3 Enterprise software integration

Integrating BigInsights, and its analysis of big data, with existing enterprise software is a key initiative of IBM. For this reason BigInsights provides connectivity to popular data warehouse platforms, including IBM DB2, Netezza®, and other offerings, not from IBM. As a result, BigInsights developers can join reference data from a relational database with data managed by BigInsights to refine and expand their analysis.

The IBM approach ensures that your big data and traditional enterprise data will not exist in isolation. Instead, you can use your existing enterprise software platforms to do what they do best: use BigInsights for analytical workloads that are not appropriate or practical for these platforms, and broaden your business analysis capabilities in an integrated fashion.

Using technologies such as Hadoop MapReduce, data analytics can process very large amounts of both structured and unstructured data. In contrast, the traditional relational database (RDB) with structured data is a different tool for a different job. Relational databases are designed for many concurrent users, with many small transactions (such as inventory events, reservations, and banking), with all of the related Structured Query Language (SQL), table, row, column, and join design assumptions. Hadoop and RDB solutions can (and often do) work together in commercial tasks to reduce an ever-expanding ocean of data into useful information.

Big Data Networked Storage Solution for Hadoop was designed to meet several (often conflicting) technical and market demands. The enterprise Hadoop ecosystem is a hybrid of open source and enterprise sensibilities, and many aspects of this solution take its hybrid nature into account.

Hadoop has its origins in distributed computing. This form of computing divides the data set into thousands of pieces that can be analyzed without intervention from any of the other pieces. This programming or computing style is often called *shared nothing*; these shared-nothing programs run best on hardware platforms that also share little or nothing.

The “divide and conquer” strategy of distributed computing is not new to enterprise customers. A relational database management system (RDBMS) such as Oracle or DB2 use parallel technology to parallelize the scanning of large tables to satisfy a particular class of queries. This technology parallelizes the scan, filters and sorts the intermediate results, and merges them into a query result set that is returned to the user. What they care about is time. Hadoop uses map (filter and sort) and reduce (merge) approach to accomplish the same effect.

The two components of Hadoop are as follows:

- ▶ MapReduce is the Hadoop framework for parallel processing.
- ▶ Hadoop Distributed File System (HDFS) is the distributed file system that provides petabyte-size storage and data management.

## 1.2 Hadoop Distributed File System

Hadoop Distributed File System (HDFS) is optimized to support very large files that typically range in size from megabytes to petabytes. A better approach is to have a relatively modest number of these large files than to have a large number of smaller files.

HDFS is designed for streaming data access. At load time, HDFS automatically spreads data across nodes in the cluster to facilitate parallel processing of that data at read time. Unlike many traditional file systems, HDFS is designed to minimize the time required to read an entire data set rather than the time required to read the first record or a small subset of the data. Thus, HDFS is not suitable for applications that require low-latency data access; instead, it is designed to provide efficient, batch access to large volumes of data. HDFS has a simple concurrency model that presumes data will be written once (by one writer) and read many times by various applications. There is no support for multiple concurrent writers or for updating a subset of data at an arbitrary point within a file.

HDFS was designed to provide fault tolerance for Hadoop. By default, data blocks (that is, portions of a file) are replicated across multiple nodes at load or write time. The degree of replication is configurable; the default replication is three. Replication protects the data and helps Hadoop recover from hardware failures that can periodically occur when using large clusters of low-cost commodity hardware.

A traditional Hadoop cluster consists of a few basic components:

- ▶ The NameNode, which manages the HDFS namespace.
- ▶ The Checkpoint node, which is a secondary NameNode that manages the on-disk representation of the NameNode metadata.
- ▶ The JobTracker node, which manages all jobs submitted to the Hadoop cluster and facilitates job and task scheduling.
- ▶ Hundreds of subordinate nodes that provide both TaskTracker and DataNode functionality. These nodes perform all of the real work done by the cluster. As DataNodes, they store all of the data blocks that make up the file system, and they serve I/O requests. As TaskTrackers, they perform the job tasks assigned to them by the JobTracker.

Because Hadoop is built on top of a distributed file system, HDFS, this arrangement allows every DataNode and TaskTracker node to see and access all data across the cluster. The NameNode houses and manages the metadata, so whenever a TaskTracker must read or write an HDFS block, the NameNode informs the TaskTracker where a block exists or where one should be written.

HDFS is a feature-rich, integrated, and distributed file system that offers data protection, fault tolerance, and the ability to balance workloads. Data protection is implemented by using a cluster-aware, software-based mirroring scheme. Unlike a classic triple hardware mirroring scheme, HDFS dynamically determines where a given block's mirror block (replica) will be located. Another innovation of HDFS is that the mirroring scheme can be set on a file-by-file basis. The replication parameter is used to set the mirroring for a given file. The default practice in Hadoop is to set all data files to a replication count of three. Another innovation of HDFS is the ease with which the replication or mirroring level can be adjusted. To change the replication count of a file, you issue the **hadoop fs -setrep** command, and Hadoop schedules the additional copies or reclaims the space from mirror copies that are no longer used.

Because HDFS is distributed, all data resides on individual DataNodes. If a DataNode or sections of the HDFS residing on a DataNode become inaccessible or corrupt, actions are taken to redirect access to the affected data to one of the other copies, and to initiate repairs in the event of block corruption.

HDFS can provide job completion in the face of hardware failures because copies of the data are located on other nodes. If a node or disks in a node fail where a portion of a MapReduce job (task or set of tasks) was running, the affected tasks are all relocated by the JobTracker to other nodes where the data does still exist and are then re-executed. In addition to data protection, HDFS mirroring ensures that MapReduce jobs complete successfully regardless of disk or node failures. Loss of service rarely involves the destruction of data, but HDFS handles both forms of loss.



## 1.3 Triple-mirror trade-offs

Although HDFS is distributed, feature rich, and flexible, both throughput and operational costs are associated with server-based replication for ingest, redistribution of data following recovery of a failed DataNode, and space utilization.

Costs of triple replication include the following items:

- ▶ For each usable terabyte (TB) of data, 3 TB of raw capacity are required.
- ▶ Server-based triple replication creates a significant load on the servers themselves. At the beginning of an ingest task, before any data blocks are actually written to a DataNode, the NameNode creates a list of DataNodes where all three of the first block replicas will be written, forming a pipeline and allocating blocks to those nodes. The first data block arrives over Ethernet; traverses the server Southbridge, Northbridge, and memory bus; and is written to RAM. After being passed to the storage drivers, the data then traverses back across the Northbridge and the Southbridge and is eventually written out to drives. That DataNode forwards the same block to the next node in the pipeline and the preceding procedure is repeated. Finally, this second DataNode sends the block to the third DataNode and the write to storage follows the same I/O path as previous writes. This results in a significant load on server resources.
- ▶ Server-based triple replication also creates a significant load on the network. For ingest, three network trips are required to accomplish a single block write. All compete for and consume the same network, CPU, memory, and storage resources allocated to process the Hadoop analysis tasks. The IBM solution can provide better reliability with double replication or needing only two copies of data.

With the advent of more powerful entry-level servers, modern Hadoop clusters are evolving into a hybrid of symmetric multiprocessing (SMP) and massively parallel processing (MPP) supercomputing clusters. A typical low-cost DataNode now includes 8 - 12 CPU cores, following a trend toward increased capability that is driven by Moore's Law (which observes that the number of transistors on a given chip doubles approximately every two years) and by the highly competitive x64 server market. A cluster-wide MapReduce job might have 16 tasks running on each of 64 nodes. This modest cluster has a degree of parallelism of 1,024 (16 x 64). This is a core concept to Hadoop, so most customers quickly understand how much computing they can bring to bear on the analysis of their data. What they often overlook is the fact that this computing power can (and must) be used to load the data also.

Anything done to reduce the consumption of server and network resources during the loading of data increases cluster efficiency.

## 1.4 When drives fail

Inevitably, disk drives in a Hadoop cluster can fail at some point, and the larger the cluster, the more likely that a drive failure event will occur. Although most customers do not deploy 5,000-node clusters, a cluster of this size might contain 60,000 disks. Even though means are not a good metric for drive failure, mean time between failure data suggests that a drive will fail every few hours in a 60,000-spindle deployment. In a modest cluster, the number of failures is much lower, but if one or more drives fail, the entire DataNode may be removed from service. At the least, all tasks accessing data on that drive will fail and be reassigned to other DataNodes in the cluster. This in itself can result in severe degradation of completion time for running jobs.

In addition to that, the drive must be physically replaced, mounted, partitioned, formatted, and reloaded with data from nearby copies before HDFS can make full use of that drive. Repopulating the new disk with data can be accomplished by using the Hadoop balancer, which redistributes HDFS data across all DataNodes, by moving blocks from over-utilized to under-utilized DataNodes. The balancer runs in the background and, by default, it copies at only 1 MBps. That default copy rate can be modified by setting the following property:

```
dfs.balance.bandwidthPerSec
```

Based on that setting, rebalancing to repopulate a replaced 2 TB disk might take hours or days, depending on allowed resource utilization. Either way, the process diverts network bandwidth away from running jobs. Also, remember that HDFS is able to generate only about 30 MBps per SATA disk, making disk I/O bandwidth another limited resource, which is affected by the balancer.

For every full 2 TB SATA disk that fails, approximately 70 hours or more is required for rebalancing the distribution of blocks following replacement of that disk, depending on tolerance for network resource consumption. The trade-off is between rebalancing time, network utilization, and disk I/O bandwidth utilization. Remember that higher resource use affects MapReduce job performance, especially during data ingest operations. In some cases, rebalancing may also require some manual movement of blocks between disk drives.

## 1.5 Reference architecture for big data analytics

As enterprises begin to move Hadoop deployments out of the lab and into production, they typically expect a certain degree of scalability and efficiency. IBM Open Solution for Hadoop uniquely complements the IBM feature-rich BigInsights software suite to create an end-to-end solution that focuses on functionality, efficiency, and quality that are so critical. Figure 1-1 shows IBM integration with Hadoop.

