Redbooks

ibm.com/redbooks

IBM Spectrum Computing Solutions

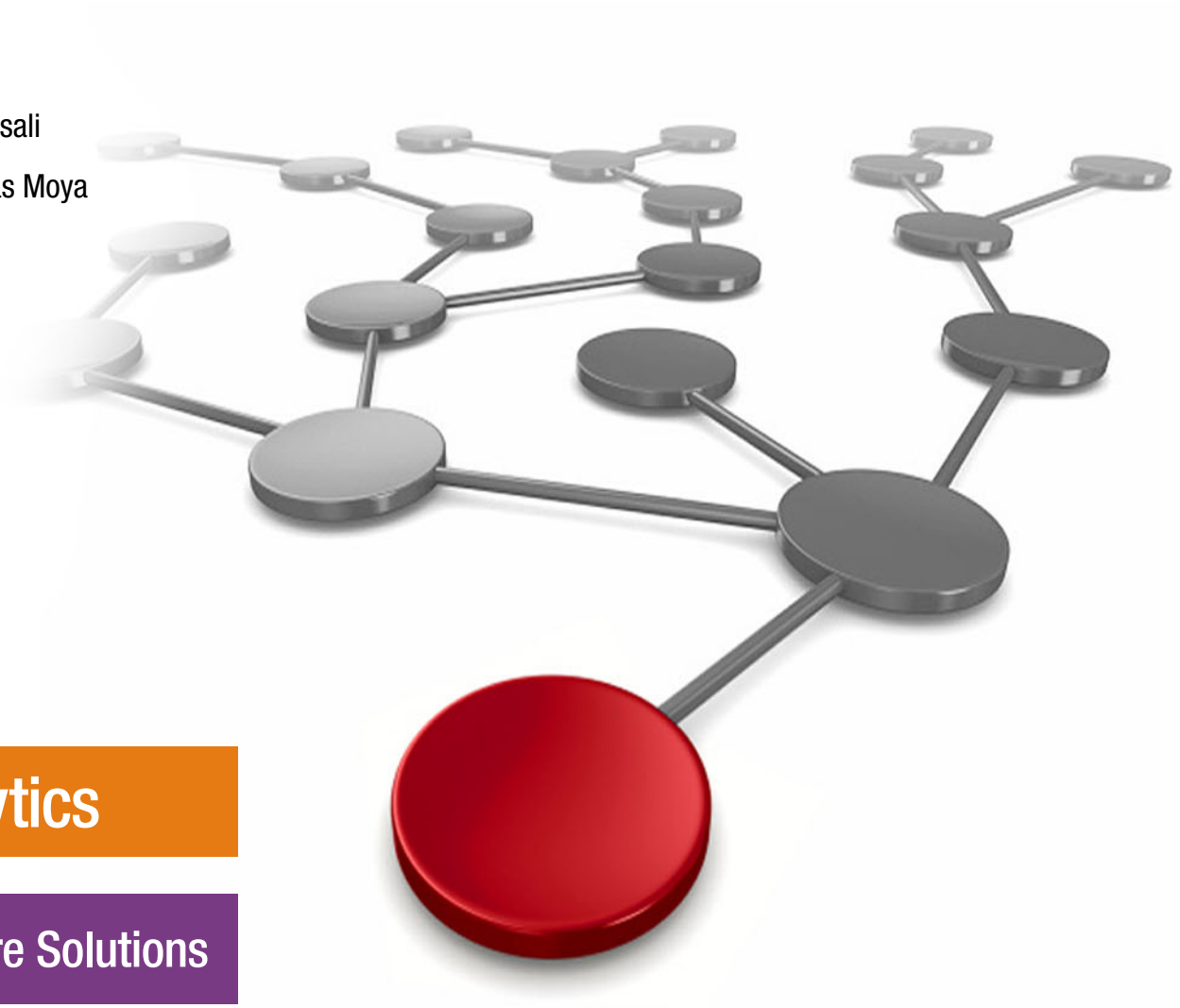
Dino Quintero

Daniel de Souza Casali

Eduardo Luis Cerdas Moya

Federico Fros

Maciej Olejniczak



Analytics

Infrastructure Solutions

IBM

Redbooks

International Technical Support Organization

IBM Spectrum Computing Solutions

May 2017

SG24-8373-00

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

First Edition (May 2017)

This edition applies to:
Red Hat Linux ppc64 Little Endian version 7.2
IBM Spectrum Scale version 4.2.1
IBM Cluster Foundation version v4.2.2
IBM Spectrum Conductor with Spark version 2.2
IBM Spectrum MPI version 10

Contents

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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, MD-NC119, Armonk, NY 10504-1785, US

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 jurisdictions 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 provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 actual people or business enterprises 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

AIX®	IBM Spectrum Archive™	POWER®
BigInsights®	IBM Spectrum Conductor™	Power Systems™
Bluemix®	IBM Spectrum Control™	POWER8®
DB2®	IBM Spectrum Protect™	PowerHA®
developerWorks®	IBM Spectrum Scale™	PowerLinux™
GPFS™	IBM Spectrum Storage™	PowerVM®
IBM®	IBM Spectrum Symphony™	Redbooks®
IBM Cloud Managed Services®	IBM Spectrum Virtualize™	Redbooks (logo) ®
IBM Flex System®	Linear Tape File System™	Symphony®
IBM Spectrum™	LSF®	Tivoli®
IBM Spectrum Accelerate™	Passport Advantage®	XIV®

The following terms are trademarks of other companies:

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.

Microsoft, 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.

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Preface

This IBM® Spectrum Computing Solutions Redbooks® publication is a follow-up book to update each of the available offerings that are part of the IBM portfolio of Cloud, analytics, technical computing, and high-performance computing (HPC) solutions for our clients. This publication delivers descriptions of the available offerings from IBM Spectrum™ Computing (formerly IBM Platform Computing) that address challenges for our clients in each industry. We include a few implementation and testing scenarios with selected solutions.

This publication helps strengthen the position of IBM Spectrum Computing solutions with a well-defined and documented deployment model within an IBM Systems environment. This deployment model offers clients a planned foundation for dynamic cloud infrastructure, provisioning, large-scale parallel HPC application development, cluster management, and grid applications.

This book is targeted toward technical professionals (consultants, technical support staff, IT Architects, and IT Specialists) responsible for delivering cost effective cloud services, big data, and analytics solutions on IBM Power Systems™ helping to uncover insights among client's data so they can take actions to optimize business results, product development, and scientific discoveries.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Dino Quintero is a Complex Solutions Project Leader and an IBM Level 3 Certified Senior IT Specialist with the ITSO in Poughkeepsie, New York. His areas of expertise include enterprise continuous availability, enterprise systems management, system virtualization, technical computing, and clustering solutions. He is an Open Group Distinguished IT Specialist. Dino holds a Master of Computing Information Systems degree and a Bachelor of Science degree in Computer Science from Marist College.

Daniel de Souza Casali is an IBM Cross Systems Senior Certified and has been working at IBM for 13 years. Daniel works for the Systems and Technology Group in Latin America as an IBM Software-Defined Infrastructure IT Specialist. Daniel holds an Engineering degree in Physics from the Federal University of São Carlos (UFSCar). His areas of expertise include UNIX, SAN networks, IBM Disk Subsystems, clustering, cloud, and big data analytics solutions.

Eduardo Luis Cerdas Moya is a CompTIA Cloud Essentials and CompTIA Cloud + certified specialist, currently working as a Cloud Infrastructure Specialist for IBM Cloud Managed Services® in IBM Costa Rica, where he joined in 2015. His current areas of expertise include IBM AIX®, Linux, IBM PowerHA®, virtualization on Power, OpenStack, and containerized environments using Docker and Kubernetes, software-defined networking (SDN), and Network Functions Virtualization (NFV).

Federico Fros is an IT Specialist who currently works for IBM Global Technologies Services leading the UNIX and Storage team for the IBM Innovation center in Uruguay. He has worked in IBM for more than 12 years, including 10 years of experience in IBM Power Systems and IBM Storage. He is an IBM Certified Systems Expert for UNIX and High Availability. His areas of expertise include AIX, Linux, PowerHA, IBM PowerVM®, Cloud Computing, and IBM Spectrum Storage™ Family including IBM Spectrum Scale™.

Maciej Olejniczak is an IBM Certified IT Expert in Actualizing IT Solutions and The Open Group Certified Master IT Specialist. He has been in IBM since 2008. He has more than 15 years of experience in the areas of systems management, networking, big data computing, and software. He is a cross-functional technology consultant and the Global Technical Account Advocate. He is the author of complex IT solutions for public sector clients, and the co-author of four IBM Redbooks publications. He achieved a master level in implementing all activities that transform IT from a vision to a working solution, and is an IBM Academy of Technology talented mentor for new hires and IT Specialists. His focus has been on security and Platform Computing for the last three years.

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

Wade Wallace
International Technical Support Organization, Poughkeepsie Center

Nicolas Joly, Anthony Frery, Duane Witherspoon
IBM USA

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 help increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run two - 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

Comments welcome

Your comments are important to us.

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

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

  ibm.com/redbooks

- Send your comments in an email:

  redbooks@us.ibm.com

- Mail your comments:

  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

1

Introduction to IBM Spectrum Computing

This chapter describes today's challenges in the IT industry associated mostly with the massive amount of data that is generated everyday and the complexity of designing, implementing, and managing the scale-out infrastructure needed to mine new insights from all this data.

In this chapter, the following topics are described:

- Overview
- Big data and resource management
- The new era for high-performance computing (HPC)
- Hybrid cloud bursting
- The big data challenge
- IBM Spectrum Cluster Foundation
- IBM Spectrum Computing

1.1 Overview

This section provides an overview of the challenges the IT is facing, and what solutions are available to help manage these challenges.

1.2 Big data and resource management

Most of the problems in the IT industry are associated with the massive amount of data that is generated every second. How to obtain this data, where to store the data, how to analyze the data, and maybe the most important question, how to provision and manage the scale-out infrastructure, software components, and be able to do it at affordable costs are many of the principal problems or challenges for organizations today.

Resource management began in the fifties when the first operating system for IBM computers were written for IBM customers. These customers did not want to have their multi million dollar machines sitting idle while operators set up jobs manually, so they wrote a mechanism for maintaining or manage a queue of jobs.

Today there are different needs and different jobs but more than ever the data explosion has resulted in a fairly complex collection of scale-out infrastructure and software components that all needs to be managed, tuned, and balanced to work well together.

A resource manager provides the underlying system infrastructure to enable multiple applications to operate within a shared resource infrastructure. A resource manager manages the computing resources for all types of workload.

The complexity, interdependence, and business need for the IT infrastructure has become so critical, that it must be managed as a whole.[1]

Fundamentally, software-defined architecture enables automation of infrastructure configuration that supports rapid deployment and is aligned to real-time application requirements.

1.3 The new era for high-performance computing (HPC)

The OpenPOWER initiative has sharpened IBM focus on HPC in particular.[2] IBM is building one of the fastest CPUs in the world, which has the tightest connection with accelerators (such as NVIDIA and Xilinx), networking devices (such as Mellanox), and storage (such as flash), using the CAPI and NVLink interfaces. The IBM POWER® roadmap is backed by a strong semiconductor process roadmap, as is demonstrated with the recent breakthrough in 7nm technology. IBM has a comprehensive set of products for the HPC market as follows:

- High-performance IBM POWER processors and systems, with tight integration with accelerators, networking, and storage devices
- High-performance IBM Spectrum Scale data management software, and IBM Elastic Storage Server (ESS) storage solutions built with embedded computing servers
- IBM Spectrum LSF® and IBM Spectrum Symphony™ data center management, scheduling, and orchestration software

[1] Infrastructure resource management: A holistic approach.
[2] OpenPOWER for HPC and big data analytics data centers.

IBM in collaboration with OpenPOWER partners has developed a strong OpenPOWER HPC roadmap (Figure 1-1) consisting of a series of progressively advanced systems to be introduced in 2015 and 2017. Several system vendors that are part of the OpenPOWER Foundation will also offer a variety of POWER-based servers that integrate these innovations.

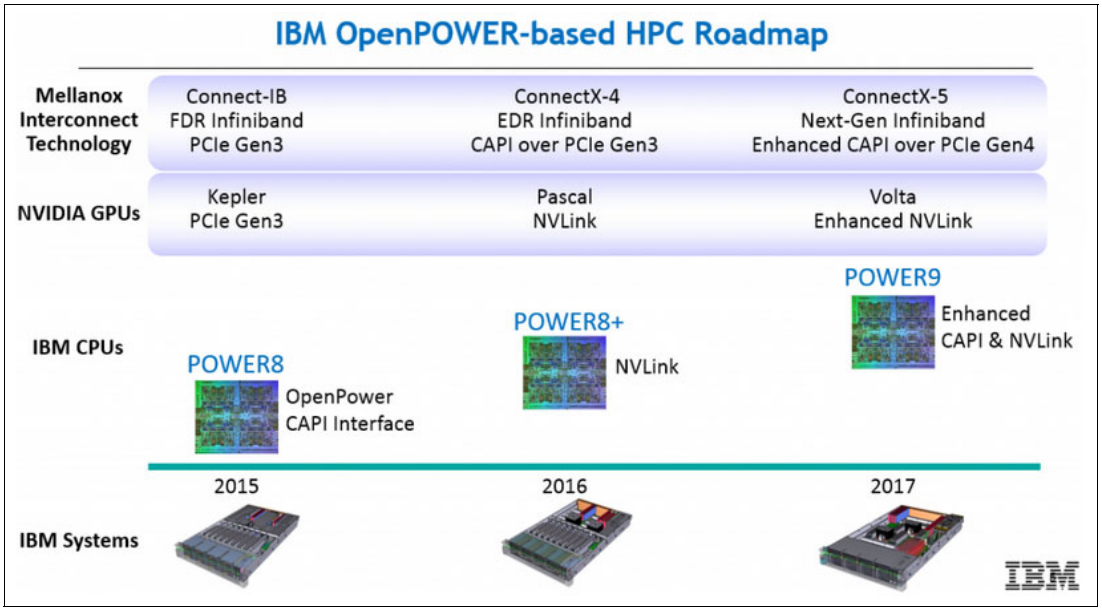


Figure 1-1 OpenPOWER-based HPC roadmap

1.4 Hybrid cloud bursting

Cloud bursting is an application deployment model in which an application runs in a private cloud or data center and bursts into a public cloud when the demand for computing capacity spikes. The advantage of such a hybrid cloud deployment is that an organization only pays for extra compute resources when they are needed.

One example is the holiday shopping season. In this time of the year, the shopping activity increases more than any other time in the year so that the web traffic increases and the load in the infrastructure. The problem now becomes how to process the load and how to provide enough storage capacity during these peak times.

The most common solution to this problem is to anticipate it and rearrange the resources, shut down some unnecessary services or buy some additional hardware and software to provide the extra capacity needed, but this is not always easy or possible.

Fortunately there is another option called *cloud bursting* or *burst capacity* where an application can be deployed locally and then burst to the cloud to meet peak demands, or the application can be moved to the public cloud to free up local resources for business-critical applications. These peaks can be treated as an operational expense (OPEX) rather than the capital expense (CAPEX) needed to buy new hardware.

To potentialy achieve better ROI, no more purchasing servers to run them at 50% capactiy. For peak performance, it is better to know where the data is and to have it closer.

1.5 The big data challenge

Data is growing exponentially and it is becoming the new natural resource as you can read from the next facts:

- More data was created in the last two years than in the whole human history
- By 2020, world data is expected to grow from 4.4 zettabytes to 44 zettabytes (44 trillion Gigabytes)
- In five years, there will be 50 billion smart, connected devices in the world

These are only some examples of the data explosion, and organizations need to manage and analyze this data to (Figure 1-2):

- Obtain results faster
- Share resources more efficiently
- Improve overall cost-effectiveness of the IT infrastructure

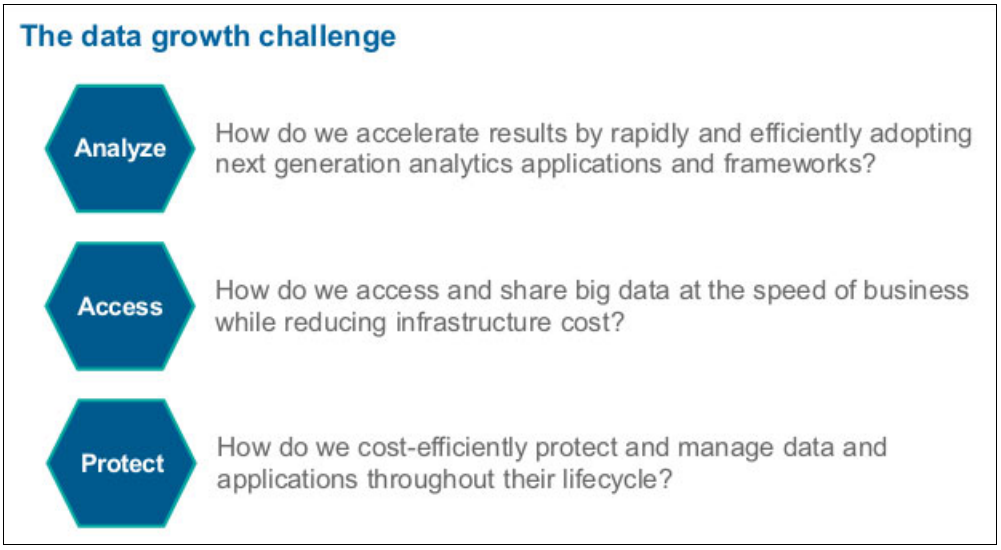


Figure 1-2 Data growth challenge

Organizations are challenged to do more with less and this is fundamentally impacting the way that the IT infrastructure is deployed and managed.

1.5.1 Hadoop

Apache Hadoop is a major element in the collection of data. It assists in the following functions:

- Collection of nearly limitless amounts of data
- Loading data without a schema
- Exploration
- Cleansing
- Organization
- Making the data available for analysis

Early implementations of Hadoop depended entirely on the batch-oriented processes of MapReduce, and its Structured Query Language (SQL) derivative Hive. The problem encountered by most organizations is that although MapReduce is useful for big-batch ingestion, where analytics can wait until loading and transformation is complete, the need for speed in some cases makes this approach impractical.

1.5.2 Apache Spark

The more rapid alternative to processing and exploring Hadoop data is Apache Spark. Spark is a memory-optimized data processing engine that can execute operations against Hadoop data (in HDFS) with better performance than MapReduce. It can also process data on other clustering environments such as Cassandra or even on its own clusters and files.

Spark delivers high performance access to dynamic, complex data collections and provides interactive functionality such as ad hoc query support in a scalable way. Spark supports programming in Java, Scala, Python, and R. For structured data access on Spark, there are several SQL-based options available including Apache Spark SQL.

1.5.3 Hadoop Distributed File System (HDFS)

Hadoop Distributed File System (HDFS) is a distributed file system that provides high-throughput access to application data. HDFS is designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as infrastructure for the Apache Nutch web search engine project. HDFS is part of the Apache Hadoop Core project.

1.5.4 Multi-tenancy

The term multi-tenancy refers to an architecture in which a single instance of software application serves multiple customers or tenants. Each customer or group of users who share a common access with specific privileges is called a *tenant*.

With a multitenant architecture, a software application is designed to provide every tenant a dedicated share of the instance including its data, configuration, user management, tenant individual functionality, and non-functional properties. Multi-tenancy contrasts with multi-instance architectures, where separate software instances operate on behalf of different tenants.

The use of multitenant *resources* avoid infrastructure spending, expensive application, and departmental silos and improve the utilization by sharing the infrastructure and at the same time protecting the SLAs.

Maintaining the SLA is important because downtime can cost money, failure to meet tight reporting deadlines can result in financial penalties or sanctions that impact the business.

IBM Spectrum Computing schedulers can address these challenges by providing no single point of failure, and continuing to run even during multiple hardware or software failures.

Optional solutions, such as IBM Spectrum LSF Process Manager, enable repetitive sequences of tasks to be automated with built-in error handling and recovery. By providing software infrastructure that is fault tolerant and avoids the problem of human error (the most frequent contributor to system downtime), you can increase availability and productivity to better meet deadlines and SLAs.

IBM Spectrum Computing addresses availability issues as well, helping to make cluster users more productive by ensuring cluster downtime is kept to a minimum. Features, such as the cluster file manager and repository snapshots, take the risk out of several tasks, such as software upgrades and patching. These features enable administrators to easily roll back to a last-known good configuration if anything goes wrong with a software update or patch installation, reducing downtime and providing peace of mind for users who need to make software changes.

IBM Spectrum LSF can take the relative power-efficiency of systems into account when scheduling work to ensure that workloads are deployed to the most cost-efficient systems. As a result, more work can be completed in time to meet business deadlines.

1.6 IBM Spectrum Cluster Foundation

IBM Spectrum Cluster Foundation is a lifecycle management solution for scale-out environments that provides:

- Faster time to cluster readiness
- Unified management interface
- Integrated and extensible monitoring, reporting, and alerting
- Reduced infrastructure and management costs
- Improved diagnostics with advanced log-file analytics and health checks
- Support for mixed computing environments
- In-place software upgrades
- Single infrastructure supporting multiple business needs

IBM Spectrum Cluster Foundation automates the creation of multiple scale-out environments on a shared infrastructure used by multiple teams. The software creates an agile environment for running both high-performance computing (HPC) and analytics workloads. By doing so, it allocates the appropriate resources to the correct workloads and consolidates disparate cluster infrastructures and multiple workload schedulers, which results in increased resource utilization, the ability to meet or exceed service level agreements, and reduced infrastructure and management costs. IBM Spectrum Cluster Foundation provides the ability to quickly create and manage multiple clusters on a shared IT infrastructure, and easily move resources between clusters as needed.

1.7 IBM Spectrum Computing

The new intelligent resource and workload management software, called IBM Spectrum Computing, is designed to make it easier for organizations to extract full value from data to accelerate performance-intensive analytics or machine learning. This technology can be used across industries to help sequence genomes for improved cancer treatment, or for engineers to design a championship-winning Formula One race car or bankers to personalize financial services to attract new customers.

The IBM Spectrum Computing platform offers new cognitive, resource-aware scheduling policies that help increase the utilization of existing compute resources, controlling costs and at the same time speeding results for high-performance computing, big data analytics, new generation applications and open source frameworks, such as Hadoop and Apache Spark.

IBM Spectrum Computing assists organizations with consolidating data center infrastructure and sharing resources across on-premise, cloud, or hybrid environments. IBM Spectrum Computing includes three new software solutions: IBM Spectrum Conductor™, IBM Spectrum LSF, and IBM Spectrum Symphony. The expanded software-defined computing portfolio complements the IBM Spectrum Storage software-defined storage family. The IBM Spectrum portfolio provides clients with a unique set of software-defined infrastructure capabilities.

1.7.1 IBM Spectrum Conductor with Spark

IBM Spectrum Conductor is designed to speed analysis of data. IBM Spectrum Conductor works with cloud applications and open source frameworks, speeding time to results by enabling increasingly complex applications to share resources, all while protecting and managing data throughout its lifecycle.

IBM Spectrum Conductor integrates Apache Spark. Hence IBM Spectrum Conductor with Spark simplifies the adoption of Apache Spark, an open source big data analytics framework, while delivering faster analytical results.

1.7.2 IBM Spectrum LSF

IBM Spectrum LSF helps accelerate research and design. IBM Spectrum LSF is a comprehensive workload management software featuring flexible and easy to use interfaces to help organizations accelerate research and design, and at the same time controlling costs through advanced resource sharing and improved utilization.

1.7.3 IBM Spectrum Symphony

IBM Spectrum Symphony is a highly scalable, high-throughput, low-latency workload and resource management software solution for compute and data-intensive analytics applications. IBM Spectrum Symphony can reallocate more than 1,000 compute engines per second to different workloads and, with sub-millisecond resource requirements per task, provides throughput of up to 17,000 tasks per second.

Note: On June 2, 2016, IBM Platform Computing was rebranded IBM Spectrum Computing, reflecting the portfolio alignment with IBM Spectrum Storage and enhancements to support new generation cloud-native applications and open source frameworks, such as Apache Spark.

2

IBM Spectrum Computing family

This chapter focuses on the IBM Spectrum Computing family of solutions and how they integrate with the IBM Spectrum Storage family to help solve the challenges described in Chapter 1, “Introduction to IBM Spectrum Computing” on page 1.

This chapter describes the IBM Software-Defined Infrastructure, which is a solution that helps you maximize hardware utilization. The technology also ensures an open approach to new solutions while maintaining enterprise business aspects. It also assists you in correctly handling heterogeneous needs from different lines of business, so maintaining the correct service level agreements (SLAs). SLAs are still a challenge that cannot be forgotten when implementing this new software-defined paradigm.

This chapter describes the following topics:

- IBM Software-Defined Infrastructure
- IBM Spectrum Computing
- IBM Spectrum Storage

2.1 IBM Software-Defined Infrastructure

IBM Software-Defined Infrastructure is based on a better utilization of resources. Through software, the information technology building blocks (compute, storage, and network) can be managed through a central control point.

Virtualization is the way the control software spreads workload across a given infrastructure. This virtualization means multiple applications sharing a single machine, using virtual machines, cgroups, or containers to limit usage. The concept is shown in Figure 2-1.

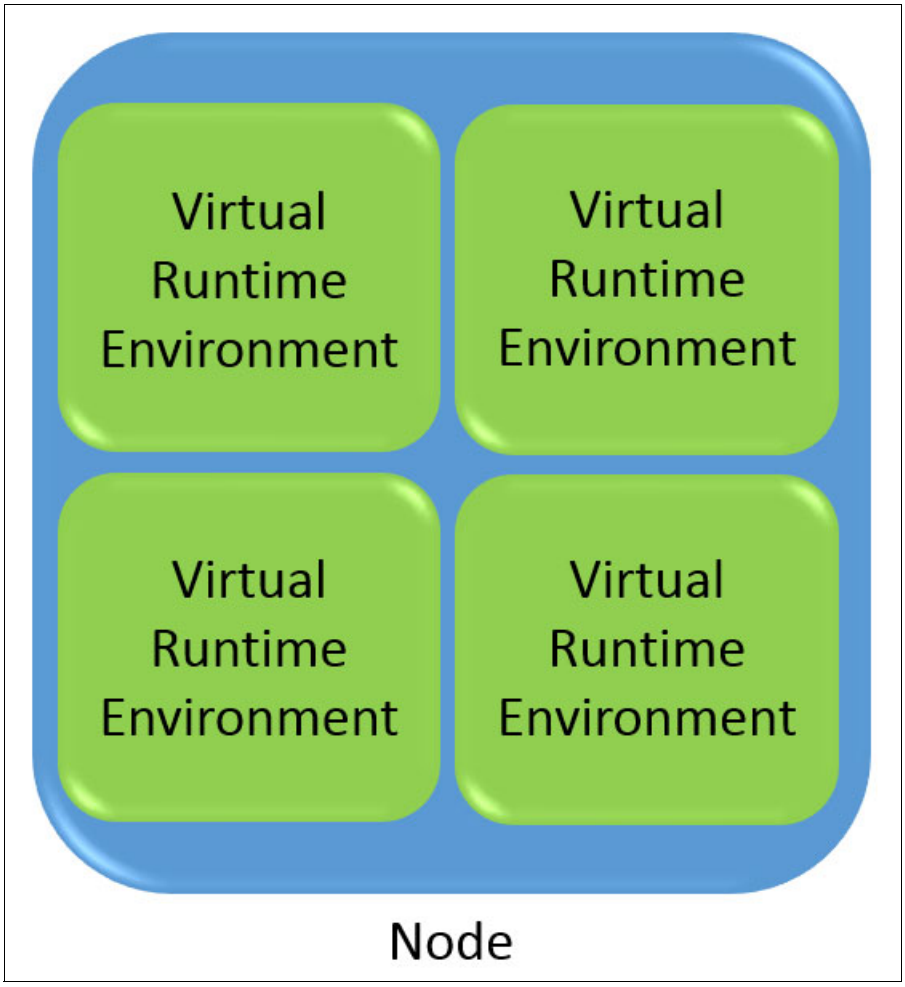


Figure 2-1 Virtualization within a single node

However, there is another approach that is not often mentioned as virtualization, but is widely used, where multiple hardware nodes can be managed as an entity and the application can run as a single virtual environment. This scale-out architecture can be seen in Figure 2-2.

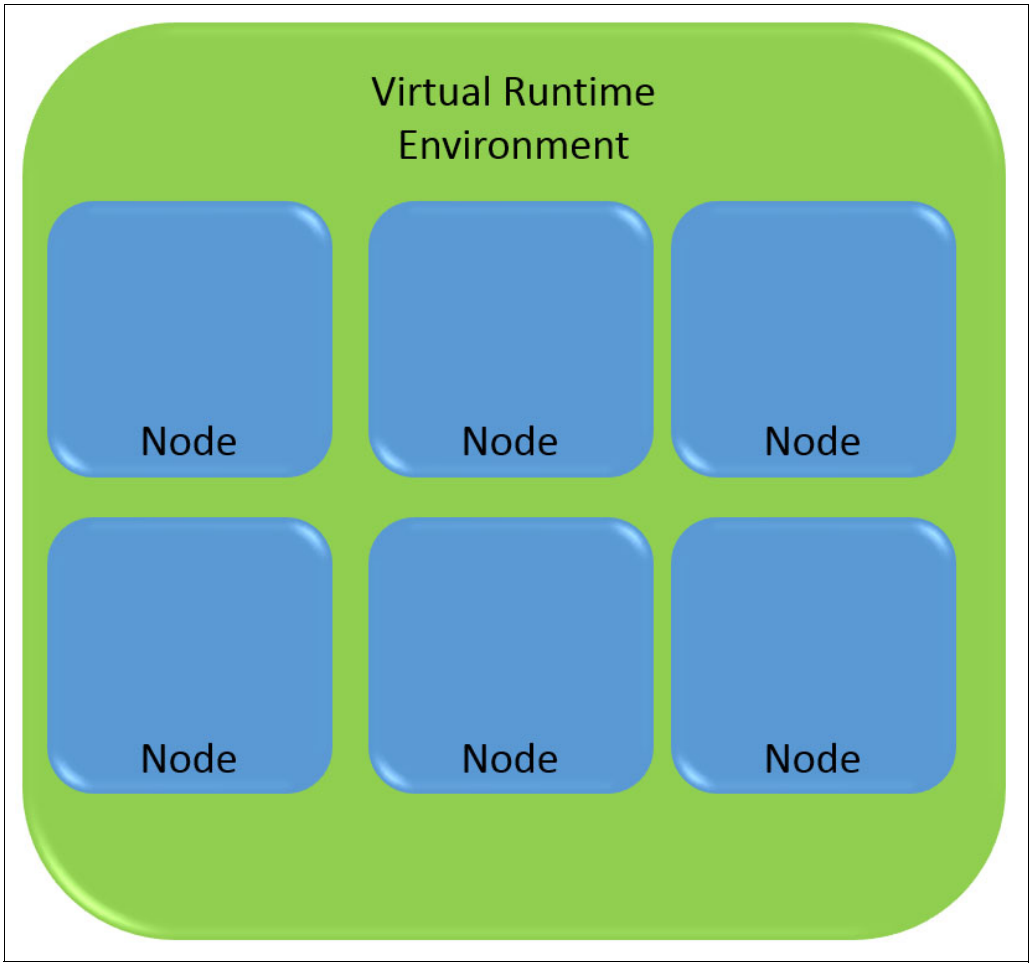


Figure 2-2 Scale-out or grid virtualization

In both cases, the application needs a hypervisor as an abstraction layer to make good use of the infrastructure. This book shows examples of both ways to use IBM Software-Defined Infrastructure.