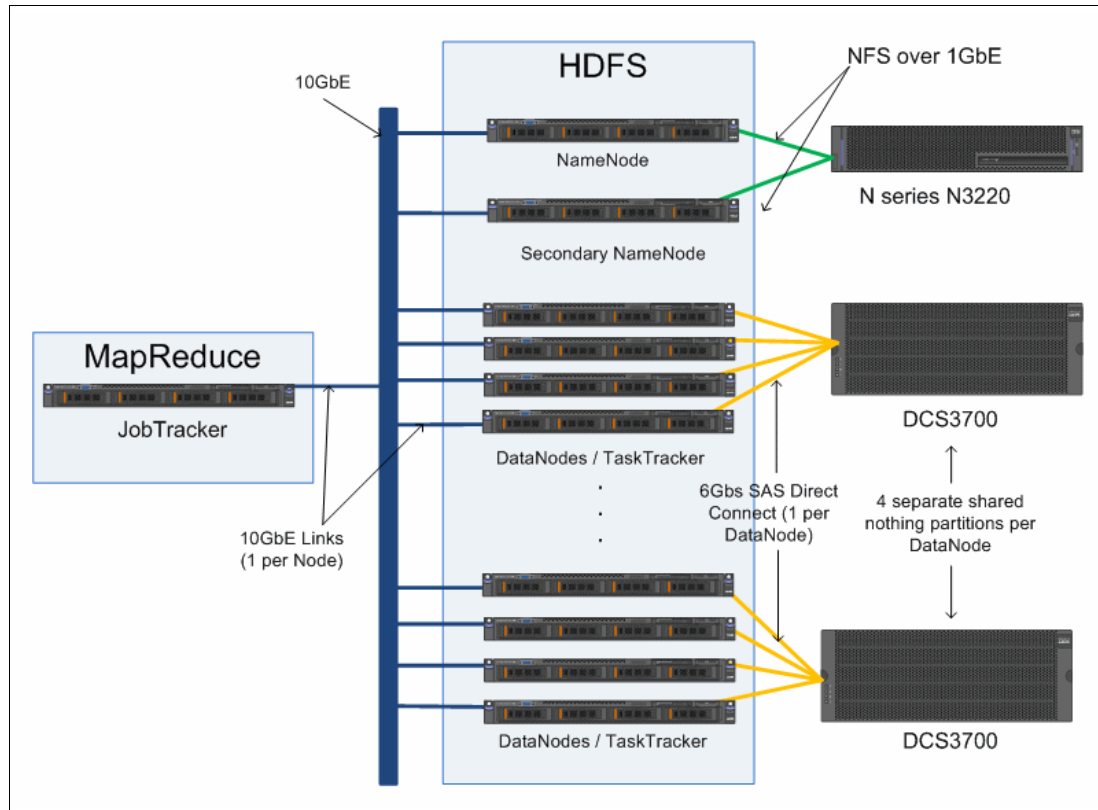


Figure 1-1 IBM Integration with Hadoop

IBM Open Solution for Hadoop (Figure 1-1) addresses two operational limitations of current Hadoop clusters:

- ▶ Reducing the operational and functional costs associated with distributed, software-based mirroring (block replication) with a replication count of two, instead of three.
- ▶ Addressing the most significant single point of failure in Hadoop, the NameNode, by maintaining an up-to-date copy of all HDFS metadata, external to the NameNode server. This enables fast recovery of HDFS after a NameNode server crashes.

The IBM Open Solution for Hadoop includes the following hardware and software:

- ▶ Hardware
  - Hadoop data storage array

One IBM System Storage® DCS3700 serves data for four DataNodes. Using RAID 5, the storage array is able to protect the data and minimize the performance impact of disk drive failures.
  - Hadoop metadata storage

N series N3220 or other N3000 Express series storage array provides a Data ONTAP grade of data protection for the NameNode and a single, unified repository for Linux, Hadoop, and other software (and scripts) to simplify deployment, updates, and maintenance.
- ▶ Software
  - NFS license for IBM System Storage N3220
  - Hadoop distribution included in IBM InfoSphere BigInsights

## 1.6 Benefits of the Big Data Networked Storage Solution for Hadoop

The core benefit of the IBM storage and IBM BigInsights combination is an enterprise-grade HDFS platform for implementing Hadoop that quickly and easily converts all their types of data into business insights:

- ▶ Lower cluster downtime with transparent RAID operations and transparent rebuild of failed media
- ▶ Less storage needed with replication count of two
- ▶ Robustness of HDFS through reliable NFS server based Metadata Protection
- ▶ Performance for the next generation of DataNodes (SMP) and networks (10GbE)
- ▶ Highly reliable and fully manageable BigInsights Hadoop Platform
- ▶ Proven server, storage and networking hardware based on enterprise-grade designs, offering enterprise-grade performance, uptime, and support
- ▶ Fully online serviceability
- ▶ Industry-leading analytics tooling
- ▶ Stable, scalable and supported production-ready code from the open source community
- ▶ Business process accelerators
- ▶ Comprehensive set of analytical tools including leading edge text analytics, and spreadsheet style analysis tools
- ▶ RDBMS, warehouse connectivity
- ▶ Easier and simpler administration with a web-based console, flexible job scheduler, LDAP authentication

## 1.7 Solution summary

The Big Data Networked Storage Solution for Hadoop provides industry standard enterprise data protection through the Data ONTAP-based NFS storage and IBM InfoSphere BigInsights Management layer.

The IBM DCS3700 arrays provide storage services to DataNodes. The DCS3700 is not a shared-storage infrastructure because Hadoop is a shared-nothing programming paradigm. Each DataNode accesses its own dedicated set of 15 disks from the DCS3700 array. In Hadoop, the computing is brought to the data. The Big Data Networked Storage Solution for Hadoop reference architecture described in Chapter 2, “Solution architecture” on page 11 acknowledges the shared-nothing nature of Hadoop by providing each DataNode with two dedicated RAID-protected volumes with enterprise-grade capabilities often found in a traditional storage area network (SAN) environment.

With support for the latest Intel Xeon processor technology, the IBM System x3530 M4 offers cost-optimized performance and memory with better energy efficiency. The x3530 M4 provides a simpler, more affordable alternative to traditional enterprise offerings without sacrificing performance.

The x3530 M4 offers added flexibility and an innovative design to help you more easily and cost effectively add new features as your requirements change. Depending on the model you select, these can include graphic card support, advanced RAID and light path diagnostics, hot-swap hard disk drives, rich remote-management capabilities and a shared or dedicated port for system management. Expandable PCIe slots and NIC ports let you pay for new capabilities as you need them.

### 1.7.1 Data protection and job completion

HDFS provides data protection and other benefits by implementing host-based, software triple mirroring. For every block of data loaded into HDFS, two copies are written to two other nodes in the cluster. This is a replication count of three, and it is the default or standard practice for most Hadoop deployments that use standard “just a bunch of disks” (JBOD) SATA DAS disks co-located in a DataNode. HDFS is one of the first file systems to implement a replication strategy on a file-by-file basis. This flexibility allows customers to adjust the protection level to suit particular data sets.

HDFS replication provides data protection, job completion in the event of disk failure or DataNode failure, and load balancing. Replication for an entire cluster can be set by using a single parameter, `dfs.replication`, which has a default value of three. It can also be set for an individual file at the time of file creation, and existing files can be changed by using the **`hadoop fs -setrep`** command. The higher the HDFS replication count, the more copies of the data must be stored, but the more resilient and balanced the access is to that data. The limitation is the same found with all data-based (versus parity-based) protection schemes: storage utilization efficiency.

When a customer deploys with the default replication count of three, for every 1 PB (petabyte) of data loaded into a Hadoop cluster, another 2 PB must be copied across the network fabric to other nodes. High-replication-count files consume a lot of space and take a long time to ingest because 2 PB of the 3 PB must be written over the network. The IBM System Storage DCS3700 platform offers a more efficient approach to significantly reduce the overhead of data protection, making HDFS more efficient and faster (certainly on loads).

This is the single most important feature of the IBM solution: offloading some or even all of the data protection aspects of HDFS into enterprise-grade, industry-proven IBM DCS3700

controllers. The second most important reason for having replicated copies is for job completion. If job tasks on a given DataNode fail for any reason, the JobTracker must reassign the failed tasks to other nodes in the cluster. Eventually the JobTracker may put the node on a blacklist and the NameNode may recognize the node as being in a failed state. If task reassignment is not possible, then the MapReduce job cannot complete.

An important aspect to understand is that this risk window is open only during a MapReduce job. This job-completion mechanism can function only if there are one or more copies of the data in the cluster. IBM advises that jobs can run safely with a replication count of two. The data is already protected by using RAID 5 on the DCS3700 logical volumes used for data storage.

There might be use cases where a replication count higher than two benefits MapReduce job performance, especially when multiple jobs that are using the same data run simultaneously. In that case, higher replication counts may offer greater parallelization of job tasks while using HDFS data locality. Higher replication counts also enable the data scientist to better use the Hadoop “speculative execution” feature, which can help balance task execution across resources in a cluster.

Speculative execution enables the JobTracker to assign instances of a relatively slow-running task to other TaskTracker nodes. In this scenario, when the first instance of the task finishes, the other instance is killed and its results are discarded. This feature can benefit MapReduce performance in clusters where unused node resources exist, but it may actually be bad for performance in busy clusters. Given all the advantages and disadvantages of various replication counts discussed earlier, with the IBM Open Solution for Hadoop, a replication count of three is no longer a requirement for data protection. A setting of two is sufficient and represents a good balance between performance and data protection for a very large number of Hadoop use cases. As with all database and analytics systems, MapReduce jobs often benefit from tuning at both the system and job code levels. Replication count is one of many tuning parameters to consider.

The combination of removing and reducing most (but not all) of the risk means that MapReduce jobs can run successfully at HDFS with a replication count of one. Consider, however, that IBM suggests a base replication count of two. A replication count of one greatly increases the risk that a MapReduce job will fail and introduces a small amount of risk that data might actually be lost. That risk should be carefully weighed against any perceived advantages of disabling replication. A replication count of one also greatly affects the ability of job tasks to use data locality, resulting in significantly higher network utilization, because more data blocks must be accessed over network fabric.



## Solution architecture

This chapter describes the hardware components, software requirements, and the solution architecture details for the Big Data Networked Storage Solution for Hadoop reference design.

### 2.1 Hardware requirements

The Big Data Networked Storage Solution for Hadoop reference architecture consists of the following hardware components.

#### 2.1.1 Servers

The solution requires at least one IBM System x3530 M4 server; this server (or servers) must include the following items:

- ▶ Support all Intel Xeon E5-2400 Series processors
- ▶ Up to 2.4 GHz (6-core) or 2.3 GHz (8-core) processors
- ▶ Memory: 12 DIMMs (up to 384 GB); Up to 1600 MHz
- ▶ Hard disks: 2 x 3.5" HS HDDs
- ▶ Network: 2 x 1 Gbps Ethernet ports, 10 Gb network port
- ▶ Host Bus Adapter: LSI SAS 9200-8e
- ▶ SFF-8088 external mini-SAS cable (For maximum signal integrity, the cable should not exceed 5 m.)