The IBM Spectrum family addresses both compute and storage requirements. This book focuses on the compute part of the software-defined Infrastructure, but briefly describes the IBM Spectrum Storage products as illustrated in section 2.3, “IBM Spectrum Storage” on page 17 which shows how solutions can be built with the IBM Spectrum Computing family of offerings.

2.2 IBM Spectrum Computing

The approach that is used to solve new compute challenges, especially in analytics, are driving growth in clusters and grid solutions within the enterprise. It is common to have different clusters for different workloads and even for different departments. Many of these clusters have variable usage patterns where they are busy one day and inactive the next. Purchasing and managing multiple clusters increases cost to the organization.

In addition, each cluster can have replicated data or islands of data that is not available to the organization as a whole. Lastly, due to the cost of purchasing and maintaining these clusters, they can have insufficient total resources to provide the required qualities of services during busy times.

One way to overcome these issues is to build significantly fewer clusters where the resources can be used across the enterprise. In effect, what is needed is a grid hypervisor that finds a way to share resources.

The software requirements to be able to manage the physical resources, such as network, compute, and storage, or at least be aware of them so that constrains are addressed. Sharing resources allows for significant cost savings across the organization and the possibility to improve quality of service for each workload. It allows the organization to maintain fewer copies of data, and to make use of enterprise data as an enterprise asset. Lastly, this grid hypervisor needs to provide the appropriate levels of control and security.

The IBM Spectrum Compute family of products can be thought of as a *grid hypervisor* that has many specialized software products. These can be used across both bare metal and hypervisor-controlled instances. The IBM Spectrum Compute family can also manage the underlying infrastructure and present it to different workloads as a single logical compute unit.

This compute environment can then be divided again into groups of resources and use technologies like containers to subdivide even a single node and use it for multiple workloads. A macro block view of the abstraction layers over the infrastructure can be seen in Figure 2-3.

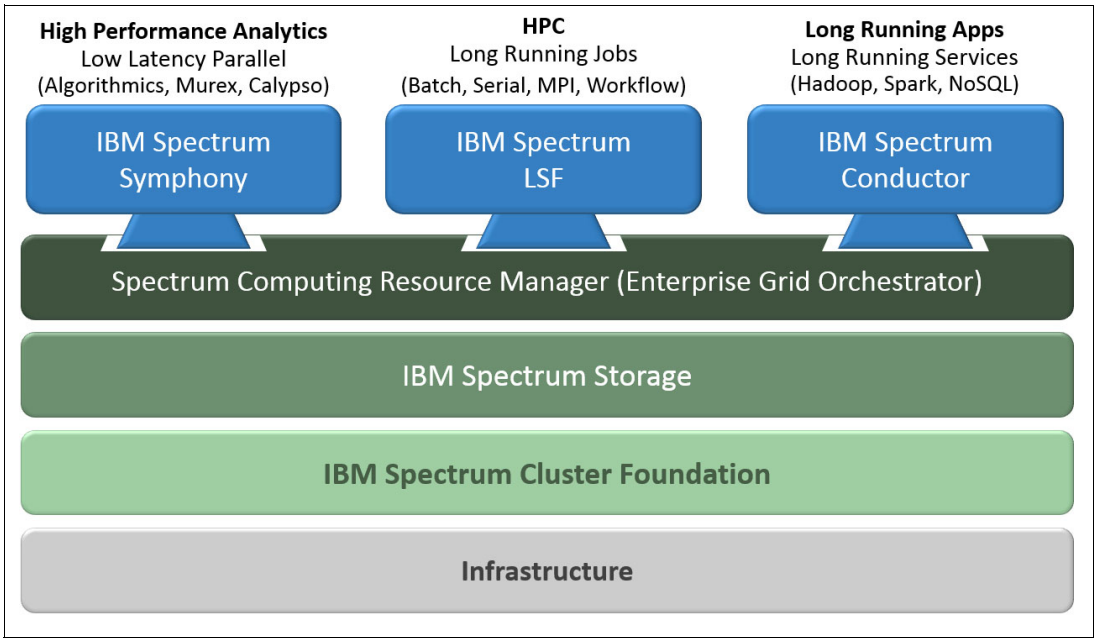


Figure 2-3 Abstraction layers for the grid hypervisor

Table 2-1 offers a closer look at the grid hypervisor for IBM Spectrum Computing.

Table 2-1 IBM Spectrum Computing family

Product	Hypervisor focus	Workload example
IBM Spectrum LSF	High-Performance Computing	Life sciences, oil and gas, structural analysis, fluid dynamics, and electronic design automation
IBM Spectrum Symphony	High Performance Analytics	Embarrassingly parallel workloads, financial risk analysis, and co-processors workloads
IBM Spectrum Conductor	New workloads applications and data-optimized platform	NoSQL databases, Spark, Hadoop, and containerized applications
IBM Cluster Foundation	Bare metal management	Deployment and management of clusters and grids from bare-metal to working grid hypervisor

2.2.1 IBM Spectrum LSF

IBM Spectrum LSF (formerly IBM Platform LSF) is a complete workload management solution for demanding High Performance Computing (HPC) environments. Efficient workload scheduling can help ensure that resources are effectively used, ultimately speeding time to market or time to solution.

As a result, workload managers are expected to perform efficiently and remain responsive, even in the most demanding environments.

IBM Spectrum LSF accelerates technical computing, providing up to 150 times greater throughput for faster time to result. See the benchmark white paper.

With a comprehensive set of intelligent, policy-driven scheduling features, IBM Spectrum LSF enables you to make the most of all your compute infrastructure resources and ensure optimal application performance. The highly scalable and available architecture allows you to schedule complex workloads, and manage up to petaflop-scale resources.

A family of add-ons helps you broaden the cluster hypervisor features and an overview is offered in Table 2-2.

Table 2-2 IBM Spectrum LSF add-on family

Add-on	Helps you
IBM Spectrum LSF Application Center	Manage your LSF job submission offering an intuitive graphical user interface (GUI). It provides ways of customizing templates for job submission helping the user focus on the business not on technical details.
IBM Spectrum LSF Process Manager	Organize your computational workflow, automating complex interdependent batch jobs to enhance your time to results on the growing simulation and analysis process.

Add-on	Helps you
IBM Spectrum LSF RTM	Create an extended operational dashboard that can help you on performance views and system availability issues helping you solve problems with a consolidated view of your environment.
IBM Spectrum LSF Analytics	Visualize and analyze massive amounts of IBM Spectrum LSF workload data. It enables managers, planners and administrators to correlate jobs, resource and license data easily for one or more clusters. It is your tool for data-driven decision making driving better cluster usage and capacity planning.
IBM Spectrum MPI	Provide a production-quality implementation of MPI accelerating application performance in distributed computing environments. It provides a familiar interface that is easily portable, it incorporates advanced CPU affinity features, dynamic selection of interface libraries, superior workload manager integrations leading to improved performance.
IBM Spectrum LSF License Manager	Create intelligent license sharing between local or global project teams. It helps you ensure license availability for the correct workload, user, and project, and that licenses are optimally used.
IBM Spectrum LSF Data Manager	Take control of data transfers helping you to improve data throughput and lower your costs. It helps you automate the transfer of data used by workloads running on different sites, bringing frequently used data closer to compute resources.
IBM Spectrum LSF Session Scheduler	Get the best of both worlds: responsiveness and job volume management. It is designed to work with IBM Spectrum LSF to provide high-throughput, low-latency scheduling for a wide range of workloads. It is suited to environments that run high volumes of short-duration jobs and where users require faster and more predictable job turnaround times.

A deeper view on IBM Spectrum LSF is available in Chapter 4, “IBM Spectrum LSF” on page 29.

2.2.2 IBM Spectrum Symphony

IBM Spectrum Symphony software helps you control the massive compute power needed for current and future grid computing systems that address problems that are so challenging and complex that a single scale up system cannot not achieve. This grid hypervisor can help you achieve breakthrough results in business and analytics by addressing challenges in parallel application development and deployment, and the computing infrastructure management that is needed for maintaining correct SLAs. Table 2-3 on page 15 shows the IBM Spectrum Symphony add-on family of solutions.

Although a conventional batch scheduler can schedule jobs in seconds or minutes, IBM Spectrum Symphony can schedule tasks in milliseconds. This key difference enables the solution to deliver faster, better quality results to support online or near real-time requirements, even while using a smaller amount of infrastructure.

It can support up to 5,000 compute nodes, 128,000 cores, and 300 applications. Designed with the flexibility to adapt when priorities change, it can reallocate more than 1,000 compute engines per second to different workloads in accordance with sharing policies and application priorities you define. This translates into better application performance, better utilization, and an ability to respond quickly to business demands.

To ensure correct SLAs, a fair-share scheduling scheme with up to 10,000 priority levels can be used for multiple jobs for a single application. Also, preemptive and resource threshold-based scheduling is available with runtime change management. Slot allocations change dynamically based on job priority and server thresholds, loaning, and borrowing.

Table 2-3 IBM Spectrum Symphony add-on family

Add-on	Helps you
Application Service Controller for IBM Spectrum Symphony	Create a generalized service controller for complex long-running application services. It is designed to address the requirements of a new generation of distributed application workloads that stem from the wide adoption of cloud technology, providing increased resiliency, and high availability. The Application Service Controller for IBM Spectrum Symphony is available on the Advanced edition only.
Desktop, server, and VM harvesting	To harvest nondedicated systems, including desktop, server, and virtual server machines. Virtual server harvesting support is available for clusters with Windows or Linux master hosts.
Intel Xeon Phi	Use Intel Xeon Phi devices as resources for applications that run on this kind of co-processor in your IBM Spectrum Symphony cluster.
GPU application support	Run GPU applications on Linux and Windows compute hosts in clusters with any supported Windows or Linux or UNIX master hosts.
Platform Symphony connector for Microsoft Excel	Support workloads that use Microsoft Excel 2007 and 2010 32-bit editions.
Platform Symphony connector for MATLAB	Support workloads that use MATLAB 2011A, MATLAB 2008B, 2012B (64-bit application), and 2013A on specific operating systems.
Open Source R support	Support running R applications using Open Source R version 2.14.2 on 64-bit Linux.

For detailed information about, and support for, the IBM Spectrum Symphony add-on family, see the add-ons website.

Also, more details about this topic are available in Chapter 5, “IBM Spectrum Symphony” on page 53.

2.2.3 IBM Spectrum Conductor

IBM Spectrum Conductor manages new classes of clouds, analytics, mobile, and social (CAMS) workloads, which are based on complex, long-running services. The major difference on other grid hypervisors is that it enables such services to run in a shared multitenant environment. It enables a wide variety of applications, including cloud-native ones such as MongoDB and Cassandra to share resources and coexist on the same infrastructure.

As seen in 2.1, “IBM Software-Defined Infrastructure” on page 10, this is exactly what is needed in a software-defined environment. By minimizing application silos and increasing resource utilization, IBM Spectrum Conductor provides the following benefits:

- Cuts costs by using a service orchestration framework for diverse workloads.
- Improves performance and efficiency with granular resource sharing.
- Simplifies management with consolidated cluster and application service tools to enable elastic services for these Cloud ready applications.

IBM Spectrum Conductor with Spark

IBM Spectrum Conductor with Spark is a multitenant solution for Apache Spark, enabling you to efficiently deploy and manage multiple Spark deployments in a single virtual hypervisor avoiding cluster sprawl. You can manage your Spark deployments independent of your infrastructure, or you can choose an integrated infrastructure solution, where system infrastructure and storage can be monitored and managed along with your Spark deployments.

IBM Spectrum Conductor with Spark eliminates resource silos that are tied to multiple Apache Spark implementations, it provides multitenancy through Spark instance groups, which are the equivalent to tenants on a Spark framework. You can run multiple instances of Spark, including different Spark versions, in a single and shared environment.

This solution can also be deployed as integrated infrastructure from bare metal. In this option, the solution provides infrastructure management from bare metal deployment of operating systems and shared storage with IBM Spectrum Scale, in addition to Spark deployment capabilities. The integrated solution delivers a complete platform, providing:

- Simplified administration with a single interface that provides management and monitoring across a scale-out, distributed infrastructure
- Improved data availability with global, shared access to ensure that data is available when it is needed
- Minimized deployment time by automating deployment of physical, virtual, and containerized resources

More on this topic can be seen in Chapter 6, “IBM Spectrum Conductor” on page 67.

2.2.4 IBM Cluster Foundation

IBM Spectrum Cluster Foundation automates the self-service assembly of multiple heterogeneous high-performance computing (HPC) and analytics clusters on shared compute infrastructure. The cluster manager creates a secure multi-tenant analytics and HPC cloud for users running technical computing and analytics workloads to dynamically create clusters, grids, and HPC clouds on demand, consolidate a scattered cluster infrastructure, increase hardware usage, gain access to larger cluster infrastructures, and rapidly deploy multiple heterogeneous HPC environments.

IBM Spectrum Cluster Foundation can deliver:

- Increases agility and innovation by enabling self-service provisioning of HPC and analytics clusters in minutes
- Decreases operating costs through increased usage of existing servers and increased operational efficiency (hundreds of servers per administrator)
- Reduces capital expenditure by reusing existing hardware resources
- Increases use of pooled resources by providing larger clusters and grids, and by reprovisioning nodes to meet the needs of the workload

IBM Spectrum Cluster Foundation has the following capabilities:

- Faster time to cluster deployment
- Reduces infrastructure management costs
- Improves administrator and user productivity
- Increases cluster up-time for high system availability
- Greater cluster provisioning capability
- On-demand self-service cluster provisioning
- Manages multiple separate clusters as a single resource pool
- Creates a secure multi-tenant environment
- Provisions physical, virtual, and hybrid physical-virtual clusters
- Dynamically grow and shrink a user's logical cluster size based on workload and resource allocation policy

IBM Spectrum Cluster Foundation runs on the current rack-based servers, and is supported on the most-recent generation of IBM POWER8® based systems, including OpenPOWER. It is also supported on industry-standard, generic IPMI-based x86-64 hardware. By prequalifying and certifying these platforms at scale, IBM can help you take the risk out of deploying mission-critical grid computing deployments.

2.3 IBM Spectrum Storage

Within the IBM Spectrum family, a whole software-defined storage solution is available too. Particularly, IBM Spectrum Scale fits well for the workloads that we focus on in this book These workloads are distributed and cloud-ready applications. So you can have an idea of the whole IBM Spectrum Storage family, Table 2-4 offers a high-level overview of the IBM Spectrum Storage family. It pairs descriptions with the products that provide the functions.

Table 2-4 IBM Spectrum Storage Family descriptions

	IBM Spectrum Storage family member	**Description**	**Former name**
	IBM Spectrum Control™	Automated control and optimization of storage and data infrastructure	IBM Tivoli® Storage Productivity Center, management layer of IBM Virtual Storage Center, and IBM Spectrum Control Base Edition (IBM Storage Integration Server)

	IBM Spectrum Storage family member	Description	Former name
	IBM Spectrum Protect™	Optimized data protection for client data through backup and restore capabilities	IBM Tivoli Storage Manager Suite for Unified Recovery, and IBM Spectrum Protect Snapshot
	IBM Spectrum Virtualize™	Core SAN Volume Controller function is virtualization that frees client data from IT boundaries	IBM SAN Volume Controller
	IBM Spectrum Accelerate™	Enterprise storage for cloud that is deployed in minutes instead of months	IBM XIV® as software
	IBM Spectrum Scale	Storage scalability to yottabytes and across geographical boundaries	IBM General Parallel File System (GPFS™)
	IBM Spectrum Archive™	Enables long-term storage of low activity data	IBM Linear Tape File System™ Enterprise Edition, Library Edition, and Single Drive Edition

IBM Spectrum Scale

IBM Spectrum Scale is used in the book as the underlying storage for analytics, so a subsection describing its capabilities is deserved. To handle massive unstructured data growth, the whole solution must scale seamlessly and at the same time matching data value to the capabilities and costs of different storage tiers and types.

It is a software-defined storage for high performance, large scale workloads (on-premises or in the cloud) that fits the best for the type of workload IBM Spectrum Computing is meant to manage. This scale-out storage solution provides file, object, and integrated data analytics for the following areas:

- Compute clusters (technical computing)
- Big data and analytics
- Hadoop Distributed File System (HDFS)
- Private cloud

3

IBM Spectrum Computing requirements

This chapter describes the hardware and software requirements for IBM Spectrum LSF, IBM Spectrum Symphony, and IBM Spectrum Conductor with Spark.

This chapter contains the following topics:

- IBM Spectrum LSF system requirements
- IBM Spectrum Symphony system requirements
- IBM Spectrum Conductor with Spark requirements
- Our lab test environment

3.1 IBM Spectrum LSF system requirements

This section describes the system requirements and compatibility details for IBM Spectrum LSF, specifically version 10.1.0.

3.1.1 Operating system support

Table 3-1 shows the operating systems support for IBM Spectrum LSF.

Table 3-1 Operating system support for IBM Spectrum LSF

Operating system	Version	CPU/hardware	Package name	Support	Edition
HP-UX	11.31 (IA64)	HP Integrity Servers (Itanium2)	lsf10.1_hpuxia64.tar.Z	Supported	Standard
IBM AIX	7.x	IBM Power 7/8	lsf10.1_aix-64.tar.Z	Supported	Standard
SUN Solaris on SPARC	10	SPARC 64 bit (T2)	lsf10.1_sparc-sol10-64.tar.Z	Supported	Standard
	11	SPARC 64 bit (T2)		Supported	Standard
Linux x64 kernel	Kernel 2.6, glibc 2.5 RHEL 5.x	X86/64	lsf10.1_linux2.6-glibc2.3-x86_64.tar.Z	Supported	Express, Standard
	Kernel 2.6, glibc 2.11 RHEL 6.x			Supported	Express, Standard
	Kernel 3.10, glibc 2.17 RHEL 7.x		lsf10.1_lnx310-lib217-x86_64.tar.Z (for kernels above 3.10)	Certified	Express, Standard
	Kernel 3.0, glibc 2.11 SLES 11.x			Supported	Express, Standard
	Kernel 3.11, glibc 2.18 SLES 12.x			Certified	Express, Standard
	Kernel 3.10, glibc 2.17 CentOS 7.x			Certified	Express, Standard
	Kernel 4.4, glibc 2.23 Ubuntu 16.04 LTS			Certified	Express, Standard
SGI Performance Suite	Kernel 2.6, glibc 2.3			Supported	Express, Standard
Generic Linux: Debian, Ubuntu and OEL	Kernel 2.6/3.0, glibc 2.3 or later			Supported	Express, Standard
Linux kernel on IBM Power LE (little endian)	Kernel 3.13, glibc 2.19 Ubuntu 14.04	IBM Power 8 LE	lsf10.1_lnx310-lib217-ppc64le.tar.Z	Supported	Standard
	Kernel 3.10, glibc 2.17 RHEL 7.x			Certified	Standard

Operating system	Version	CPU/hardware	Package name	Support	Edition
	Kernel 3.12, glibc 2.19 SLES 12.x			Certified	Standard
	Kernel 4.4, glibc 2.23 Ubuntu 16.04 LTS			Certified	Standard
Apple OS X (slave host only}	10.8	x86/64	lsf10.1_macosx.tar.Z	Supported	Standard
	10.9			Supported	Standard
Linux ARM	ARMv7	x1_358	lsf10.1_lnx36-lib215-armv7.tar.Z	Supported	Standard
	ARMv8	AArch64	lsf10.1_linux3.12-glibc2.17-armv8.tar.Z	Supported	Express, Standard, Advanced
Cray XE6/XT6/XC30	Kernel 2.6, glibc 2.3	X86/64	lsf10.1_lnx26-lib23-x64-cray.tar.Z	Certified	Standard
Microsoft Windows	Windows 2008 x64	X86/64	lsf10.1_win-x64.msi	Supported	Standard
	Windows 7 x64			Supported	Standard
	Windows 8.1 x64			Supported	Standard
	Windows 2008 R2 x64			Supported	Standard
	Windows HPC Server 2008			Supported	Standard
	Windows 2012 x64			Supported	Standard
	Windows 2012 R2 x64			Certified	Standard
	Windows 10 x64			Supported	Standard
	Windows server 2016 TP4 x64			Certified	Standard

Note: For more information about IBM Spectrum LSF requirements, see the IBM developerWorks article about this topic.

3.1.2 Hardware requirements for the master host

IBM Spectrum LSF has no minimum CPU requirement. For the platforms on which IBM Spectrum LSF is supported, any host with sufficient physical memory can run IBM Spectrum LSF as master host. Swap space is normally configured as twice the physical memory. IBM Spectrum LSF daemons in a cluster of Linux x86-64 use about 488 MB of memory when no jobs are running. Active jobs use most of the memory that IBM Spectrum LSF requires. See Table 3-2 for additional information about the master host sizing.

To achieve the highest degree of performance and scalability, use a powerful master host.

Note: If a Microsoft Windows host must be installed as the master host, only 2008 R2 Server and Windows 2012 R2 Server are suggested as LSF Master hosts.

Table 3-2 IBM Spectrum LSF master host requirements

Cluster size	Active jobs	Minimum required memory (typical)	Recommended server CPU (Intel, AMD, OpenPower, or equivalent)
Small (<100 hosts)	1,000	1 GB (32 GB)	Any server CPU
	10,000	2 GB (32 GB)	Recent server CPU
Medium (100 - 1000 hosts)	10,000	4 GB (64 GB)	Multi-core CPU (2 cores)
	50,000	8 GB (64 GB)	Multi-core CPU (4 cores)
Large (>1000 hosts)	50,000	16 GB (128 GB)	Multi-core CPU (4 cores)
	500,000	32 GB (256 GB)	Multi-core CPU (8 cores)

3.1.3 Server host compatibility

IBM Spectrum LSF V8.0.x, V8.3, and V9.1 or later servers are compatible with IBM Spectrum LSF V10.1 master hosts. All IBM Spectrum LSF V8.0.x, V8.3, and V9.1 or later features are supported by IBM Spectrum LSF V10.1 master hosts.

Important: To take full advantage of all the new features that are introduced in the current release of IBM Spectrum LSF, you must upgrade all hosts in your cluster.

3.2 IBM Spectrum Symphony system requirements

This section describes the system requirements for IBM Spectrum Symphony.

3.2.1 The Minimum hardware requirements for IBM Spectrum Symphony Developer Edition V7.2

Note: You can install IBM Spectrum Symphony Developer Edition on a maximum of two hosts.

Table 3-3 shows the IBM Spectrum Symphony Developer Edition V7.2 hardware requirements.

Table 3-3 IBM Spectrum Symphony Developer Edition V7.2 hardware requirements

Specification	Management (master) host	Compute host
Number of hosts	1	1
CPU power	2.4 GHz	2.4 GHz
RAM	4 GB	1 GB
Disk space (for installation)	1 GB	1 GB
Additional disk space (such as for applications or logs)	30 GB	10 GB in addition to capacity required by user programs

3.2.2 Software requirements

This section describes the software requirements for IBM Spectrum Symphony.

NET prerequisites (Windows only)

If you are planning on developing in .NET, ensure that your development host has the following software:

- Microsoft Visual Studio 2010
- .NET Framework 4.0, and 4.5

Java prerequisites

If you are planning on developing an IBM Spectrum Symphony workload in Java, ensure that your development host has the following:

- Java version (required): For detailed information, see the following website that lists supported operating systems.
- Ant version (recommended): 1.6.5.

Note: Java must be installed before IBM Spectrum Symphony Developer Edition is started.

If you are planning on developing a MapReduce workload in Java (available only on Linux 64-bit hosts), ensure that your development host has the prerequisite software installed. This includes setting `JAVA_HOME` to point to the directory where the JDK is installed.

Operating system requirements

For more information, see IBM Knowledge Center about operating system support for IBM Spectrum Symphony workload and MapReduce workload.

Note: MapReduce workload is only supported on Linux 64-bit operating systems.

Distributed file systems

IBM Spectrum Symphony Developer Edition supports the following distributed file systems for MapReduce workload:

- Apache Hadoop Distributed File System (HDFS) included with the following Hadoop MapReduce API 2.7.2
- Cloudera's Distribution including Apache Hadoop (CDH) 5.5.1
- IBM Spectrum Scale on Linux 64-bit and on Linux on POWER (IBM Spectrum Scale-FPO version 4.1.1 or 4.2.1)
- Appistry CloudIQ Storage 4.5.1.6

Note: You can develop and test MapReduce applications in IBM Spectrum Symphony Developer Edition using the stand-alone (local) mode in which the MapReduce flow runs on the local host in a single JVM. In this case, you do not require HDFS.

For more information about IBM Spectrum Symphony requirements, go to IBM Knowledge Center.

3.3 IBM Spectrum Conductor with Spark requirements

This section describes the requirements for IBM Spectrum Conductor with Spark.

3.3.1 Hardware requirements

IBM Spectrum Conductor with Spark v2.2 is supported on the following servers.

Table 3-4 on page 25 lists the minimum system requirements for running IBM Spectrum Conductor with Spark in production environments.

Table 3-4 Minimum hardware requirements for production environment

Requirement	Management hosts	Compute hosts	Notes
Number of hosts	Without high availability: 1 or more With high availability: 3 or more	1 or more	For high availability purposes, a quorum of eligible master hosts is required. A loss of quorum results in an inoperable cluster and potential loss of data. The minimum number of eligible hosts required is dynamically determined as `((number_of_management_hosts/2)+1)`. To allow for the loss of 1 management host, your cluster must have a minimum of 3 master eligible hosts when high availability is enabled.
Cores used	8	1 or more	Choose a modern processor with multiple cores. Common clusters use two to eight core machines. When it comes to choosing between faster CPUs or more cores, it is recommended to choose hosts with more cores.
CPU power	>=2.4 GHz	>=2.4 GHz	
RAM	64 GB	32 GB	In general, the more memory your hosts have, the better performance is.
Disk space to install	6 GB	6 GB	

Requirement	Management hosts	Compute hosts	Notes
Additional disk space (for Spark instance group packages, logs, etc.)	Can be 30 GB for a large cluster	1 GB*N slots + sum of service package sizes (including dependencies)	Disk space requirements depend on the number of Spark instance groups and the Spark applications that you run. Long running applications, such as notebooks and streaming applications, can generate huge amounts of data that is stored in Elasticsearch. What your applications log can also increase disk usage. Consider all these factors when estimating disk space requirements for your production cluster. For optimal performance, look at tuning how long to keep application monitoring data based on your needs.

Heap considerations for the elastic stack

The default Elasticsearch installation uses a 4 GB heap. Elasticsearch recommends that you assign 50 percent of available memory to Elasticsearch, but not cross 30.5 GB. Based on these recommendations, configure the Elasticsearch heap in IBM Spectrum Conductor with Spark to use ~6 - 8 GB. Further, the default garbage collector for Elasticsearch is Concurrent-Mark and Sweep (CMS). To prevent long stop-the-world pauses, do not configure the heap size to be higher than what the CMS garbage collector was designed for (~6 - 8 GB).

For information about changing the heap size, go to IBM Knowledge Center and read “How do I change the heap size for Elasticsearch?”

3.3.2 Software requirements

IBM Spectrum Conductor with Spark V2.2 is supported on the following operating systems.

On x86-based and Linux 64-bit servers, use one of the following options:

- Red Hat Enterprise Linux (RHEL) AS 7.1 and 7.2
- RHEL AS 6.5, or later
- SUSE Linux Enterprise Server 11 SP4
- SUSE Linux Enterprise Server 12
- Ubuntu Server 16.04 LTS

On Power Systems, POWER 64-bit LE, and OpenPower, use one of the following options:

- Red Hat Enterprise Linux (RHEL) 7.1 and 7.2
- SUSE Linux Enterprise Server 12
- Ubuntu 16.04 LTS

Note: The program's specifications and specified operating environment information can be found in documentation accompanying the program, if available, such as a readme file, or other information published by IBM, such as an announcement letter. Documentation and other program content is supplied only in the English language.

3.4 Our lab test environment

This section describes the environment, hardware, and software used for testing purposes while developing content for this publication.

Hardware

For testing purposes during the time of writing this book, POWER8 servers were used with eight Red Hat Enterprise Linux 7.2 servers instances. Six servers were used to create a multi-tenant environment using IBM Spectrum Conductor with Spark and IBM Spectrum Symphony.

The other two Red Hat servers were used to install and test IBM Spectrum Cluster Foundation in a high availability configuration, as described in Chapter 7, "Cluster management with IBM Spectrum Cluster Foundation" on page 103.

For the multi-tenant environment, we used partitions with 0.7 entitled cores, 7 virtual processors each, and 12 GB of memory. All of them were connected to two networks and with access to the internet. For a production environment, bare metal machines can be used for more CPU and memory availability for applications, such as the IBM Power System S822L. The environment configuration is shown in Figure 3-1.

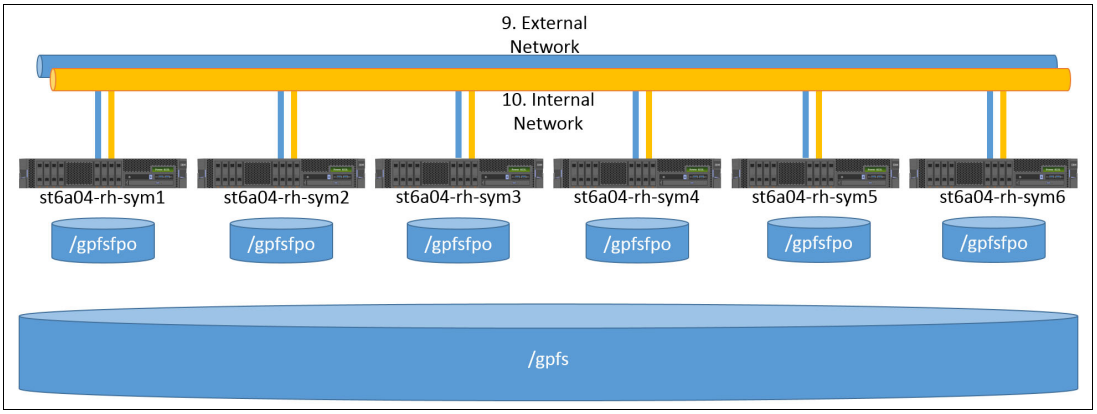


Figure 3-1 Test environment infrastructure diagram

Software

For this testing purpose, the following software versions were used:

- Red Hat Linux LE V7.2 for POWER
- IBM Spectrum Symphony V7.1.2
- IBM Spectrum Conductor with Spark V2.2
- MongoDB V3.5

4

IBM Spectrum LSF

This chapter describes the IBM Spectrum Load Sharing Facility (LSF) product family.