#### 2.1.2 Storage

The solution requires at least one IBM System Storage DCS3700; the storage array (or arrays) must include the following items:

- ▶ DCS3700 controller with power cords
- ▶ 4 SAS cables
- ▶ 8 GB of cache
- ▶ 60 2TB or 3TB, 7200 rpm NL-SAS 3.5 disks
- ▶ IBM System Storage DS® Storage Manager V10.8x

The solution requires at least one IBM System Storage N series N3220; the storage array (or arrays) must include the following items:

- ▶ 7,900 GB SAS drives
- ▶ NFS licence

## 2.2 Software requirements

Requirements are as follows:

- ▶ IBM InfoSphere BigInsights Hadoop Software Suite
- ▶ Red Hat Enterprise Linux RHEL5.7 or later

## 2.3 Solution architecture details

The Big Data Networked Storage Solution for Hadoop reference design has two major hardware components. The IBM N series N3220 provides enterprise-grade protection of the NameNode metadata, and the IBM DCS3700 storage array provides the data storage services to four Hadoop DataNodes.

Each DataNode has its own dedicated, nonshared set of disks. This exactly follows the way in which Hadoop DataNodes configure capacity when these drives are physically co-located in the chassis.

In practical terms, Linux is not able to distinguish between a set of logical unit numbers (LUNs) presented by the LSI 1068 controller chip on the host bus adapter (HBA), and a set of disks presented by either an embedded SAS or a SATA controller. There is no logical difference. However, there are differences in performance and reliability. This set of disks is configured as two blocks, each using one of two parity-based protection schemes. Eight volume groups are configured in the DCS3700, and they are configured so that each DataNode sees only its set of two private blocks. This design is an improved alternative to packaging JBOD DAS media for four DataNodes, offering better performance, reliability, data availability and uptime, and manageability.

Each block is a RAID-protected volume group that contains a single virtual logical volume, which can be created and exported to a given DataNode as a LUN. These LUNs appear as physical disks in Red Hat Enterprise Linux (RHEL). They can be partitioned, and file systems can be created and mounted to store HDFS blocks. HDFS blocks can be 64 MB, 128 MB, 256 MB, or even 512 MB in size.

### 2.3.1 Network architecture

The network architecture of a Hadoop cluster is critical. With a default replication count of three, more than two-thirds of all I/O traffic must traverse the network during ingest, and perhaps a third of all MapReduce I/O during processing runs might originate from other nodes. The IBM DCS3700 storage modules provide a level of performance that is significantly better than JBOD SATA. The DCS3700 offers a well-designed, enterprise-class performance platform based on four 6 Gbps SAS ports, supported by proven caching and best-in-class hardware RAID.

The solution uses two network backplanes. The core of this solution is the HDFS network which is a Gb network backplane served by high performance IBM BNT® /10/40Gb Switch Solution for Top of Rack switch. The other network is a Gb Ethernet (GbE) network, which



caters to the administration network and also connects the NameNode, JobTracker, and the IBM N series storage systems.

On the 10GbE network, all components should be configured for jumbo frames. This requirement also applies to the switch ports to which the components are connected. Any ports on the 10GbE switches that are configured for GbE connections (such as N3220 NFS and uplinks) should not have jumbo frames enabled.

### 2.3.2 BigInsights cluster network

The servers described in this reference architecture are interconnected by the following three dedicated networks:

- ▶ Data network
  - Private interconnect between all BigInsights nodes
  - Uses its own VLAN and subnet
  - Primary purpose: Data ingest and movement within cluster
- ▶ Administration network
  - System administration network
  - Isolated VLAN and subnet
  - Primary purpose: Administration of cluster servers through **ssh** and **rsh** commands
- ▶ Management network
  - IBM Integrated Management Module
  - Isolated VLAN and subnet
  - Primary purpose: Hardware monitoring

### 2.3.3 Network switch

Designed with top performance in mind, the IBM BNT RackSwitch™ G8264 is ideal for today's big data optimized workloads. They are enterprise-class and full-featured data-center switches that deliver line-rate, high-bandwidth switching, filtering, and traffic queuing without delaying data. Large data-center grade buffers keep traffic moving. Redundant power and fans along with numerous high availability features equip the switches for business-sensitive traffic.

The RackSwitch G8264 is ideal for latency-sensitive applications, such as high-performance computing clusters and financial applications. The G8264 supports IBM Virtual Fabric to help clients reduce the number of I/O adapters to a single dual-port 10 Gb adapter, helping reduce the cost and complexity. The RackSwitch G8264T can be used as part of a comprehensive solution based on the 10GBase-T Ethernet standard, better management, and a cost-effective option for next-generation data centers. Flexible connectivity across distances up to 100 m at a low cost makes the G8264T an optimal choice for connecting high-speed server and storage devices.

Figure 2-1 shows an IBM BNT RackSwitch G8264.

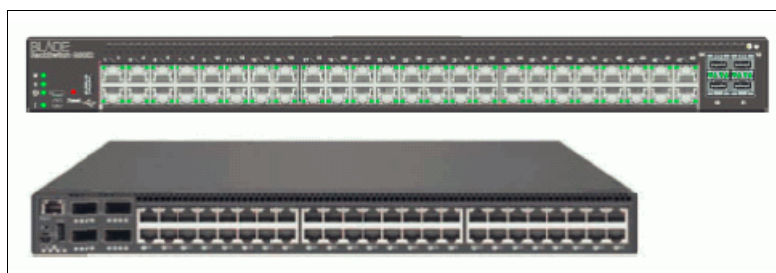


Figure 2-1 IBM BNT RackSwitch G8264

### 2.3.4 Cabling topology

The server-to-storage-array cabling is simple. Each DataNode requires a two-port eSAS HBA. Only one of these ports is used to connect the DataNode to the eSAS port on the DCS3700 that has been designated for this DataNode. The cable length is typically 1 m, but it can be up to 5 m. The eSAS specification indicates that cables can be up to 10 m, but for a variety of manufacturing reasons, the longest cable is probably 5 m.

The solution was designed to be compact and to minimize the collocation rack space footprint. In a vertical racking deployment, the DCS3700s and N3220 would be found in the lower 14U of a rack (because of weight), and the remaining 18U is for 15-1U servers and the Ethernet switch. Most racks are not populated above the 32U line. In this rack example, 2 m cables are sufficient for all DataNodes to connect to their respective DCS3700s.

Figure 2-2 shows the value configuration.

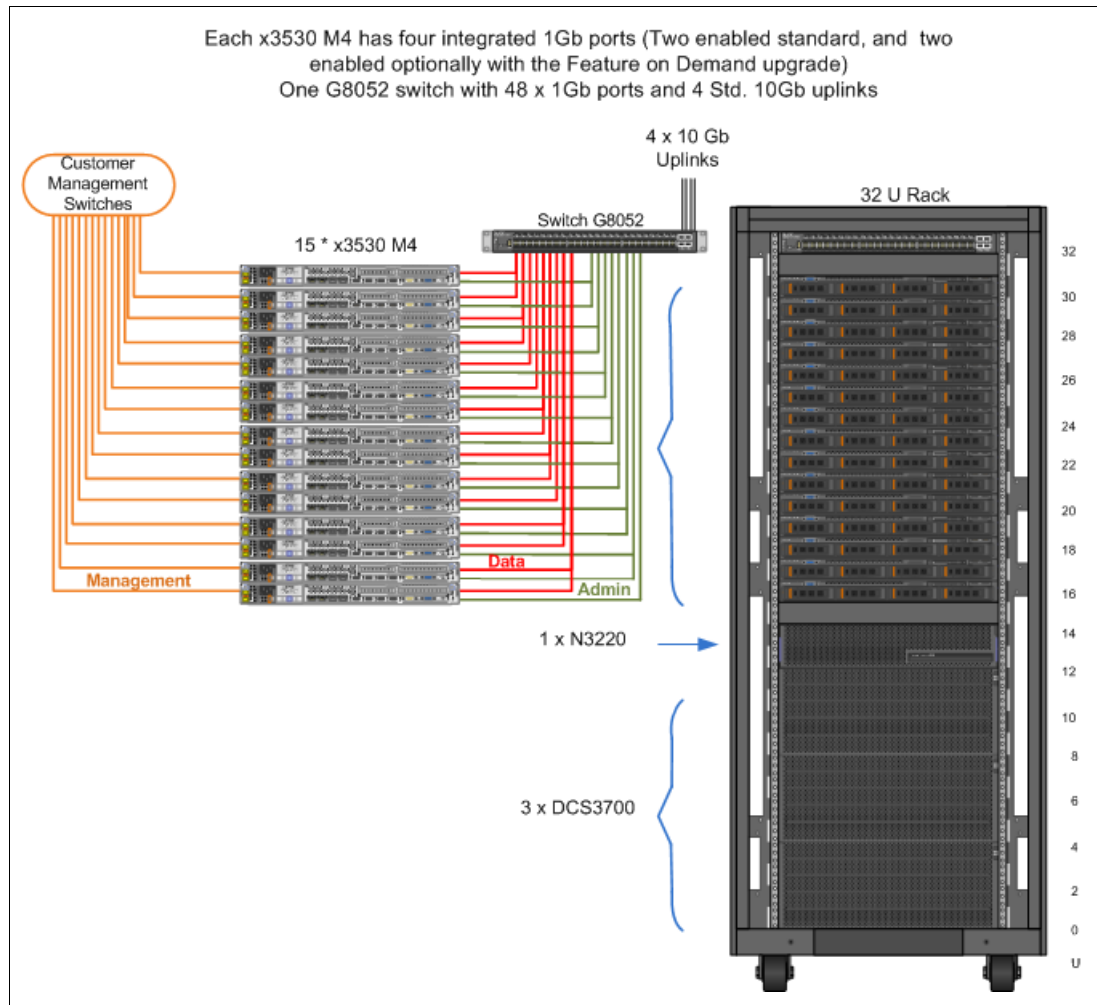


Figure 2-2 Value configuration

Horizontal deployment is another racking topology where the DCS3700s and the N3220 are in a dedicated rack and the servers are in the adjacent rack. In this case, 0.5 m cables are adequate.

### 2.3.5 Storage architecture

As much as possible, the DCS3700 operates as eight completely independent storage modules. Each array is configured as eight independent RAID groups of seven disks that can be set up in a RAID 5 (2 x 6 + 1) configuration. This consumes 56 disks (8 x 7). The remaining four disks are global hot spares. If customers deploy all of their files with replication count of two, then using a single-parity drive over six spindles provides a good balance between storage space utilization and RAID protection. As a result, there are constantly two forms of protection when the DCS3700s are running with files set to replication count of two.

Figure 2-3 shows a DCS3700 configured with replication count of two.

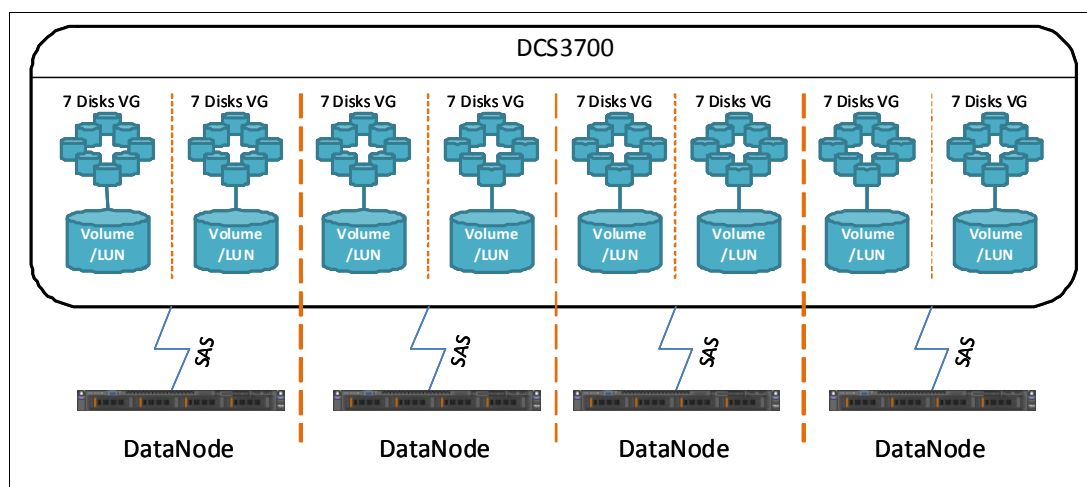


Figure 2-3 DCS3700 configured with replication count two

### 2.3.6 Storage sizing and performance considerations

The amount of usable space in HDFS varies by the replication count for any given set of files. The standard Hadoop cluster usually runs all files at replication count of three. This results in having only one-third of all the file system space available for use. Typically, HDFS clusters are measured by their available space and usable or loadable space.

A customer with 9 TB of available space mounted as file systems can load or use only 3 TB of data when using replication count of three. The IBM Open Solution for Hadoop allows customers to run at replication count of two, increasing the amount of loadable space per file system by about 50% compared to HDFS configured with replication count of three.

Storage must also be allocated for temporary Map task output. Map task output is usually written to local Linux file systems, not HDFS. A common practice is to spread the Map I/O across the same disks used by HDFS. The IBM Open Solution for Hadoop follows this practice also. The accepted guideline is to allocate 30% of storage capacity for Map output. This must be considered in sizing storage for HDFS.

### 2.3.7 NameNode metadata protection

As discussed previously, this solution offers a unique feature of NameNode metadata protection and the robustness and reliability of an enterprise grade product. This protection is offered by introducing an IBM N series N3220 NFS storage system. The NameNode is configured to keep a copy of its HDFS metadata (FSImage) into the HFS mounted storage. The same NFS mounted storage is also made available to the secondary NameNode.

In case of a NameNode failure, the critical metadata is still safe on the IBM N series storage system. At this point, we can recover the metadata and restart services on the secondary NameNode, to get the HDFS back and running promptly.

Figure 2-4 shows NameNode configuration stores metadata on NFS mounted storage.

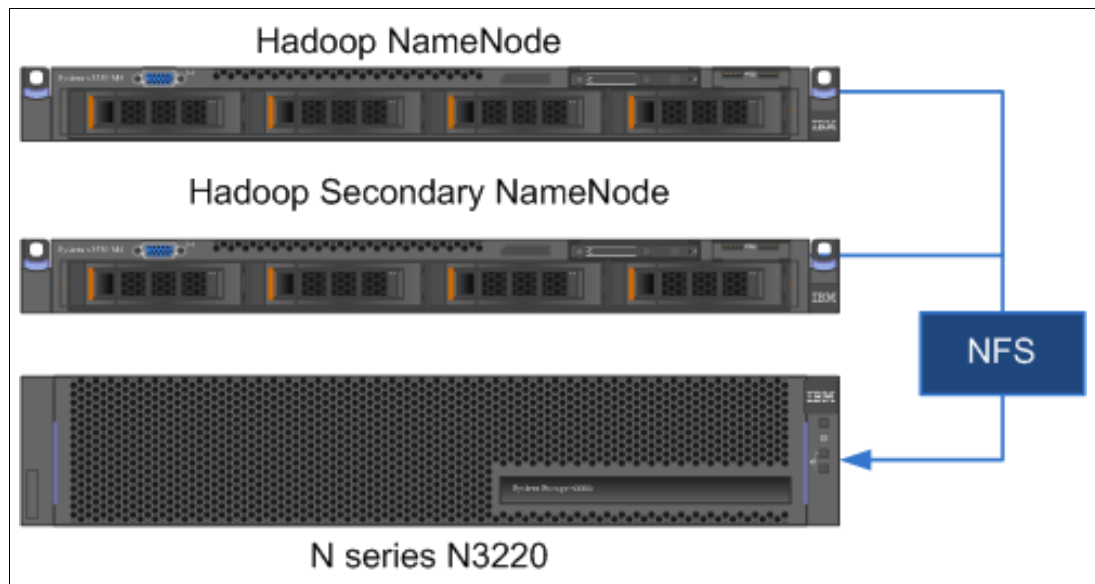


Figure 2-4 NameNode configuration stores metadata on NFS mounted storage

### 2.3.8 Rack-awareness implementation

We chose to implement rack-awareness in our Hadoop solution because of the benefits it offers customers. Using this feature, we are able to configure Hadoop to know the topology of the network.

This way is important for two reasons:

- ▶ In most rack network configurations, more network bandwidth is available within a single rack than between racks. Rack awareness enables Hadoop to maximize network bandwidth by favoring block transfers within a rack over transfers between racks. Specifically, with rack awareness, the JobTracker is able to optimize MapReduce job performance by assigning tasks to nodes that are “closer” to their data in terms of network topology. This is particularly beneficial in cases where tasks cannot be assigned to nodes where their data is stored locally.
- ▶ By default, during HDFS write operations, the NameNode assigns the second and third block replicas to nodes in a different rack from the first replica. This provides data protection even against rack failure; however, this is possible only if Hadoop was configured with knowledge of its rack configuration.

If rack-awareness is not configured, all DataNodes in the cluster are assumed to be in the same rack, which is named `default-rack`. To assign rack awareness to each node, follow the steps that are outlined in the Rack Awareness topic at the Hadoop Cluster Setup web page:

[http://hadoop.apache.org/docs/stable/cluster\\_setup.html](http://hadoop.apache.org/docs/stable/cluster_setup.html)

Be careful to ensure that for each HDFS block, the corresponding replica is stored on another DCS3700 controller. In multi-rack clusters, providing the actual rack location of each DataNode will accomplish this. If all the DataNodes are located in the same rack, “rack” mapping must be extended to reflect the DCS3700 controller used for DataNode storage.


Basically, use the following guidelines to prevent data loss in the unlikely event of storage controller failure:

- ▶ For multi-rack clusters, define the topology based on the actual rack location of each DataNode.
- ▶ For single-rack clusters define the topology based on the DCS3700 controller used for the storage by each DataNode.

In addition to the guidelines, be careful not to define network topologies that will result in unbalanced I/O across storage controllers. Consider a scenario where the defined topology results in one or more DataNodes receiving a higher number of block copies than others.

For example, the network topology for a four-DataNode cluster might be mistakenly configured with three nodes on one rack and one node on another rack. In that configuration, the second replica from three nodes will be copied to the fourth node with the second replica of the fourth node being copied to one of the other nodes. The storage controller supporting the fourth node will quickly become a bottleneck during data ingest. This might also introduce a bottleneck in read performance by affecting the ability of the JobTracker to use data locality in the assignment of job tasks.

Although this example is simplistic, and the scenario is unlikely for a four-DataNode cluster, as the node count increases, so does the likelihood that this can happen. Also, consider that since the defined network topology determines replica placement, rack awareness must be set up at the beginning, before any data is ingested. Otherwise, loss of data will likely occur in the unlikely event of storage controller failure.



## Big Data Networked Storage Solution for Hadoop reference architecture tests

The tested configuration for the Big Data Networked Storage Solution for Hadoop consisted of the following items:

- ▶ Three HDFS building blocks (single building block illustrated in Figure 3-1 on page 20), each consisting of the following components:
  - One IBM System Storage DCS3700 array containing sixty 2 TB or 3 TB NL-SAS disk drives
  - Four IBM x3530 M4 servers, each connected to the DCS3700 with a single 6Gbps SAS connection (one connection per DataNode)
- ▶ One NameNode server, IBM x3530 M4
- ▶ One secondary NameNode server, IBM x3530 M4
- ▶ One JobTracker server IBM x3530 M4
- ▶ One IBM N series N3220 equipped with seven 900 GB SAS disks
- ▶ All nodes (DataNodes) interconnected with 10GbE
- ▶ NameNode and secondary NameNode. JobTracker and N3220 connected through GbE (1 Gbps)

Figure 3-1 shows a single HDFS building block.

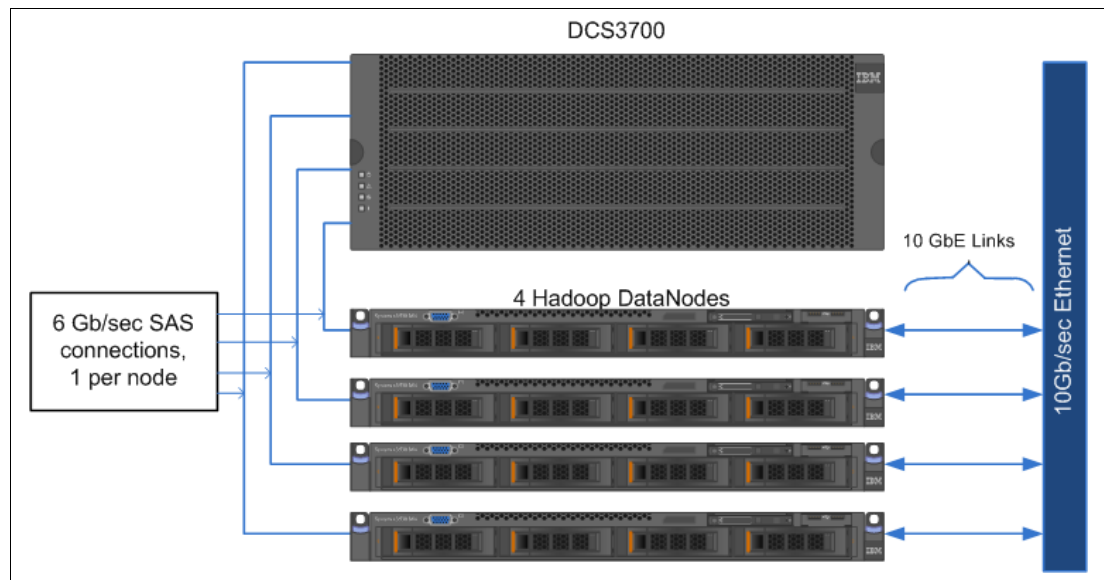


Figure 3-1 Single HDFS building block

Testing the Big Data Networked Storage Solution for Hadoop was conducted based on the objectives in the previous list. The test cases specific to these objectives are described in 3.2, “Solution testing” on page 21. An underlying objective was to demonstrate the scalability of our solution. Therefore, we chose to test a Hadoop cluster containing six HDFS building blocks. Figure 2-2 on page 15 shows the configuration that we actually tested.