IBM Spectrum LSF is a powerful workload management platform for demanding, distributed high-performance computing (HPC) environments. IBM Spectrum LSF provides a comprehensive set of intelligent, policy-driven scheduling features that enable full utilization of your compute infrastructure resources, and ensure optimal application performance.

This chapter covers the following topics:

- IBM Spectrum LSF family overview
- IBM Spectrum LSF integration with Docker
- IBM Spectrum LSF Data Manager
- IBM Spectrum Symphony MapReduce Accelerator for IBM Spectrum LSF
- IBM Spectrum LSF MultiCluster capability
- Resource connector for IBM Spectrum LSF

Note: The IBM Spectrum LSF installer package, product distribution packages, product entitlement packages, and documentation packages can be found in the IBM Passport Advantage® website.

You can find additional help with downloading IBM Spectrum LSF through Passport Advantage in a YouTube video.

The following website provides detailed IBM Spectrum LSF system support information.

4.1 IBM Spectrum LSF family overview

IBM Spectrum LSF manages and accelerates research, simulation, and design workloads across distributed compute environments. It provides a comprehensive set of intelligent scheduling capabilities that help ensure that the correct resources are automatically allocated to the correct jobs for maximum application performance and efficiency.

IBM Spectrum LSF pools resource and manage application workloads across highly distributed environments. Rather than relying on a single cluster administrator, IBM Spectrum LSF provides the flexibility to delegate administrators at multiple levels through the organization.

The IBM Spectrum LSF is a workload management that is employed to coordinate shared access and optimized use of computing resources of an HPC cluster. It provides the following features, among others:

- Policy-driven work scheduling and load balancing
- Compute resources allocation
- Cluster resources administration
- Cluster monitoring
- Supports heterogeneous resources and multicluster
- Fault tolerance
- Security

Flexibility within environment growth

IBM Spectrum LSF supports organizations on their journey from small clusters to large, distributed computing environments, on-premises and in the cloud.

IBM Spectrum LSF Suite for Workgroups and *IBM Spectrum LSF Suite for HPC* deliver complete high-performance computing (HPC) management solutions for organizations running compute environments for science and engineering. Both feature the following capabilities:

- Cluster management and deployment
- Powerful yet simple workload management
- Web-enabled job management
- Support for Linux on IBM POWER8 Little Endian and x86

IBM Spectrum LSF Community Edition is a no-charge, fully-integrated solution for HPC featuring cluster provisioning and management, workload scheduling, an application-centric portal and an MPI library.

Optional add-ons extend IBM Spectrum LSF to provide a complete set of workload management capabilities-all designed to work together to address your high-performance computing needs.

Figure 4-1 shows the broad IBM Spectrum LSF family, which includes a rich set of available add-on products.

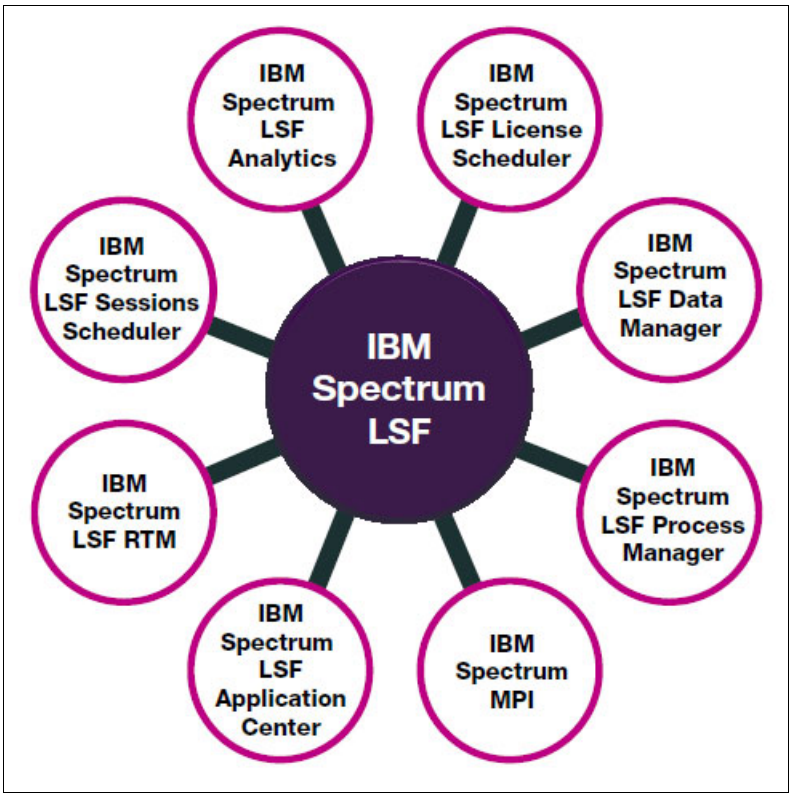


Figure 4-1 IBM Spectrum LSF add-ons

4.1.1 IBM Spectrum LSF family offerings

At the time this publication was written, we implemented the current version, 10.1.0. IBM Spectrum LSF 10.1 is available as the following offering packages: IBM Spectrum LSF Community Edition 10.1, IBM Spectrum LSF Suite for Workgroups 10.1, and IBM Spectrum LSF Suite for HPC 10.1.

The IBM Spectrum LSF offerings are shown in Table 4-1.

Table 4-1 IBM Spectrum LSF offerings

Name	Components
IBM Spectrum LSF Community Edition 10.1 It is a no-charge community edition of the IBM Spectrum LSF workload management platform.	**IBM Spectrum LSF Express Edition 10.1** is a solution for Linux customers with simple scheduling requirements and simple fairshare setup. **IBM Spectrum LSF Application Center Basic Edition 10.1** provides a flexible, easy to use interface for cluster users and administrators. IBM Spectrum LSF Application Center Basic Edition offers basic job submission and monitoring. **IBM Spectrum MPI Community Edition 10.1** is a no-charge community edition of IBM Spectrum MPI supporting the core MPI features. It has the core capabilities of IBM Spectrum MPI and provides a no-cost offering that includes higher rank counts, always available IBM customer support, fix packs, and upgrade protection. IBM Spectrum MPI is a production-quality implementation of the Message Passing Interface (MPI) that supports the broadest range of industry standard platforms, interconnects, and operating systems. **IBM Spectrum Cluster Foundation Community Edition 4.2.2** is powerful cluster management software that delivers a comprehensive set of functions to help manage hardware and software from the infrastructure level.
IBM Spectrum LSF Suite for Workgroups 10.1 IBM Spectrum LSF Suite for Workgroups can be connected to an IBM Spectrum LSF RTM server, or an IBM Spectrum LSF Analytics Server without the need to purchase Data Collector entitlements for the IBM Spectrum LSF Suite for Workgroups hosts.	**IBM Spectrum LSF Standard Edition 10.1** is a powerful workload management platform for demanding, distributed HPC environments. It provides a comprehensive set of intelligent, policy-driven scheduling features that enable you to utilize all of your compute infrastructure resources and ensure optimal application performance. **IBM Spectrum LSF Application Center Express Edition 10.1** provides a flexible, easy to use interface for cluster users and administrators. IBM Spectrum LSF Application Center Express Edition includes job submission, monitoring, reporting, application templates, simplified application configuration, and application repositories. **IBM Spectrum MPI 10.1** is a production-quality implementation of the Message Passing Interface (MPI) that supports the broadest range of industry-standard platforms, interconnects, and operating systems to help ensure that parallel applications can run on any platform. **IBM Spectrum LSF License Scheduler Basic Edition 10.1** is intended to replace an external load information manager (ELIM) to collect external load indexes for licenses managed by FlexNet or Reprise License Manager. To replace this ELIM, License Scheduler Basic Edition limits the license use of jobs of a single cluster to prevent overuse of the licenses and tracks license use of individual jobs by matching license checkouts to these jobs. **IBM Spectrum Cluster Foundation Community Edition 4.2.2** is powerful cluster management software that delivers a comprehensive set of functions to help manage hardware and software from the infrastructure level.

Name	Components
IBM Spectrum LSF Suite for HPC IBM Spectrum LSF Suite for HPC can be connected to an IBM Spectrum LSF RTM server, or an IBM Spectrum LSF Analytics Server without the need to purchase Data Collector entitlements for the IBM Spectrum LSF Suite for HPC hosts.	**IBM Spectrum LSF Standard Edition 10.1** is a powerful workload management platform for demanding, distributed HPC environments. It provides a comprehensive set of intelligent, policy-driven scheduling features that enable you to utilize all of your compute infrastructure resources and ensure optimal application performance. **IBM Spectrum LSF Application Center Standard Edition 10.1** provides a flexible, easy to use interface for cluster users and administrators. It is the full-featured offering, including job submission, monitoring, reporting, application templates, simplified application configuration, extended template and page customization, visualization support through VNC, access control lists, and integration with IBM Spectrum LSF Process Manager. **IBM Spectrum LSF Process Manager 10.1** helps individuals when complex scripts are often used to automate lengthy computing tasks as these scripts can be risky to modify, and can depend on the expertise of a few key individuals. IBM Spectrum LSF Process Manager simplifies the design and automation of complex computational processes, capturing and protecting repeatable best practices. **IBM Spectrum MPI 10.1** is a production-quality implementation of the Message Passing Interface (MPI) that supports the broadest range of industry standard platforms, interconnects, and operating systems to help ensure that parallel applications can run on any platform. **IBM Spectrum LSF License Scheduler Basic Edition 10.1** is intended to replace an external load information manager (ELIM) to collect external load indexes for licenses managed by FlexNet or Reprise License Manager. To replace this ELIM, License Scheduler Basic Edition limits the license use of jobs of a single cluster to prevent overuse of the licenses and tracks license use of individual jobs by matching license checkouts to these jobs. **IBM Spectrum Cluster Foundation Community Edition 4.2.2** is powerful cluster management software that delivers a comprehensive set of functions to help manage hardware and software from the infrastructure level.

Note: The IBM Spectrum LSF Process Manager failover feature is not supported in IBM Spectrum LSF Suite for HPC.

Note: If you need more comprehensive support, a paid support offering is available from IBM. Community assistance is available through the forums on the IBM developerWorks® website.

4.1.2 IBM Spectrum LSF optional add-ons

Optional add-ons extend IBM Spectrum LSF to provide a complete set of workload management capabilities all designed to work together to address your high performance computing needs.

Table 4-2 describes the IBM Spectrum LSF add-ons.

Table 4-2 IBM Spectrum LSF add-ons

Feature	Description
IBM Spectrum LSF Analytics Analyze business decisions	An advanced analysis and visualization tool for analyzing massive amounts of IBM Spectrum LSF workload data. It enables managers, planners and administrators to correlate job, resource and license data easily from one or more IBM Spectrum LSF clusters for data-driven decision-making. For more information, see the IBM Spectrum LSF Analytics Data Sheet.
IBM Spectrum LSF Application Center	A rich environment for building easy-to-use, application-centric web interfaces, simplifying job submission, management, and remote visualization. Use the web-based interface to remotely monitor jobs, access job-related data, and perform basic operations. For more information, see the IBM Spectrum LSF Application Center Data Sheet.
IBM Spectrum LSF Process Manager	A powerful interface for designing complex engineering computational processes, capturing repeatable best practices that can be leveraged by other users. Integrate with IBM Spectrum LSF Application Center to create a consistent web-based environment. For more information, see the IBM Spectrum LSF Process Manager Data Sheet.
IBM Spectrum LSF Data Manager	An intelligent data manager for automating data transfer within and between IBM Spectrum LSF clusters and to and from the cloud. It takes control of data transfers to help organizations improve data throughput and lower costs by minimizing wasted compute cycles and conserving disk space. For more information, see the IBM Spectrum LSF Data Manager Data Sheet.
IBM Spectrum LSF License Scheduler	A license management tool that enables policy-driven allocation and tracking of commercial software licenses. Monitor license usage in real time to help improve productivity and increase overall access to license resources. For more information, see the IBM Spectrum LSF License Scheduler Data Sheet.
IBM Spectrum LSF RTM	A flexible, real-time dashboard for monitoring global workloads and resources. Gain timely insights into the current status of your HPC environment to help improve decision-making, reduce costs, and improve service levels. For more information, see the IBM Spectrum LSF RTM Data Sheet.
IBM Spectrum LSF Session Scheduler	A high throughput, low-latency scheduling solution for IBM Spectrum LSF environments. Schedule high throughput, low-latency workloads for faster and more predictable job delivery times. For more information, see the IBM Spectrum LSF Session Scheduler Data Sheet.

4.2 IBM Spectrum LSF integration with Docker

Docker is a lightweight Linux container technology built on top of Linux Containers (LXC) and control groups (*cgroups*). Modern distributed applications are often assembled from existing components and rely on other services and applications. Due to these dependencies, deploying these applications on a notebook, in the data center and in the cloud can be challenging and require completely different processes. Docker offers a technology that streamlines building, testing, and deploying such applications, enabling an application stack to be consistently deployed anywhere.

Note: The current information about Docker solution is available on the Docker Docs website.

HPC containers

The HPC community has used container technology in production for many years for workload resource isolation, process tracking, job controlling, and even checkpoint, restart, and job migration.

For example, the workload manager (WLM) on IBM AIX has been used for workload memory and CPU resource enforcement, and IBM PowerVM along with workload partitions (WPAR), and logical partitions (LPAR) have been used for application isolation and mobility.

In a Linux environment, control groups (cgroup) provide similar mechanisms for resource enforcement, device access, work-load accounting, and process tracking. It is one of the crucial features to guarantee a reliable HPC environment for both x86 and IBM PowerLinux™ clusters and has become a mandatory feature in any modern workload manager and scheduler system.

Virtual machines (VM) are another form of workload container that are used in HPC clusters and cloud environments. VMs can be deployed in response to user requests, or dynamically created in response to workload demands (such as with IBM Spectrum Dynamic Cluster).

How is Docker different?

With an operating system control group, all applications need to share the same operating system image, so upgrading the environment for one group can impact others. With a VM, you can provide complete isolation for the application, but at the expense of a managing an additional operating system instance.

Docker provides the isolation of a VM without the resources required by a guest operating system. Each Docker container only comprises the application and its dependencies, thus the size of the package is much smaller. It runs as an isolated process in user space on the host operating system, sharing the kernel with other containers.

Docker is also fast. It provides near bare metal speeds for application runtime performance so management operations (boot, reboot, stop, and so on) can be done in seconds or even milliseconds while typical VM operations can take minutes to complete.

In summary, applications running within a Docker container benefit from the resource isolation and allocation capabilities of a VM, but is much more portable and efficient overall. Refer to Figure 4-2.

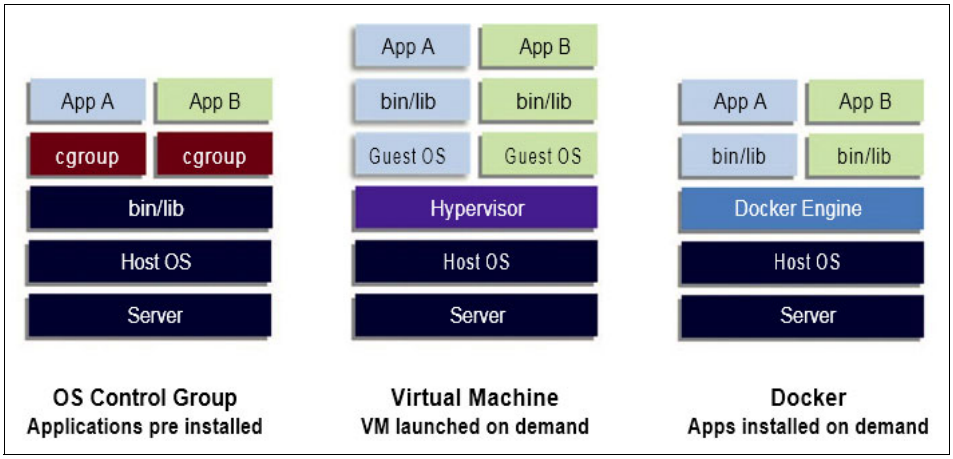


Figure 4-2 How is Docker different

Application encapsulation and cloud mobility

Applications typically depend on numerous libraries (both operating system and other applications) for correct execution. Seemingly minor changes in library versions can result in the application failing, or even worse, subtly different results. This can make moving applications from one system to another-or out on to the cloud-problematic. Docker can make it easy to package up and move an application from one system to another. Users can run the applications they need, where they need them, and at the same time administrators can stop worrying about library clashes or helping users get their application working in a specific environment.

Application lifecycle management

In genomics, for example, applications have a short development lifecycle, and an even shorter shelf life. The environment is constantly evolving, with one tool being rapidly replaced by a new variant, or something else entirely. This presents a major challenge to administrators who are trying to have all these different versions coexist, when many depend on different versions of the same library, or incompatible versions.

With Docker, application binary files, dependencies, and configurations can all be fully encapsulated within a container. This is independent from the host operating system and other applications on the host. Users can run different versions of the software on the same host without worrying about conflicts. Furthermore, new software packages and applications can easily be pushed out to compute nodes on demand, which potentially eases the application management burden for administrators.

4.2.1 IBM Spectrum LSF and Docker integration

IBM Spectrum LSF features an integration with Docker, which allows users to submit and manage the lifecycle of Docker container-based applications. Docker offers many attractive benefits in an HPC environment. To test these functions, IBM Spectrum LSF and Docker have been integrated outside the core of IBM Spectrum LSF to use the rich IBM Spectrum LSF plug-in framework:

- The integration takes the Docker image as one of the submission options, schedules the job on a Docker enabled host, and starts a Docker container on the selected host.

- During job launch, the integration sets proper resource constraints based on allocation (CPU, memory, and affinity), passes all job submission environment variables and mounts the job's current working directory for the container.
- This ensures that the application running within the Docker container gets the same runtime environment that it normally uses outside of the container.
- During job execution, the integration supports full IBM Spectrum LSF job lifecycle management functions, including monitoring and reporting the container resource usage, and conducting container control actions to respond to job suspension, resumption, and termination requests.

Docker offers both HPC users and administrators a solution to application conflicts, installation, and mobility. The simple integration that is presented enables a user to transparently run the application (BWA) under IBM Spectrum LSF in the same manner as without Docker. There is no need for the administrator to install the software for the user because it is all done on demand and without conflicts, significantly simplifying administration.

4.2.2 IBM Spectrum LSF 10.1 solution Docker job support

The solution allows IBM Spectrum LSF to run job in Docker containers on demand. IBM Spectrum LSF manages the entire lifecycle of jobs running in the container as common jobs. This solution enables IBM Spectrum LSF to run and manage jobs in Docker containers.

Prerequisites

Docker Engine version 1.12, or later must be installed on IBM Spectrum LSF server hosts. The Docker daemon must be started on hosts and can successfully start containers.

- In the `lsf.conf` file, configure the following parameters:

```
LSF_PROCESS_TRACKING=Y
LSF_LINUX_CGROUP_ACCT=Y
LSB_RESOURCE_ENFORCE="cpu memory"
```

- In the `lsf.shared` file, configure the Boolean resource `docker`:

```
Begin Resource
RESOURCENAME TYPE    INTERVAL INCREASING DESCRIPTION
docker       Boolean ()       ()         (docker host)
```

- In the `lsf.cluster` file, attach the Boolean resource `docker` to IBM Spectrum LSF server hosts on which Docker Engine is running. This enables IBM Spectrum LSF to automatically dispatch Docker jobs to the `docker` hosts.

```
Begin Host
HOSTNAME model type server r1m mem swp RESOURCES
host1    !     !    1      3.5 ()  ()  (docker)
```

Configuration

This solution introduces a new parameter `CONTAINER` in the `lsb.applications` file for configuring the Docker job application profile. You need to enter it to run the integration properly, using the following syntax:

```
CONTAINER=docker[image(image-name) options(docker-run-options) starter(user-name)]
```

In this configuration, the following key words are used:

image	Required. This keyword configures the Docker image name that is used in running jobs.
options	Optional. This keyword configures the Docker job run options, which are passed to the job container by `docker run` in IBM Spectrum LSF.
starter	Optional. This keyword specifies the name of the user that starts the **docker run** to launch containers for jobs. The default user is the IBM Spectrum LSF primary administrator.

Before specifying the Docker job run options, make sure that these options work in the Docker run command line:

1. The `--cgroup-parent, --user (-u), and --name` options are reserved for IBM Spectrum LSF internal use. Do not use these options in the options keyword configuration.

   The `-w and --ulimit` options are automatically set for IBM Spectrum LSF automatically. Do not use these options in the options keyword configuration because the specifications here override the IBM Spectrum LSF settings.

   The `-v` option is automatically used to mount the working directories that LSF needs: current working directory, job spool directory, destination file for the **bsub -f** command, tmp directory, top-level IBM Spectrum LSF, and checkpoint directory on demand.

2. `--rm` is proposed to be configured in options keyword configuration to automatically remove containers after job is done.

3. The starter account must be root or the user configured in the `docker` user group. To add a user to the `docker` user group, run the **sudo usermod -aG docker starter_username** command.

Examples

- Sequence job

  ```
  CONTAINER=docker[image(image-name) options(--rm)]
  ```

- Parallel job

  To make **blaunch** work, the network and IPC must work across containers, the execution user ID and user name mapping file must be mounted into the container for **blaunch** authentication.

  ```
  CONTAINER=docker[image(image-name) options(--rm --network=host --ipc=host -v
  /path/to/my/passwd:/etc/passwd)]
  ```

The passwd file is in the following format:

```
user1:x:10001:10001:::
user2:x:10002:10002:::
```

Installation and configuration steps

Info: The solution allows IBM Spectrum LSF to run a job in Docker containers on demand. The current installation package is available on the IBM FixCentral page.

You can also find it by typing `RFE#86007` in the search area of the IBM Support pages.

The following are the installation and configuration steps:

1. Check that the following step is done before installation:

   ```
   LSF_TOP=Full path to the top-level installation directory of LSF.
   ```

2. Log on to the IBM Spectrum LSF master host as `root`.
3. Set your environment:
   - For csh or tcsh: `% source LSF_TOP/conf/cshrc.lsf`
   - For sh, ksh, or bash: `$ . LSF_TOP/conf/profile.lsf`
4. Perform installation steps:
   a. Go to the patch install directory: `cd $LSF_ENVDIR/../10.1/install/`
   b. Copy the patch file to the install directory: `$LSF_ENVDIR/../10.1/install/`
   c. Run patchinstall: `./patchinstall <patch>`
5. After installation, complete the following actions:
   a. Log on to the IBM Spectrum LSF master host as `root`
   b. Run `lsfrestart`
6. To roll back a patch, complete the following steps:
   a. Log on to the IBM Spectrum LSF master host as `root`
   b. Run `./patchinstall -r <patch>`
   c. Run `lsfrestart`

4.3 IBM Spectrum LSF Data Manager

As global organizations consolidate IT resources in larger, centralized data centers and deliver services through public and private clouds, application performance suffers. With business users, customers and partners all accessing the same systems and data, critical workloads can slow down as data moves from storage resources to compute resources and back again.

Whether you are running these data-intensive applications in a single cluster or you want to share data and compute resources across geographically separated clusters, IBM Spectrum LSF Data Manager provides the following key features:

- Manage data transfers independent of cluster workloads to improve throughput and optimize use of compute resources.
- Leverage an intelligent, managed cache to eliminate duplication of data transfers and lower storage costs.
- Gain full visibility and control of data transfer jobs using IBM Spectrum LSF scheduling policies.
- Simplify administration with the ability to configure dedicated I/O nodes for inbound/outbound data transfers.
- Provide user and administrator access control over cached data.
- Input data can be staged from an external source storage repository to a cache that is accessible to the cluster execution hosts.
- Output data is staged asynchronously (dependency-free) from the cache after job completion.
- Data transfers run separately from the job allocation, which means more jobs can request data without using resources waiting for large data transfers.
- Remote execution cluster selection and cluster affinity are based on data availability in an IBM Spectrum MultiCluster environment. IBM Spectrum LSF Data Manager transfers the required data to the cluster that the job was forwarded to.

4.3.1 Concepts and terminology

The following concepts and terminology are specific for IBM Spectrum LSF Data Manager:

- IBM Spectrum LSF Data Manager

  The LSF data manager runs on dedicated IBM Spectrum LSF server hosts. The IBM Spectrum LSF Data Manager hosts are configured to run the IBM Spectrum LSF Data Manager daemon (dmd). The IBM Spectrum LSF Data Manager daemon communicates with the clusters it serves, and manages the transfer of data in the staging area.

- IBM Spectrum LSF Data Manager administrator

  The administrator of the IBM Spectrum LSF Data Manager must be an IBM Spectrum LSF administrator for all clusters that are connected to the data manager. IBM Spectrum LSF Data Manager administrators make sure that dmd is operating smoothly and reconfigure IBM Spectrum LSF Data Manager as needed.

  IBM Spectrum LSF Data Manager administrators perform administrative functions on the IBM Spectrum LSF Data Manager:

  - Manage the IBM Spectrum LSF Data Manager data transfer queue in the `lsb.queues` file
  - Run **`bdata admin reconfig`** to reconfigure LSF data manager
  - Run **`bdata admin shutdown`** to shut down LSF data manager
  - Run **`bdata tags`** to list or clean intermediate files that are associated with a tag for users

- lsf.datamanger file

  The `lsf.datamanager file` controls the operation of IBM Spectrum LSF Data Manager features. There is one IBM Spectrum LSF data management configuration file for each cluster, called `lsf.datamanager.cluster_name`. The `cluster_name` suffix is the name of the cluster that is defined in the `Cluster` section of `lsf.shared`. The file is read by the IBM Spectrum LSF data management daemon `dmd` during startup and reconfiguration.

- Data transfer node

  A *data transfer node*, also referred to as an *I/O node*, is an IBM Spectrum LSF server host in the cluster that is mounted with direct read/write access to the cluster staging area. This host can access the source of staged-in data and the destination of staged-out data.

- Data transfer job

  IBM Spectrum LSF Data Manager submits transfer jobs to copy required data files for stage in or stage out operations. Transfer jobs run on data transfer nodes as the execution user of the job that triggered the transfer.

  Transfer jobs have the following function:

  - To pre-stage files that are requested in the **`bsub –data`** option from their source location into the staging area cache.
  - Stage out files that are requested by the **`bstage out`** command from the staging area cache to their remote destination.

- Data transfer queue

  IBM Spectrum LSF Data Manager submits transfer jobs to a transfer queue, which is configured to accept transfer jobs only.

- Data transfer tool command

  Transfer jobs run the **`transfer tool`** command that is specified in `FILE_TRANSFER_CMD` in `lsf.datamanager`.

- Data staging

  A staging area, also known as a cache, is a managed file system on an IBM Spectrum LSF server host or mounted directory that is accessible to the cluster compute nodes. The staging area stores files that jobs request for staging in or out. There must be one IBM Spectrum LSF Data Manager instance for each staging area.

  Data staging is the transfer of data to the location where it is used. Input data for a job is staged in two steps:

  – By copying input data files from the data source repository to the data staging area cache through a data transfer job

  – By copying data that is required by an application from the staging area to the job execution host

  Output data that is produced by a job is staged out by copying files back to a location that the job submitter can use it. Stage-out operations asynchronously copy job output from the cache area to the data destination that you specify.

- Data specification file

  A data specification file is a text file that is used for specifying many data requirement files for one job.

  Each line in a data specification file specifies the name of the path to a source file to be transferred to the staging area before a job is submitted and scheduled. The path can point to a file or a directory.

  The following example contains lines for three files. Each line specifies a `host_name:file_path` pair:

```
#@dataspec
datahost:/proj/userA/input1.dat
datahost:/proj/userA/input2.dat
datahost:/proj/userA/input3.dat
```

- Data tags

  A data tag can be created for a job with a data staging requirement with the **`bstage out`** command. A tag allows users to transfer files from the job's current working directory to the staging area, associate those files with a chosen name, and to have the IBM Spectrum LSF Data Manager report the existence of that tag if it is queried later.

- Data queries

  File-based cache query with bdata cache displays the job IDs of jobs that request the file under `REF_JOB. The REF_JOB` column is not displayed for job-based query in `bdata` cache.

4.3.2 How IBM Spectrum LSF Data Manager works

This section describes how IBM Spectrum LSF Data Manager works.

Schedule data transfers to reduce costs

IBM Spectrum LSF Data Manager takes control of data transfers to help organizations improve data throughput and lower costs by minimizing wasted compute cycles and conserving disk space. IBM Spectrum LSF Data Manager automates the transfer of data used by application work-loads running on IBM Spectrum LSF clusters and the cloud, bringing frequently used data closer to compute resources by storing it in an intelligent, managed cache that can be shared among users and workloads.

Data is staged in and out independent of workloads, freeing compute resources to work on other jobs while data is transferred back and forth among compute resources and storage. IBM Spectrum LSF Data Manager also leverages the data movement infrastructure already in place, allowing it to be easily integrated into existing cluster infrastructure.

Smart, managed cache

IBM Spectrum LSF Data Manager leverages intelligent, managed cache to reuse data common to multiple workloads and accelerate time to results. Transferred files are automatically cached on the execution cluster with an optional time to live. Cached copies can be leveraged for all workloads that need access to the data, and can be shared between multiple users where appropriate. Workloads running on execution clusters can also write intermediate data to the local cache (such as restart data) for use by other jobs.

Every IBM Spectrum LSF cluster that shares the same staging area also communicates to the same IBM Spectrum LSF Data Manager instance. The clusters query the data manager for the availability of data files. If the files are not in the cache, the IBM Spectrum LSF Data Manager stages them and notifies the cluster when a requested data for the job is ready. After the files are staged, the clusters can retrieve them from the staging area by consulting the data file information stored in the staging area by the IBM Spectrum LSF Data Manager.

4.3.3 Single cluster implementation

This section describes the architecture of a single cluster implementation of IBM Spectrum LSF Data Manager, and how each component works together to accomplish the task of staging data and submitting jobs with data requirements. Figure 4-3 shows the architecture for a single cluster implementation of IBM Spectrum LSF Data Manager.