Figure 3-2 shows a Big Data Networked Storage Solution for Hadoop tested configuration.

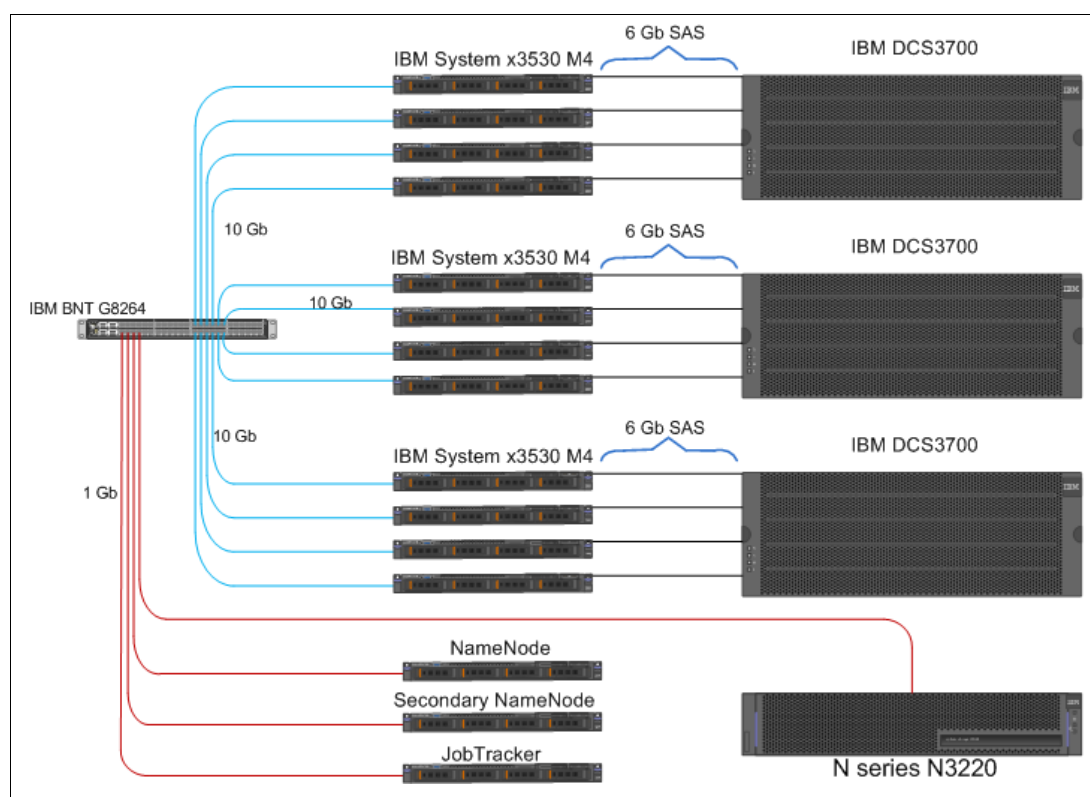


Figure 3-2 The Big Data Networked Storage Solution for Hadoop tested configuration

## 3.1 Solution objectives

Quality assurance testing was performed to meet the following key objectives:

- ▶ Verify that HDFS could be successfully implemented on servers using an IBM DCS3700 as disk storage.
- ▶ Demonstrate the ability to capture a backup copy of the HDFS (NameNode) metadata, in real time, on an IBM N series N3220 storage array through a mounted NFS file-share.
- ▶ Test the resiliency of the HDFS during simulated a DCS3700 storage array disk failure.
- ▶ Demonstrate the ability to restore the HDFS metadata from the IBM N series N3220 storage array, thereby restoring the proper state of the HDFS after a simulated Hadoop NameNode crash.

## 3.2 Solution testing

The pass/fail criteria for test cases 1-A, 1-B, and 2 (see 4.1, “Test case details” on page 36) were successful Hadoop MapReduce job completions with no Hadoop task failures. Because task failures are common in the Hadoop processing environment, Hadoop was designed to recover gracefully from transient task failures. Zero tolerance was chosen so that even failures that might look routine would not be dismissed without proper root cause analysis.

The pass/fail criteria for test case 3 was to have a complete file system metadata recovery from the backup copy on the N3220 storage array, and to successfully bring the HDFS back online.

Testing was divided into three test cases to verify that the objectives (described in 3.1, “Solution objectives” on page 21) were met.

### **3.2.1 Test case 1**

This test case was used to initially verify that all solution components and related SAS and network interconnects were working according to the architectural design. It was also used to verify that all of the Hadoop software components were properly installed and configured. This test case was used as a baseline test to verify that Hadoop MapReduce jobs could be repeatedly run and rerun with consistent results. Baseline job timing was recorded.

### **3.2.2 Test case 2**

This test case was performed by running the same MapReduce jobs performed in test case 1. For this test, a disk was failed in each DCS3700 array, resulting in a total of three failed drives, one drive on each of three DCS3700 storage arrays. MapReduce job timing was recorded for comparison to those of the baseline runs.

### **3.2.3 Test case 3**

This test case was performed to demonstrate the possibility to recover the Hadoop file system after the data corruption at the HDFS NameNode. The loss was simulated by halting the NameNode server while a TeraSort MapReduce job was running and deleting the metadata directory. An HDFS check was performed before and after the recovery efforts and compared to verify file system integrity. A complete TeraSort MapReduce job was rerun to demonstrate that the entire file system was intact and that results consistent with previous runs would be achieved.

## **3.3 Test environment configurations**

This section summarizes the configurations of the various components in the Hadoop solution, as tested.

### **3.3.1 DCS3700 storage arrays**

Each of three DCS3700 storage arrays had the following configuration:

- ▶ Vendor: IBM
- ▶ Storage system: DCS3700 with Dual-active Intelligent controllers
- ▶ Disk type: Sixty 2 TB 7200 rpm NL-SAS 3.5 disks
- ▶ Host interface: Four 6 Gb SAS interface ports
- ▶ Controller firmware: Version 7.8 (minimum requirement for solution)
- ▶ Storage Manager: Version 10.8 (minimum requirement for solution)

Each DCS3700 storage array supported four Hadoop DataNode servers through a single SAS connection.

Each array allocated eight volume groups (RAID sets). The supported RAID type is RAID 5. Each volume group was composed of seven disks. Two volume groups were assigned to each DataNode server.

The remaining four disks in the array were designated as “hot spares” in case of disk failure.

One volume was configured to use all available storage space in each volume group. Each volume was configured with a segment size of 512 KB to maximize use of disk streaming I/O capabilities. Two volumes were mapped as LUNs to each of the Hadoop DataNode servers to which they were assigned.

In this test environment, the seven-disk RAID 5 volume group yielded a single LUN with approximately 10.913 TB of usable storage. Two LUNs per node gave each server approximately 21.826 TB of storage space.

Table 3-1 lists the DCS3700 configuration parameters.

*Table 3-1 DCS3700 storage configuration parameters*

Storage parameter	Recommended setting	Default setting	Description
Cache block size	32 KB	4 KB	This is the block size for DCS3700 read/write cache. 32 KB is the largest size available and was chosen to optimize I/O performance for the large block I/O generated by HDFS. This parameter is set for the entire DCS3700 array.
Read cache	Enabled	Enabled	Enables caching of prefetch data blocks to improve read performance. Read cache is set for each individual volume.
Write cache	Enabled	Enabled	Enables caching of writes, thereby improving write performance. Write cache is set for each individual volume.
Write cache without batteries	Disabled	Disabled	By default, batteries provide backup power to cache memory to avoid data loss during a power outage. This parameter is set at the volume level.
Write cache with mirroring	Disabled	Enabled	Maintains cache consistency between controllers to enable controller failover, but results in less available write cache and increased operational overhead. The Big Data Networked Storage Solution for Hadoop does not use the controller failover capability, so this capability is disabled. It is enabled or disabled on a per-volume basis.
Dynamic cache read prefetch	Enabled	Enabled	Provides dynamic, intelligent read-ahead for improved read performance. Enabled at the volume level.
Volume segment size	512 KB	128 KB	The amount of I/O written to a single disk drive before moving to the next drive in a RAID array. The largest segment size available was chosen to optimize data layout for the large block I/O generated by HDFS. The segment size is set for each individual volume.

### 3.3.2 Hadoop servers

Fifteen IBM x3530 M4 servers were employed in the solution tested. The server utilization is shown in Table 3-2.