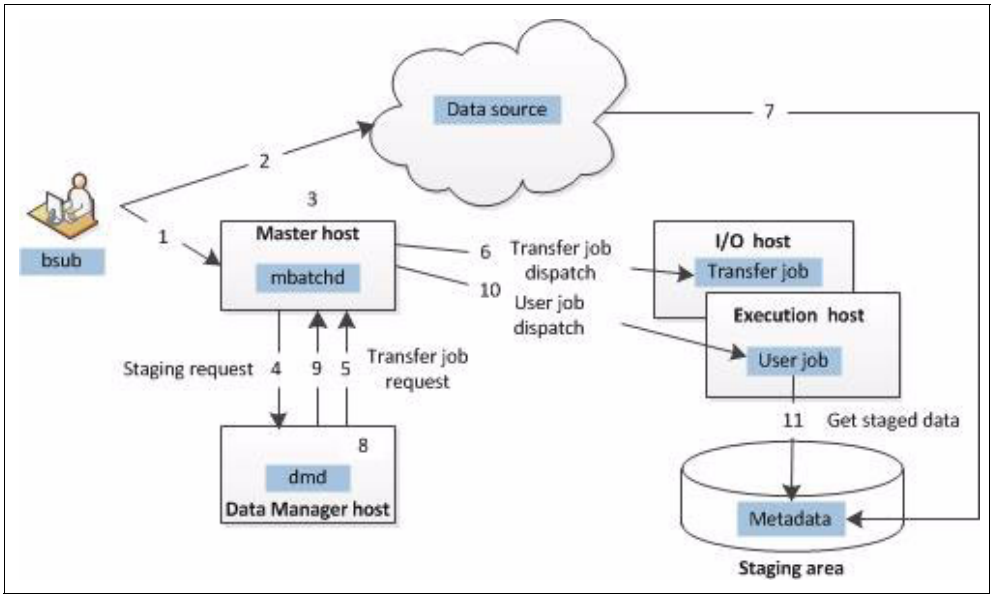


Figure 4-3 Single cluster architecture for IBM Spectrum LSF Data Manager

How IBM Spectrum LSF Data Manager works in single cluster

A typical IBM Spectrum LSF Data Manager single cluster process includes these steps:

1. A user submits a job using **bsub** with a requirement for a data file, including the name of the data source host and the full path to the required data.
2. The **bsub** command reads information about the data source host, the file path, the size of the file and last modified time (if available) and sends that information to the IBM Spectrum LSF master host along with the job submission data.
3. The job request is received by `mbatchd` on the master host, which detects that the job has an input staging requirement. The main user job is put in a hold state (`shows as PEND`).
4. The `mbatchd` process sends the data requirement information to the IBM Spectrum LSF Data Manager, with a request that the data file gets copied to the staging area.
5. If the data file was not already staged, IBM Spectrum LSF Data Manager requests a transfer job for the required files. A transfer job is submitted to `mbatchd` for each new file record created for the job.
6. The `mbatchd` process schedules and dispatches the transfer job, and notifies LSF data manager of the success or failure of the transfer job.
7. The transfer job copies file data from the source repository to the staging area. If the data file was already staged and in `TRANSFERRED` status, there is no need for a transfer job.
8. If the transfer job was successful, IBM Spectrum LSF Data Manager sets the status of the file to `TRANSFERRED`. If the transfer job fails, IBM Spectrum LSF kills jobs that require the file that the transfer job is transferring.
9. The IBM Spectrum LSF Data Manager notifies `mbatchd` that the data staging step for the job is complete.
10. The `mbatchd` process schedules the user job and dispatches it to the execution node.
11. The running user job finds where the required data files are staged, and job execution starts. The IBM Spectrum LSF Data Manager daemon (`dmd`) determines the location of the staged file in the cache. The **bstage in** command in the user job determines the location of the file in the execution environment.

4.3.4 MultiCluster implementation

This section describes the architecture of a MultiCluster implementation of IBM Spectrum LSF Data Manager with a queue configuration that is not a remote-only queue, and how each component works together to accomplish the task of staging data and submitting jobs with data requirements.

Figure 4-4 shows the architecture of a multi-cluster implementation of IBM Spectrum LSF Data Manager.

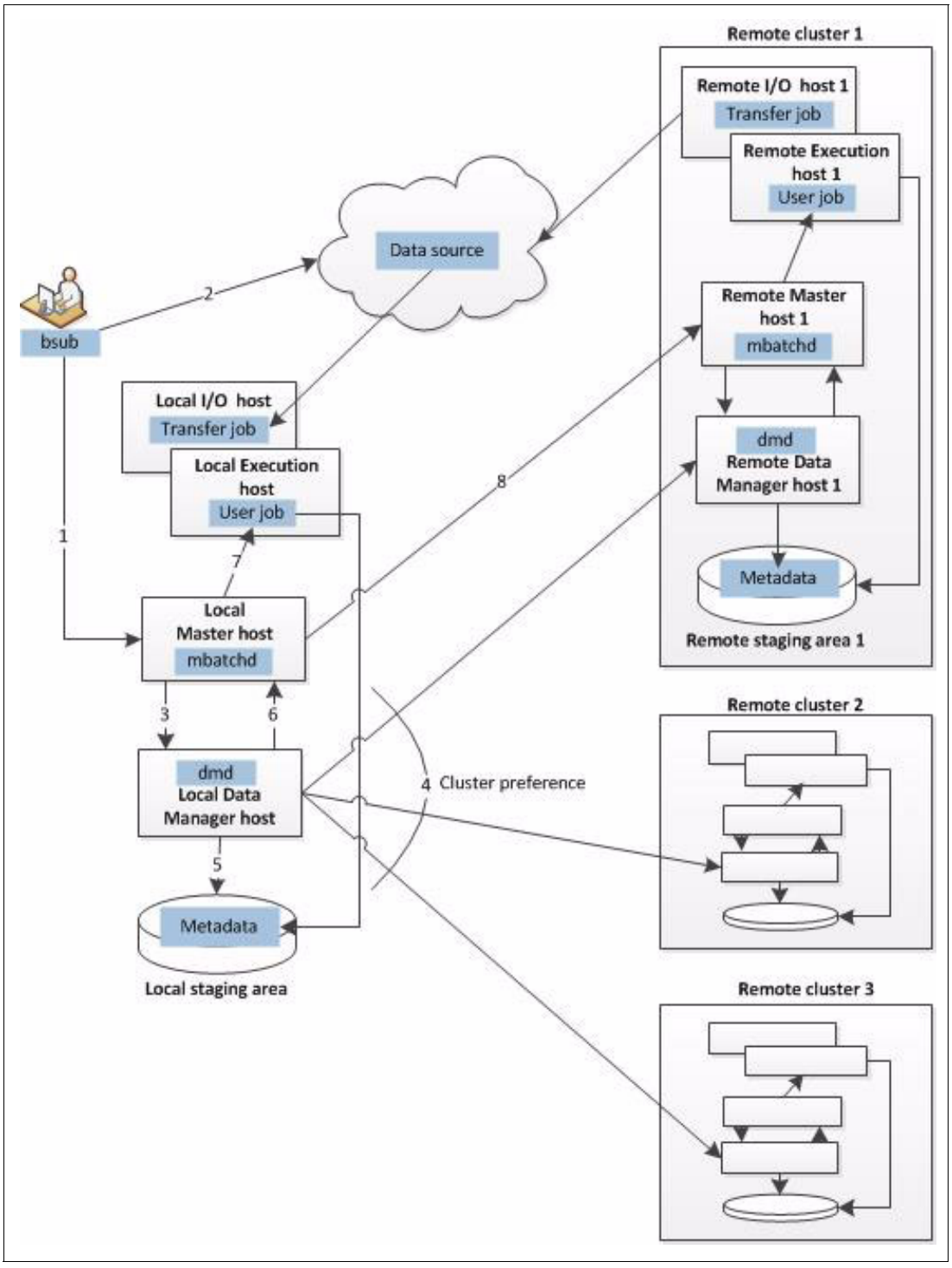


Figure 4-4 Multicluster infrastructure for IBM Spectrum LSF Data Manager

How IBM Spectrum LSF Data Manager works in multicluster

A typical IBM Spectrum LSF Data Manager multi-cluster process includes these steps:

1. A user submits a job to the local cluster using **bsub** with a requirement for a data file.
2. The **bsub** command reads information about the data source host, the file path, the size of the file and last modified time (if available) and sends that information to the IBM Spectrum LSF master host along with the job submission data.
3. The local `mbatchd` sends the data requirement information to the local IBM Spectrum LSF Data Manager, with a request that the data file get copied to the local staging area. The data requirement information includes the candidate clusters that the job is eligible to be forwarded to. If the requested file does not exist, and the local cluster is a candidate for the job, a transfer job is submitted and the required files are staged to the local staging area.
4. If the user does not specify a cluster preference in the job, the data manager queries all remote data managers configured in the candidate cluster list for the requested files. The IBM Spectrum LSF Data Manager generates a cluster preference for the job.
5. The data manager creates a file containing data file information for the job in its staging area current working directory.
6. After data is staged to the local staging area, the data manager informs IBM Spectrum LSF that the data for the job is ready in the local cluster so the job can be scheduled. IBM Spectrum LSF appends the cluster preference to the job and schedules it normally. If local staging was not needed, this step happens immediately.
7. If the job is scheduled locally, **bstage in** gets the required files and the job runs using the locally staged data.
8. If the job is forwarded to a remote cluster, it is accepted at the remote side and data is staged for it on the remote staging area though it is a locally submitted job.

4.4 IBM Spectrum Symphony MapReduce Accelerator for IBM Spectrum LSF

IBM Spectrum Symphony MapReduce Accelerator for IBM Spectrum LSF enables you to submit and work with Apache Hadoop MapReduce jobs in IBM Spectrum LSF. The Hadoop MapReduce Processing framework is a distributed runtime engine for enterprise-class Hadoop MapReduce applications and shared services deployments.

About IBM Spectrum LSF with Apache Hadoop

Apache Hadoop is a framework for large-scale distributed data storage and processing on computer clusters that uses the Hadoop Distributed File System (HDFS) for the data storage and MapReduce programming model for the data processing. Because MapReduce workloads might only represent a small fraction of overall workload, but typically require their own stand-alone environment, MapReduce is difficult to support within traditional HPC clusters.

However, HPC clusters typically use parallel file systems that are sufficient for initial MapReduce workloads, so you can run MapReduce workloads as regular parallel jobs running in an HPC cluster environment. Use the IBM Spectrum LSF integration with Apache Hadoop to submit Hadoop MapReduce workloads as regular IBM Spectrum LSF parallel jobs.

Using IBM Spectrum LSF with Apache Hadoop

To run your Hadoop application through IBM Spectrum LSF, submit it as an IBM Spectrum LSF job. After the IBM Spectrum LSF job starts to run, the Hadoop connector script (lsfhadoop.sh) automatically provisions an open source Hadoop cluster within IBM Spectrum LSF allocated resources, then submits actual MapReduce workloads into this Hadoop cluster.

Considering each IBM Spectrum LSF Hadoop job has its own resource (cluster), the integration provides a multi-tenancy environment to enable multiple users to share the common pool of HPC cluster resources. IBM Spectrum LSF is able to collect resource usage of MapReduce workloads as normal IBM Spectrum LSF parallel jobs and has full control of the job lifecycle. After the job is complete, IBM Spectrum LSF shuts down the Hadoop cluster.

By default, the Apache Hadoop integration configures the Hadoop cluster with direct access to shared file systems and does not require HDFS. This allows you to use existing file systems in your HPC cluster without having to immediately invest in a new file system. Through the existing shared file system, data can be stored in common share locations, which avoids the typical data stage-in and stage-out steps with HDFS.

The Hadoop connector script for IBM Spectrum LSF is an integration that allows users to submit Hadoop MapReduce workloads as a regular LSF parallel job and run in an HPC cluster environment.

The IBM Spectrum LSF integration with Apache Hadoop provides a connector script that allows users to submit Hadoop applications as regular IBM Spectrum LSF jobs.

4.4.1 Configure the Apache Hadoop integration

Deploy Apache Hadoop and prepare IBM Spectrum LSF for the integration package. The Apache Hadoop integration supports the following platforms:

- x86_64
- ppc64
- ppc64le

Procedure

1. Download and deploy Apache Hadoop.

   This integration supports Hadoop, versions 1 and 2, and is tested with Open Source Hadoop versions 1.2.1 and 2.7.2; however, this integration also works with other versions and other Hadoop distributions.

   Tip: The current version is available from the official Apache Hadoop site. You do not need to configure Apache Hadoop after installation. The Hadoop connector scripts automatically configure Apache Hadoop for you.

2. Set the `$HADOOP_HOME` environment variable as the file path to the Hadoop installation directory.

   If you do not set the `$HADOOP_HOME` environment before using the integration, you must run the `lsfhadoop.sh` connector script with the --hadoop-dir option to specify the file path to the Hadoop installation directory.

3. Set the $JAVA_HOME environment variable as the file path to the Java runtime installation directory.

   If you do not set the $JAVA_HOME environment before using the integration, you must run the lsfhadoop.sh connector script with the --java-home option to specify the file path to the Hadoop installation directory.

   Running the --java-home option allows you to overwrite the value of the $JAVA_HOME environment variable on the command line. For more details, refer to next section: Run a Hadoop application on IBM Spectrum LSF.

4. You must ensure that the $HADOOP_HOME and $JAVA_HOME directories are accessible to each IBM Spectrum LSF server host.

5. You can overwrite the $HADOOP_HOME and $JAVA_HOME environment variables at job submission time by using the --hadoop-dir and --java-home options with lsfhadoop.sh

4.4.2 Run a Hadoop application on IBM Spectrum LSF

You must have a Hadoop application compiled and ready to run as a JAR file.

Use the **bsub** command to submit the Hadoop application to IBM Spectrum LSF.

Procedure

1. If you intend to submit multiple MapReduce workloads within a single Hadoop job to be submitted as an IBM Spectrum LSF job, create a script with multiple Hadoop commands.

   Prepare a script with multiple Hadoop commands. For example, create a file named mrjob.sh with the following content:

   ```
   #!/bin/bash
   hadoop jar sort.jar /gpfs/mapreduce/data/sort/input /gpfs/mapreduce/data/sort/
   hadoop jar wordcount.jar /gpfs/mapreduce/data/input /gpfs/mapreduce/data/output
   ```

   Change the file permissions on the script to make it executable.

   This script is run with the lsfhadoop.sh connector script when submitted as an IBM Spectrum LSF job.

2. Use the **bsub** command to submit the Hadoop application to IBM Spectrum LSF through the lsfhadoop.sh connector script.

   To make Hadoop jobs run efficiently, use the -x option to make an exclusive job allocation.

   The lsfhadoop.sh connector script creates a temporary directory under the job's working directory and places data files in this temporary directory. To allow a job-based Hadoop cluster to access these files, you must set the job's working directory to a location in a shared file system. Use the -cwd option to specify the job's working directory.

3. To specify a Hadoop command to run the Hadoop workload, use hadoop jar as a subcommand of the lsfhadoop.sh connector script (see Table 4-3 on page 48):

   ```
   bsub bsub_options lsfhadoop.sh [-h] [--hadoop-dir file_path] [--java-home
   file_path] [--config-dir file_path] [--user-hdfs] [-debug] hadoop jar
   jarfile_name jarfile_parameters
   ```

   To specify a script that runs multiple Hadoop commands within a single IBM Spectrum LSF job, run the script as an argument to the lsfhadoop.sh connector script:

   ```
   bsub bsub_options lsfhadoop.sh [-h] [--hadoop-dir file_path] [--java-home
   file_path] [--config-dir file_path] [--user-hdfs] [-debug] script_name
   ```

Table 4-3 The lsfhadoop.sh script options

Script Element	Description
`--hadoop-dir`	Specifies the Hadoop installation directory. The default is the value of the `$HADOOP_HOME` environment variable
`--java-home`	Specifies the location of the Java runtime environment. The default is the value of the $JAVA_HOME environment variable
`--config-dir`	Specifies the Hadoop configuration file directory, if you want to define your own customized Hadoop configuration. The default is the Hadoop installation directory
`--use-hdfs`	Specifies that the MapReduce job runs on an HDFS file system. By default, lsfhadoop.sh sets up a job-based Hadoop cluster to directly access a shared file system
`--debug`	Enables the job to retain the intermediate work files for troubleshooting purposes
`--A`	Specifies the end of the lsfhadoop.sh options and disables further option processing for lsfhadoop.sh. This allows you to specify other generic Hadoop options after lsfhadoop.sh, such as -D parameter=value

Example:

To run a `wordcount` application on Hadoop with three hosts, run the following command:

```
bsub -x -R"span[ptile=1]" -n 3 -cwd /gpfs/mapreduce/work lsfhadoop.sh hadoop
jar wordcount.jar /gpfs/mapreduce/data/input /gpfs/mapreduce/data/output
```

To run the `mrjob.sh` script (which contains multiple **`hadoop jar`** commands) on Hadoop with three hosts, run the following command:

```
bsub -x -R"span[ptile=1]" -n 3 -cwd /gpfs/mapreduce/work lsfhadoop.sh mrjob.sh
```

4.5 IBM Spectrum LSF MultiCluster capability

Within an organization, sites might have separate, independently managed IBM Spectrum LSF clusters. Having multiple IBM Spectrum LSF clusters can solve problems that are related to the following areas:

- Ease of administration
- Different geographic locations
- Scalability

When you have more than one cluster, it is preferable to allow the clusters to cooperate to reap the following benefits of global load sharing:

- Access to a diverse collection of computing resources
- Enterprise grid computing becomes a reality
- Get better performance and computing capabilities
- Use idle machines to process jobs
- Use multiple machines to process a single parallel job
- Increase user productivity
- Add resources anywhere and make them available to the entire organization
- Plan computing resources globally based on total computing demand
- Increase computing power in an economical way

MultiCluster enables a large organization to form multiple cooperating clusters of computers so that load sharing happens not only within clusters, but also among them. MultiCluster enables the following features:

- Load sharing across many hosts
- Co-scheduling among clusters: The job forwarding scheduler considers remote cluster and queue availability and loads before forwarding jobs
- Resource ownership and autonomy are enforced
- Non-shared user accounts and file systems are supported
- Communication limitations among the clusters are considered in job scheduling

There are two different ways to share resources between clusters by using MultiCluster. These models can be combined, for example, Cluster1 forwards jobs to Cluster2 by using the job forwarding model, and Cluster2 borrows resources from Cluster3 by using the resource leasing model.