Table 3-2 Server utilization

Server function	Quantity
Hadoop NameNode	1
Hadoop secondary NameNode	1
Hadoop JobTracker node	1
Hadoop DataNode, TaskTracker	12

Each of the servers, as tested, had the following configuration:

IBM System x3530 M4 servers

- ▶ CPU: Dual Intel Xeon E5-2407 4C/8T 2.2 GHz
- ▶ RAM: 6x8 GB DDR3 1333 MHz (48 GB total)
- ▶ Internal disk: 1 TB SATA disk drive (quantity 2)
- ▶ SAS adapter: LSI 9200-8e SAS controller 6 GB 8-port (used by DataNodes only)
- ▶ Network interfaces:
  - Two GbE (1 Gbps) integrated ports
  - One 10 GbE 2-port NIC
- ▶ Power: Dual power supply
- ▶ OS: Red Hat Linux Enterprise Server (RHEL), version 5 update 6

**Note:** All server firmware and BIOS were upgraded to the latest versions. LSI SAS adapter minimal firmware and drivers were as follows:

- ▶ Firmware: 9.00.00.00 (rom525fx.bin)
  - ▶ BIOS: 6.30.00.00
  - ▶ X86-BIOS: 07.07.00.00
- ▶ Linux driver: 9.00.00.00 (mpt2sas.ko)
- ▶ LUN partitioning
- The Linux **parted** utility was used for LUN partitioning. We used **gpt** labels because our LUNs were larger than 2 TB in size. For proper alignment with the underlying RAID 5 configuration, we started our partitions (using **mkpart**) at 6144 sectors and ended them at 12.0 TB to use the entire LUN. We also specified **xfs** as our file system type (see 3.3.3, “Local file system configuration” on page 24). The following command was used to partition a LUN with the `/dev/sdc` device name:

```
# parted -s /dev/sdc mklabel gpt mkpart /dev/sdc1 xfs 6144s 12.0 TB
```

### 3.3.3 Local file system configuration

As previously stated, HDFS uses native local file systems for actual block storage. For Linux Hadoop configurations, **ext3** has been popular; however, much progress has been made in the area of file system performance. We found **xfs** to be a good match for our data

storage configuration in terms of performance and reliability. Lab tests showed that a log size of 128 MB performed well with our configuration. We also found use of “lazy-count” logging to be good for performance. Lazy-count logging reduces the overhead of file system logging by safely reducing the frequency of superblock metadata updates. Stripe unit (su), stripe width (sw), and real-time section size of the file system (extsize) were tuned to match our underlying RAID subsystem and I/O patterns.

The following example of the command that is used to create a file system with label name DISK1 on a partition with device name /dev/sdc1:

```
/sbin/mkfs.xfs -f -L DISK1 -l size=65536b,sunit=256,lazy-count=1 -d sunit=1024,swidth=6144 /dev/sdc1
```

### 3.3.4 XFS file system mount options (/etc/fstab entries)

Example 1 shows the /etc/fstab entries used for mounting our XFS file systems:

*Example 1 Sample /etc/fstab entries*

---

```
LABEL=DISK1 /disk1 xfs allocsize=128m,noatime,nobarrier,nodiratime 0 0
LABEL=DISK2 /disk2 xfs allocsize=128m,noatime,nobarrier,nodiratime 0 0
```

---

Use of the noatime and nodiratime options results in access file and directory time stamps not being updated. These options are commonly used, and IBM suggests them for HDFS.

Write barriers make sure of the file system integrity for devices using write caches, even during a power loss. This is not necessary with battery-backed write cache, which is standard with the DCS3700 array. That being the case, we are able to improve write performance by using the nobarrier option to disable write barriers.

Finally, setting the allocsize option to 128 MB helped to optimize performance with HDFS I/O. The allocsize option sets the preallocation size for buffered I/O.

### 3.3.5 IBM N series N3220 storage

The N3220 storage array, used for backup storage of Hadoop HDFS metadata, was configured as follows:

- ▶ Controllers: Quantity 1
- ▶ Disks: 900 GB SAS (quantity 7)
- ▶ Network: Onboard 1 GB Ethernet port
- ▶ Licenses: NFS
- ▶ OS: Data ONTAP version 8.0.2 (operating in 7-Mode)

### 3.3.6 Mount options to be used on NameNode and secondary NameNode servers

The following sample /etc/fstab entry is for the N3220 volume, where st1N3220-11 is the name of the N series system:

```
st1N3220-11:/vol/vol1 /mnt/fsimage_bkp nfs rw,rsize=65536,wsiz=65536,noatime,soft
```

The rw options enable read/write I/O with the mounted volume. The rsize and wsize options were set to 64 KB for performance. The soft option was used according to Hadoop best practices to make sure that the failure of the NFS server (N series system) would not result in any type of hang-up on the NameNode.

### 3.3.7 Linux kernel and device tuning

For our test we used the `/etc/sysctl.conf` settings described in Table 3-3.

Table 3-3 Linux `/etc/sysctl.conf` settings used in tests

Parameter	Actual setting / Default value	Description
<code>net.ipv4.ip_forward</code>	0 / 0	Controls IP packet forwarding.
<code>net.ipv4.conf.default.rp_filter</code>	1 / 0	Controls source route verification.
<code>net.ipv4.conf.default.accept_source_route</code>	0 / 1	Does not accept source routing.
<code>kernel.sysrq</code>	0 / 1	Controls the system request debugging functionality of the kernel.
<code>kernel.core_uses_pid</code>	1 / 0	Controls whether core dumps append the PID to the core file name. Useful for debugging multithreaded applications.
<code>kernel.msgmnb</code>	65536 / 16384	Controls the maximum size of a message, in bytes.
<code>kernel.msgmax</code>	65536 / 8192	Controls the default maximum size of a message queue.
<code>kernel.shmmax</code>	68719476736 / 33554432	Controls the maximum shared segment size, in bytes
<code>kernel.shmall</code>	4294967296 / 2097512	Controls the maximum number of shared memory segments, in pages
<code>net.core.rmem_default</code>	262144 / 129024	Sets the default OS receive buffer size.
<code>net.core.rmem_max</code>	16777216 / 131071	Sets the max OS receive buffer size.
<code>net.core.wmem_default</code>	262144 / 129024	Sets the default OS send buffer size.
<code>net.core.wmem_max</code>	16777216 / 131071	Sets the max OS send buffer size.
<code>net.core.somaxconn</code>	1000 / 128	Maximum number of sockets the kernel can serve at one time. Set on NameNode, secondary NameNode, and JobTracker.
<code>fs.file-max</code>	6815744 / 4847448	Sets the total number of file descriptors.
<code>net.ipv4.tcp_timestamps</code>	0 / 1	Turns off the TCP time stamps.
<code>net.ipv4.tcp_sack</code>	1 / 1	Turns on select ACK for TCP.
<code>net.ipv4.tcp_window_scaling</code>	1 / 1	Turns on the TCP window scaling.
<code>kernel.shmmni</code>	4096 / 4096	Sets the maximum number of shared memory segments.
<code>kernel.sem</code>	250 32000 100 128 / 250 32000 32 128	Sets the maximum number and size of semaphore sets that can be allocated
<code>fs.aio-max-nr</code>	1048576 / 65536	Sets the maximum number of concurrent I/O requests.
<code>net.ipv4.tcp_rmem</code>	4096 262144 16777216 / 4096 87380 4194304	Sets min, default, and max receive window size.
<code>net.ipv4.tcp_wmem</code>	4096 262144 16777216 / 4096 87380 4194304	Sets min, default, and max transmit window size.
<code>net.ipv4.tcp_syncookies</code>	0 / 0	Turns off the TCP syncookies.

Parameter	Actual setting / Default value	Description
sunrpc.tcp_slot_table_entries	128 / 16	Sets the maximum number of in-flight RPC requests between a client and a server. This value is set on the NameNode and secondary NameNode to improve NFS performance.
vm.dirty_background_ratio	1 / 10	Maximum percentage of active system memory that can be used for dirty pages before dirty pages are flushed to storage. Lowering this parameter results in more frequent page cache flushes to storage, providing a more constant I/O write rate to storage. This gives better storage performance for writes.
fs.xfs.rtorstep	254 / 1	Increases the number of files to be written to an xfs allocation group before moving to the next allocation group.

Additional tuning on the Linux servers is included in the settings described in Table 3-4. The sample commands are for block device /dev/sdb. These settings must be implemented for all block devices used for HDFS data storage. These changes are not persistent across reboots, so they must be reset after each reboot. A good way to accomplish this is to include the commands in the Linux /etc/rc.local file.

Table 3-4 Additional tuning for block devices

Description of setting	Suggested value	Command-line example
Block device request queue length	512	echo 512 > /sys/block/sdb/queue/nr_requests
Block device queue depth	254	echo 254 > /sys/block/sdb/device/queue_depth
Block device readahead	1024	/sbin/blockdev --setra 1024 /dev/sdb

We also found that the deadline I/O scheduler performed better than the default CFQ scheduler. Setting the scheduler can be done in the /boot/grub/grub.conf file on the Linux server. The sample grub.conf file in Example 3-2 demonstrates how this setting can be implemented:

Example 3-2 Sample grub.conf

---

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xp
m.gz hiddenmenu
title Red Hat Enterprise Linux Server (2.6.18-238.el5) root (hd 0,0)
    kernel /vmlinuz-2.6.18-238.el5 ro root=/dev/VolGroup00/LogVol00 rhgb quiet
    elevator=deadline initrd /initrd-2.6.18-238.el5.img
```

---

### 3.3.8 Linux SAS control depth settings with procedure

For optimal performance, we increased the queue depth settings on the host-side SAS controllers by using the `lsiutil` utility, as follows:

1. Log in to the attached server as the root user and execute the following command:

```
./lsiutil.x86_64
```

You are prompted to choose a device. The choices resemble the choices in Figure 3-3.

	Port Name	Chip Vendor/Type/Rev	MPT Rev	Firmware Rev	IOC
1.	/proc/mpt/ioc0	LSI Logic SAS1068E B3	105	011e0000	0
2.	/proc/mpt/ioc0	LSI Logic SAS2008 03	200	09000000	0

Figure 3-3 Device selections

2. Choose the LSI Logic SAS2008 entry, which in this case is device 2. See Figure 3-4.

The following menu will appear.	
1.	Identify firmware, BIOS, and/or FCode
2.	Download firmware (update the FLASH)
4.	Download/erase BIOS and/or FCode (update the FLASH)
8.	Scan for devices
10.	Change IOC settings (interrupt coalescing)
13.	Change SAS IO Unit settings
16.	Display attached devices
20.	Diagnostics
21.	RAID actions
23.	Reset target
42.	Display operating system names for devices
43.	Diagnostic Buffer actions
45.	Concatenate SAS firmware and NVDATA file s
59.	Dump PCI config space
60.	Show non-default settings
61.	Restore default settings
66.	Show SAS discovery errors
69.	Show board manufacturing information
97.	Reset SAS link, HARD RESET
98.	Reset SAS link
99.	Reset port
e	Enable expert mode in menus
p	Enable paged mode
w	Enable logging

Figure 3-4 LSI Logic SAS2008 entry

3. On the main menu, select an option: 1-99; e, p, or w; or 0 to quit
4. Choose item 13 (Change SAS IO Unit Settings).
5. When prompted to enter a value for SATA Maximum Queue Depth, enter 255 and press the Return key:

```
SATA Maximum Queue Depth: [0 to 255, default is 32] 255
```



6. When prompted to enter a value for SAS max Queue Depth, Narrow, enter 256 and press the Return key:  
SAS Max Queue Depth, Narrow: [0 to 65535, default is 32] **256**
7. When prompted to enter a value for SAS Max Queue Depth, Wide, enter 2048 and press Return:  
SAS Max Queue Depth, Wide: [0 to 65535, default is 128] **2048**
8. Accept the defaults for Device Missing Report Delay and Device Missing I/O ZDelay by pressing Enter key at each prompt.
9. At the Select a Phy prompt, press Enter to quit.
10. At the main menu, enter 99 to reset the port.

This is required for implementing changes that you just made (Figure 3-5).

Device Missing Report Delay: [0 to 2047, default is 0]						
Device Missing I/O Delay: [0 to 255, default is 0]						
PhyNum	Link	MinRate	MaxRate	Initiator	Target	Port
0	Enabled	1.5	6.0	Enabled	Disabled	Auto
1	Enabled	1.5	6.0	Enabled	Disabled	Auto
2	Enabled	1.5	6.0	Enabled	Disabled	Auto
3	Enabled	1.5	6.0	Enabled	Disabled	Auto
4	Enabled	1.5	6.0	Enabled	Disabled	Auto
5	Enabled	1.5	6.0	Enabled	Disabled	Auto
6	Enabled	1.5	6.0	Enabled	Disabled	Auto
7	Enabled	1.5	6.0	Enabled	Disabled	Auto
8	Enabled	1.5	6.0	Enabled	Disabled	Auto
Select a Phy: [0-7, 8=AllPhys, RETURN to quit]						
Main menu, select an option: [1-99 or e/p/w or 0 to quit] <b>99</b>						

Figure 3-5 Menu for resetting a port

A message is displayed, indicating that the port is being reset.

11. When prompted again for a menu option, enter 0 to quit. You are then prompted to select a device. As before, enter 0 to quit, as shown in Figure 3-6:

```
Resetting port...

Main menu, select an option: [1-99 or e/p/w or 0 to quit] 0

Port Name          Chip Vendor/Type/Rev    MPT Rev    Firmware Rev    IOC
1. /proc/mpt/ioc0  LSI Logic SAS1068E B3    105        011e0000        0
2. /proc/mpt/ioc0  LSI Logic ASA2008 03     200        05000d00        0

Select a device: [1-2 or 0 to quit] 0
#
```

Figure 3-6 Resetting a port

### 3.3.9 Hadoop parameter settings

The `hadoop-env.sh` configuration file typically contains commands to set environment variables for the Hadoop environment.

The only required change to the default settings in this file is to set the `JAVA_HOME` environmental variable for Hadoop, as shown in the following example:

```
export JAVA_HOME=/usr/java/jdk1.6.0_30
```

To improve cluster efficiency, we changed `HADOOP_HEAPSIZE` from the default value of 1000 MB to 2000 MB with the following `hadoop-env.sh` entry:

```
export HADOOP_HEAPSIZE=2000
```

The `core-site.xml` configuration file contains core configuration information used by all components in the Hadoop cluster.

Table 3-9 on page 33 lists the `core-site.xml` parameter settings used in our test environment.

Table 3-5 Hadoop core-site.xml parameter settings used in test

Option name	Actual setting / Default value	Description
fs.default.name	hdfs:// 10.61.189.64:8020/ [Default Value: file:///]	Name of the default file system specified as a URL (IP address or host name of the NameNode along with the port to be used).
webinterface.private.actions	true / false	Enables or disables certain management functions in the Hadoop web user interface, including the ability to kill jobs and modify job priorities.
fs.inmemory.size.mb	200 / 100	Memory in MB to be used for merging map outputs during the reduce phase.
io.file.buffer.size	262144 / 4096	Size in bytes of the read/write buffer.
topology.script.file.name	► /etc/hadoop/conf/topology_ script ► Default value is null	Script used to resolve the subordinate nodes' name or IP address to a rack ID. Used to invoke Hadoop rack awareness. The default value of null results in all slaves being given a rack ID of /default-rack.
topology.script.number.args	1 / 100	Sets the maximum acceptable number of arguments to be sent to the topology script at one time.
hadoop.tmp.dir	► /home/hdfs/tmp ► Default value is /tmp/hadoop-\${user.name}	Hadoop temporary directory storage.

Parameters used to configure HDFS are contained in the `hdfs-site.xml` file. Table 3-6 shows the settings used in testing the IBM Open Solution for Hadoop.

Table 3-6 *HDFS-site.xml parameter settings used in test*

Option name	Actual setting / Default Value	Description
dfs.name.dir	<ul style="list-style-type: none"> <li>▶ <code>/local/hdfs/namedir,/mnt/fsimage_bkp</code></li> <li>▶ Default is <code>\${hadoop.tmp.dir}/dfs/name</code></li> </ul>	<p>Path on the local file system where the NameNode stores the name space and transaction logs persistently. If this is a comma-delimited list of directories (as used in this configuration), then the name table is replicated in all of the directories for redundancy.</p> <p><b>Note:</b> The <code>/mnt/fsimage_bkp</code> directory is a location on NFS-mounted N series storage where NameNode metadata is mirrored and protected, a key feature of our Hadoop solution.</p>
dfs.hosts	<ul style="list-style-type: none"> <li>▶ <code>/etc/hadoop-0.20/conf/dfs_hosts</code></li> <li>▶ Default is null</li> </ul>	Specifies a list of machines authorized to join the Hadoop cluster as a DataNode.
dfs.data.dir	<ul style="list-style-type: none"> <li>▶ <code>/disk1/data,/disk2/data</code></li> <li>▶ Default is <code>\${hadoop.tmp.dir}/dfs/data</code></li> </ul>	Directory path locations on the DataNode local file systems where HDFS data blocks are stored.
fs.checkpoint.dir	<ul style="list-style-type: none"> <li>▶ <code>/home/hdfs/namesecondary1</code></li> <li>▶ Default is <code>\${hadoop.tmp.dir}/dfs/namesecondary</code></li> </ul>	Path to the location where checkpoint images are stored (used by the secondary NameNode).
dfs.replication	2 / 3	HDFS block replication count. Hadoop default is 3. We use 2 in the IBM Hadoop solution.
dfs.block.size	134217728 (128MB) / 67108864	HDFS data storage block size in bytes.
dfs.namenode.handler.count	128 / 10	Number of server threads for the NameNode.
dfs.datanode.handler.count	64 / 3	Number of server threads for the DataNode.
dfs.max-repl-streams	8 / 2	Maximum number of replications a DataNode is allowed to handle at one time.
dfs.datanode.max.xcievers	4096 / 256	Maximum number of files a DataNode can serve at one time.

Table 3-7 lists the `mapred-site.xml` file settings used in our test environment. The parameters defined in that file are used to control MapReduce processing and to configure the JobTracker and TaskTrackers.

Table 3-7 *The mapred-site.xml parameters used in testing the IBM Open Solution for Hadoop*

Option name	Actual setting / Default value	Description
mapred.job.tracker	<ul style="list-style-type: none"> <li>▶ <code>10.61.189.66:9001</code></li> <li>▶ Default is local</li> </ul>	JobTracker address as a URI (JobTracker IP address or host name with port number)
mapred.local.dir	<ul style="list-style-type: none"> <li>▶ <code>/disk1/mapred/local,/disk2/mapred/local</code></li> <li>▶ Default is <code>\${hadoop.tmp.dir}/mapred/local</code></li> </ul>	Comma-separated list of the local file system where temporary MapReduce data is written.

Option name	Actual setting / Default value	Description
mapred.hosts	<ul style="list-style-type: none"> <li>▶ /etc/hadoop-0.20/conf/mapred.hosts</li> <li>▶ Default is null</li> </ul>	Specifies the file that contains the list of nodes allowed to join the Hadoop cluster as TaskTrackers.
mapred.system.dir	<ul style="list-style-type: none"> <li>▶ /mapred/system</li> <li>▶ Default is \$hadoop.tmp.dir/mapred/system</li> </ul>	Path in HDFS where the MapReduce framework stores control files
mapred.reduce.tasks.speculative execution	false / true	Enables the JobTracker to detect slow-running map tasks, assign them to run in parallel on other nodes, use the first available results, and then kill the slower running map tasks.
mapred.tasktracker.reduce.tasks.maximum	5 / 2	Maximum number of reduce tasks that can be run simultaneously on a single TaskTracker node
mapred.tasktracker.map.tasks.maximum	7 / 2	Maximum number of map tasks that can be run simultaneously on a single TaskTracker node.
mapred.child.java.opts	-Xmx1024m / -Xmx200m	Java options passed to the TaskTracker child processes (in this case, 2 GB defined for heap memory used by each individual JVM).
io.sort.mb	340 / 100	Total amount of buffer memory allocated to each io.sort.mb 340 / 100 merge stream while sorting files on the mapper, in MB.
mapred.jobtracker.task scheduler	<ul style="list-style-type: none"> <li>▶ org.apache.hadoop.mapred.FairScheduler</li> <li>▶ Default is org.apache.hadoop.mapred.JobQueue.TaskScheduler</li> </ul>	JobTracker task scheduler to use (in this case, use the FairScheduler)
io.sort.factor	100 / 10	Number of streams to merge at once while sorting files.
mapred.output.compress	false / false	Enables/disables output file compression.
mapred.compress.map.output	false / false	Enables/disables map output compression.
mapred.output.compression.type	block / record	Sets output compression type
mapred.reduce.slowstart.completed.maps	0.05 / 0.05	Fraction of the number of map tasks that should be complete before reducers are scheduled for the MapReduce job
mapred.reduce.tasks	<ul style="list-style-type: none"> <li>▶ 40 for 8 DataNodes</li> <li>80 for 16 DataNodes</li> <li>120 for 124 DataNodes</li> <li>▶ Default is 1</li> </ul>	Total number of reduce tasks to be used by the entire cluster per job
mapred.map.tasks	<ul style="list-style-type: none"> <li>▶ 56 for 8 DataNodes</li> <li>112 for 16 DataNodes</li> <li>120 for 24 DataNodes</li> <li>▶ Default is 2</li> </ul>	Total number of map tasks to be used by the entire cluster per job
mapred.reduce.parallel.copies	64 / 5	Number of parallel threads used by reduces tasks to fetch outputs from map tasks

Option name	Actual setting / Default value	Description
mapred.compress.map.output	false / false	Enables/disables map output compression.
mapred.inmem.merge.threshold	0 / 1000	Number of map outputs in the reduce TaskTracker memory at which map data is merged and spilled to disk.
mapred.job.reduce.input.buffer.percent	1 / 0	Percent usage of the map outputs buffer at which the map output data is merged and spilled to disk.
mapred.job.tracker.handler.count	128 / 10	Number of JobTracker server threads for handling RPCs from the task trackers
tasktracker.http.threads	60 / 40	Number of TaskTracker worker threads for fetching intermediate map outputs for reducers.
mapred.job.reuse.jvm.num.tasks	- 1/1	Maximum number of tasks that can be run in a single JVM for a job. A value of -1 sets the number to unlimited.
mapred.jobtracker.restart.recover	true / false	Enables job recovery after restart.

## 3.4 Test software inventory

Table 3-8 summarizes the software used by our Hadoop solution and additional tools used for monitoring.

*Table 3-8 Test software inventory*

Software or application	Version
IBM InfoSphere BigInsights Software	Version 2.0
Hadoop test tools TeraGen, & TeraSort	Part of IBM InfoSphere BigInsights
Java	1.6.0_30
Storage Manager client for Windows	10.80

## 3.5 Solution test results

Table 3-9 lists the final test results for each test case.

*Table 3-9 Solution test results*

Test case ID	Test case name	Duration	Pass/Fail (P/F)
1	Tuning run	Multiple runs over 3 days	P
2	Full functionality with fault injection	3 iterations	P
3	NameNode metadata recovery	1 iteration	P





## Conclusions and details of test cases

The Big Data Networked Storage Solution for Hadoop reference architecture is optimized for node-storage balance, reliability, performance, and storage capacity and density. Organizations that use Hadoop often use traditional server-based storage with inefficient, hard-to-scale, internal direct-access storage (DAS). The IBM BigInsights reference design employs the managed DAS model, with higher scalability and lower total cost of ownership (TCO). The lower cost is from choosing the number of clusters and storage you want and not spending money for features you do not need. It is also from nonproprietary lower-cost options and by decoupling compute nodes from storage, preventing unnecessary purchases of compute nodes for storage-intensive workloads.

The depth and breadth of IBM expertise spans a wide range of enterprise software, hardware, and services, enabling firms who partner with IBM to approach their big data projects with confidence and clarity.

See the following sources for more information:

- ▶ See details of BigInsights and connect with its community:  
<http://www.ibm.com/developerworks/wiki/biginsights>.
- ▶ Learn about IBM big data solutions:  
<http://www.ibm.com/software/data/bigdata/>
- ▶ Learn more about Hadoop and IBM InfoSphere BigInsights:  
<http://www.ibm.com/software/data/infosphere/hadoop/>

## 4.1 Test case details

This section lists the test details for executing the solution. The test scenarios start with the initial tuning and then progress through the baseline, fault injection and NameNode metadata recovery.

### 4.1.1 Test case 1-A: Initial tuning

Test case 1-A is for initial tuning:

- ▶ Test case ID: 1-A
- ▶ Test type: Initial
- ▶ Execution type: Manual
- ▶ Duration: Multiple runs, one day total

This test runs short TeraGen and TeraSort jobs (5 - 10 minutes) to aid in initial component validation and configuration tuning.

#### Prerequisites

HDFS components must be started.

#### Setup procedures

Remove any previous data artifacts from the HDFS system, before each run.

#### Execution procedures

Use included Apache TeraGen and TeraSort. Start from the JobTracker node.

#### Expected results

TeraGen and TeraSort jobs complete without error.

#### Additional information

Use integrated web or UI tools to monitor Hadoop tasks and file system.

### 4.1.2 Test case 1-B: Full functionality as baseline

Test case 1-B is for a baseline full functionality:

- ▶ Test case ID: 1-B
- ▶ Test type: Full function
- ▶ Execution type: Automated
- ▶ Duration: Multiple runs, one day total

Runs a TeraGen job with a duration greater than 10 minutes to generate a substantial data set. Runs a TeraSort job on the previously created data set.

#### Prerequisites

This test has the following prerequisites:

- ▶ HDFS components must be started.
- ▶ Test case 1-A completed successfully.



## Setup procedures

Remove any previous HDFS artifacts before each run.

## Execution procedures

The test is run as follows:

1. Use included TeraGen and TeraSort. Start from the JobTracker node.
2. Use the same TeraGen and TeraSort parameters during all iterations.

## Expected results

The results should be as follows:

- ▶ Proper output results received from the TeraSort Reduce stage.
- ▶ No tasks on individual task nodes (DataNodes) failed.
- ▶ File system (HDFS) maintains integrity and is not corrupted.
- ▶ All test environment components are still running.

## Additional information

Use integrated web and UI tools to monitor Hadoop tasks and file system.

### 4.1.3 Test case 2: Full functionality with fault injection

Test case 2 is for full functionality with fault injection:

- ▶ Test case ID: 2
- ▶ Test type: Full functionality with fault injection
- ▶ Execution type: Manual
- ▶ Duration: Multiple runs, one day total

Runs a TeraGen job with duration greater than 10 minutes to generate a substantial data set.  
Runs a TeraSort job on the previously created data set.

## Prerequisites

This test has the following prerequisites:

- ▶ HDFS components must be started.
- ▶ Test case 1-B completed successfully.

## Setup procedures

The setup steps are as follows:

1. Remove any previous HDFS artifacts prior to each run.
2. Make sure that all components are restored to a non-faulted condition.

## Execution procedures

The test is run as follows:

1. Use included Apache TeraGen and TeraSort. Start from the JobTracker node.
2. Use the same TeraGen and TeraSort parameters during all iterations.
3. After TeraSort indicates that has reached approximately 50% of the execution time, fault a disk in each of four controllers on the three DCS3700 arrays.

## Expected results

The results should be as follows:

1. Proper output results received from the TeraSort Reduce stage.
2. No tasks on individual task nodes (DataNodes) fail.
3. File system (HDFS) has maintained integrity and is not corrupted.

## Additional information

Consider the following information:

1. Use integrated web and UI tools to monitor Hadoop tasks and file system.
2. Use Ganglia to monitor Linux server in general.

### 4.1.4 Test case 3: NameNode metadata recovery

Test case 3 is for NameNode metadata recovery:

- ▶ Test case ID: 3
- ▶ Test type: Recovery
- ▶ Execution type: Manual
- ▶ Duration: 2 days

This scenario tests the failure of the Hadoop NameNode, the corruption of the metadata and the procedures to recover the cluster from these failures.

## Prerequisites

The previous test cases (1-A, 1-B, and 2) completed successfully.

## Setup procedures

The setup steps are as follows:

1. Repeat test 1-B.
2. Note the total reduce-input record count from the TeraSort job.
3. Run the **hadoop fsck /** command and note the following information:
  - Healthy status of NDFS
  - Number of files
  - Total size of the file system
  - Corrupt blocks

## Execution procedures

The procedure has two primary steps:

1. Do the following steps:
  - a. Repeat the TeraSort job.
  - b. While it is running, halt the NameNode to simulate loss of HDFS metadata.
  - c. Delete the metadata directory.
  - d. Shut down all Hadoop daemons on all servers in the cluster.
  - e. Open the namenode again and make sure the HDFS services fail because of unavailability of metadata.
2. Do the following steps:
  - a. Rebuild the master NameNode. Modify the `hdfs.site.xml` file to mount the NFS-based backup image (fsimage) and secondary location.
  - b. Start the NameNode daemon on the NameNode server.

- c. Restart the whole cluster using the `start-all.sh` script.
- d. From any node in the cluster, use the **`hadoop dfs -ls`** command to verify that the data file created by TeraGen exists.
- e. Check the total amount of HDFS storage used.
- f. Run the **`hadoop fsck /`** command to compare the results recorded before the NameNode was halted.
- g. Run TeraSort against the file originally generated by TeraGen and monitor for successful completion with no errors.

### Expected results

Output from the TeraSort should match the original output from test case 1-B.

### Additional information

Consider the following information:

- ▶ Use integrated web and UI tools to monitor Hadoop tasks and file system.
- ▶ Use Ganglia to monitor Linux server in general.



# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *IBM System Storage DCS3700 Introduction and Implementation Guide*, SG24-8037
- ▶ *IBM System Storage N series Software Guide*, SG24-7129
- ▶ *Implementing IBM InfoSphere BigInsights on System x*, SG24-8077
- ▶ *Better Business Decisions at a Lower Cost with IBM InfoSphere BigInsights*, TIPS0934

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Other publications

These publications are also relevant as further information sources:

- ▶ *Hadoop for Dummies Special Edition*. Robert D. Schneider; published by John Wiley & Sons, Canada, Ltd., 2012
- ▶ *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*, Dirk Deroos, Tom Deutsch, Chris Eaton, George Lapisand, and Paul Zikopoulos

<http://public.dhe.ibm.com/common/ssi/ecm/en/im114296usen/IML14296USEN.PDF>

## Online resources

These websites are also relevant as further information sources:

- ▶ IBM System Storage DCS3700  
<http://www.ibm.com/systems/storage/disk/dcs3700/>
- ▶ IBM System Storage N3000 Express  
<http://www.ibm.com/systems/storage/network/n3000/appliance/>
- ▶ IBM big data solutions  
<http://www.ibm.com/software/data/bigdata/>
- ▶ IBM InfoSphere BigInsights  
<http://www.ibm.com/software/data/infosphere/biginsights/>

- ▶ What is Hadoop  
<http://www.ibm.com/software/data/infosphere/hadoop/>
- ▶ What is the Hadoop Distributed File System (HDFS)?  
<http://www.ibm.com/software/data/infosphere/hadoop/hdfs/>
- ▶ Hadoop Cluster Setup web page  
[http://hadoop.apache.org/docs/stable/cluster\\_setup.html](http://hadoop.apache.org/docs/stable/cluster_setup.html)

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)





# Big Data Networked Storage Solution for Hadoop



**Redpaper™**

**Learn about the IBM big analytics storage solutions**

**Explore reference architecture based on Hadoop**

**Follow test cases that demonstrate scalability**

This IBM Redpaper provides a reference architecture, based on Apache Hadoop, to help businesses gain control over their data, meet tight service level agreements (SLAs) around their data applications, and turn data-driven insight into effective action.

Big Data Networked Storage Solution for Hadoop delivers the capabilities for ingesting, storing, and managing large data sets with high reliability. IBM InfoSphere Big Insights provides an innovative analytics platform that processes and analyzes all types of data to turn large complex data into insight.

IBM InfoSphere BigInsights brings the power of Hadoop to the enterprise. With built-in analytics, extensive integration capabilities, and the reliability, security and support that you require, IBM can help put your big data to work for you.

This IBM Redpaper publication provides basic guidelines and best practices for how to size and configure Big Data Networked Storage Solution for Hadoop.

**INTERNATIONAL  
TECHNICAL  
SUPPORT  
ORGANIZATION**

**BUILDING TECHNICAL  
INFORMATION BASED ON  
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
**[ibm.com/redbooks](http://ibm.com/redbooks)**