Choosing a model

Consider your own goals and priorities when choosing the best resource-sharing model for your site:

- The job forwarding model can make resources available to jobs from multiple clusters. This flexibility allows maximum throughput when each cluster's resource usage fluctuates. The resource leasing model can allow one cluster exclusive control of a dedicated resource, which can be more efficient when there is a steady amount of work.
- The resource leasing model is the most transparent to users and supports the same scheduling features as a single cluster.
- The job forwarding model has a single point of administration, and the lease model shares administration between provider and consumer clusters.

Job forwarding model

In this model, the cluster that is starving for resources sends jobs over to the cluster that has resources to spare. To work together, the two clusters must set up compatible send-jobs and receive-jobs queues.

With this model, scheduling of MultiCluster jobs is a process with two scheduling phases:

1. The submission cluster selects a suitable remote receive-jobs queue, and forwards the job to it.
2. The execution cluster selects a suitable host and dispatches the job to it.

This method automatically favors local hosts; a MultiCluster send-jobs queue always attempts to find a suitable local host before considering a receive-jobs queue in another cluster.

Resource leasing model

In this model, the cluster that is starving for resources takes resources away from the cluster that has resources to spare. To work together, the provider cluster must *export* resources to the consumer, and the consumer cluster must configure a queue to use those resources.

In this model, each cluster schedules work on a single system image, which includes both borrowed hosts and local hosts.

4.6 Resource connector for IBM Spectrum LSF

The resource connector for IBM Spectrum LSF feature (also referred to as host factory) enables IBM Spectrum LSF clusters to borrow resources from supported resource providers (for example, IBM Spectrum Conductor, OpenStack, or Amazon Web Services) based on workload.

IBM Spectrum LSF clusters can borrow hosts from an EGO cluster, and start instances from OpenStack or Amazon Web Services (AWS) to satisfy pending workload. The borrowed resources join the IBM Spectrum LSF cluster as hosts. If OpenStack or AWS instances become idle, IBM Spectrum LSF resource connector stops them.

The resource connector generates requests for extra hosts from the resource provider and dispatches jobs to dynamic hosts that join the IBM Spectrum LSF cluster. When the resource provider reclaims the hosts, the resource connector requeues the jobs that are running on the IBM Spectrum LSF hosts, shuts down IBM Spectrum LSF daemons, and releases the hosts to the provider.

How IBM Spectrum LSF borrows hosts from the resource provider

The following workflow summarizes how a user's job uses resources borrowed from a resource provider:

1. A user submits a job to IBM Spectrum LSF as normal, but the cluster does not have enough resources to service it.
2. IBM Spectrum LSF calculates how many of each template type it requires to run the job, and sends this demand to the resource connector `ebrokerd`.
3. IBM Spectrum LSF resource connector makes an allocation request to the resource provider, based on the demand. For example, if the resource provider is EGO, resource connector makes an allocation request based on the demand to EGO as the LSF_Consumer.
4. If enough resources are available in the `rg_shared` resource group, the allocation request succeeds.
5. Resource connector detects that the request succeeds, starts IBM Spectrum LSF daemons on the allocated hosts, and notifies LSF that the hosts are ready to use.
6. When the host joins the cluster, the job is dispatched to the host.

Example of borrowing hosts from EGO

This section describes the IBM Spectrum LFS workflow to borrow hosts from a resource provider.

1. In the following example, the resource provider is EGO:

```
bhosts -a
HOST_NAME STATUS JL/U MAX NJOBS RUN SSUSP USUSP RSV
lsfmaster ok     -      1     1   0     0     0   0
```

2. EGO has a host `ego01` with `ncpus=1`
3. Resource connector is configured to connect to the EGO cluster.
4. A template is created that provides a numeric attribute `ncpus` with range [1:1].
5. The **bsub** command submits a job that requires a single slot.

6. Eventually host ego01 joins the cluster, and the new job runs:

```
bhosts -a
HOST_NAME STATUS JL/U MAX NJOBS RUN SSUSP USUSP RSV
lsfmaster ok        -   1     1   1     0     0   0
ego01     ok        -   1     1   1     0     0   0
```

How IBM Spectrum LSF returns hosts to EGO

The workflow for returning hosts proceeds as follows. In the examples, EGO is the resource provider.

1. An EGO application requests an allocation from a hosted resource group. A reclaim request is triggered on the host with a grace period.
2. IBM Spectrum LSF resource connector detects the reclaim request and notifies IBM Spectrum LSF of the time that the host needs to be returned.
3. IBM Spectrum LSF closes the reclaimed host and send requeue signal to the jobs on the host at a configurable time before the requested return time.
4. After the host is drained of jobs, IBM Spectrum LSF notifies resource connector.
5. Resource connector stops the IBM Spectrum LSF daemons on the host and deallocates the host from EGO.
6. EGO detects that the host is now available, and passes the host to the requesting application.

The following example shows how IBM Spectrum LSF returns hosts that are reclaimed by EGO:

1. An EGO application makes an allocation request for a slot. The EGO application is entitled to one slot on host ego01.
2. The **bhosts -w** command shoes that host ego01 goes to closed is still _RC status. The job is still running:

```
bhosts -w
HOST_NAME    STATUS   JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
lsfmaster        ok      -      1      1      1      0      0      0
ego01     closed_RC      -      1      1      1      0      0      0
```

3. After a few seconds the job is requeued.

```
bhosts -w
HOST_NAME    STATUS JL/U    MAX  NJOBS    RUN  SSUSP   USUSP    RSV
lsfmaster        ok    -      1      1      1      0       0      0
ego01     closed_RC    -      1      0      0      0       0      0
```

4. Eventually, host ego01 goes to unavail status and is no longer shown in the **bhosts** command.

```
bhosts -a
HOST_NAME   STATUS   JL/U    MAX  NJOBS   RUN  SSUSP  USUSP    RSV
lsfmaster       ok      -      1      1     1      0      0      0
ego01       unavail     -      1      0     0      0      0      0

bhosts
HOST_NAME  STATUS JL/U   MAX  NJOBS    RUN  SSUSP  USUSP    RSV
lsfmaster      ok    -     1      1      1      0      0      0
```

How IBM Spectrum LSF returns hosts to OpenStack

OpenStack does not actively reclaim a host like EGO does. An allocated VM is passively returned when one of the following conditions occurs:

- It is idle for the amount of time that is configured by the `LSB_RC_EXTERNAL_HOST_IDLE_TIME` parameter in the `lsf.conf` file.
- A time-to-live period expires (configured by the `LSB_RC_EXTERNAL_HOST_MAX_TTL` parameter in `lsf.conf` file).
- Host factory notifies LSF that a host is ready to use, but the host does not join the cluster for 10 minutes. This value is hardcoded.

If the host was previously in the cluster, IBM Spectrum LSF closes it and waits for any running jobs on the host to complete. After the host is drained of jobs, IBM Spectrum LSF notifies the resource connector. The resource connector stops the OpenStack VM, which deallocates the host.

Required OpenStack services

The following OpenStack services are required for IBM Spectrum LSF resource connector with OpenStack as the resource provider:

- Identity service (Keystone)
- Image service (Glance)
- Compute service (Nova)
- Networking service (Neutron)

Considerations

Advanced reservations and resource guarantees on borrowed hosts can be configured and created, but they are not necessarily honored. Hosts can be returned to their resource provider at any time by idle time or time-to-live policy, or EGO reclaim. The hosts might be closed or unavailable when the advanced reservation starts.

It also is possible for resource connector to over-demand for its workload if an OpenStack VM joins the cluster but is not immediately usable by scheduler.

5

IBM Spectrum Symphony

This chapter describes IBM Spectrum Symphony 7.1.2.

IBM Spectrum Symphony software helps you control the massive compute power available in your current and future technical computing systems to address challenging and complex problems. This software can help you achieve breakthrough results in business and research by addressing challenges in parallel application development and deployment, and technical computing infrastructure management.

Although a conventional batch scheduler can schedule jobs in seconds or minutes, IBM Spectrum Symphony can schedule tasks in milliseconds. This key difference means that the solution can deliver faster, better quality results-even although using a smaller amount of infrastructure-to support online or near real-time requirements.

This chapter covers the following topics:

- IBM Spectrum Symphony overview
- IBM Spectrum Symphony editions
- IBM Spectrum Symphony MultiCluster
- Multidimensional schedule
- Multitenancy infractructure
- IBM Spectrum Symphony Developer Edition

5.1 IBM Spectrum Symphony overview

IBM Spectrum Symphony is an enterprise-class grid manager for running distributed application services on a scalable, shared, heterogeneous grid. It can support up to 5,000 compute nodes, 128,000 cores, and 300 applications. Designed with the flexibility to adapt when priorities change, it can reallocate more than 1,000 compute engines per second to different workloads in accordance with sharing policies and application priorities you define. This translates into better application performance, better utilization, and an ability to respond quickly to business demands.

A fair-share scheduling scheme with 10,000 priority levels can be used for multiple jobs for a single application. Also, preemptive and resource-threshold-based scheduling is available with runtime change management. Slot allocations change dynamically based on job priority and server thresholds, loaning, and borrowing. IBM Spectrum Symphony also supports multidimensional scheduling, with the ability to specify slot definition for each consumer in terms of up to four customizable resource metrics (core, memory, disk I/O and network I/O).

5.1.1 Key highlights of IBM Spectrum Symphony

IBM Spectrum Symphony includes the following functionality:

- Componentized installation and architecture enables more flexible way to install IBM Spectrum Symphony and its complementary products (IBM Spectrum Conductor 2.1, IBM Spectrum Conductor with Spark 2.1, or IBM Spectrum LSF 10.1).
- Upgraded big data integration to support the current versions of Apache Hadoop, YARN, and Ambari for IBM Open Platform (IOP) BigInsights® and for high availability (HA) in an Ambari integrated environment.
- Extended support matrix for more or upgraded versions of operating systems, browsers, file systems, distributions, frameworks, databases, and software.
- YARN fine-grained workload scheduler for YARN applications running in an EGO-YARN environment. Now get resources directly from EGO multidimensional scheduling, eliminating resource use required for workload scheduling from the YARN resource manager in previous versions.
- Seamlessly migrate the SSM for an application from one server to another, without disrupting workload.
- Current and previous versions of the RESTful APIs so you have a choice. If you have scripts that were written for the previous API version, you are not required to change those scripts, but can move to the current version to take advantage of the current functionality.

Figure 5-1 illustrates the architectural components of IBM Spectrum Computing products.

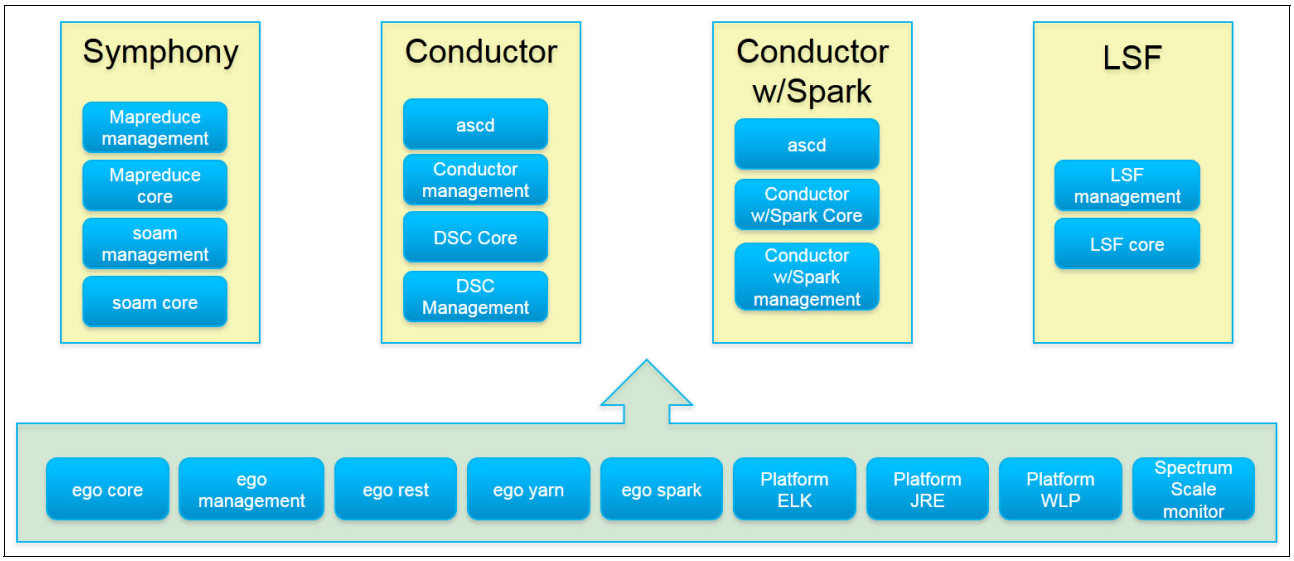


Figure 5-1 IBM Spectrum Computing package components

5.1.2 Scalability

Table 5-1 outlines the tested boundaries for IBM Spectrum Symphony. Use this information as a guide to how much you want to scale your environment for use with IBM Spectrum Symphony.

Table 5-1 Tested boundaries for IBM Spectrum Symphony

Description	Value
Maximum number of compute hosts per cluster	5000 compute hosts
Maximum number of EGO services per cluster	1000 EGO services
Maximum number of EGO service instances per cluster	40000 EGO service instances
Maximum number of cores per cluster	128,000 cores
Maximum number of IBM Spectrum Symphony application service instances per application	40,000 service instances
Maximum number of live sessions per application	10,000 sessions
Maximum number of live tasks per application	2,500,000 tasks
Maximum number of concurrent users running operations on the cluster management console or the multicluster cluster management console	20 users
Maximum number of silo clusters for IBM Spectrum Symphony MultiCluster	20 clusters

Componentized installation and architecture

In this version of IBM Spectrum Symphony, when you install IBM Spectrum Symphony, you install several packages:

- EGO (enterprise orchestrator) packages for the resource management layer
- SOAM (service-oriented application manager) packages for IBM Spectrum Symphony
- MapReduce, YARN, and Spark on EGO packages for IBM Spectrum Symphony Advanced Edition Linux

Separating out the packages provides a more flexible system architecture: You can then easily upgrade or uninstall each package, or include complementary IBM Spectrum Computing family products in your cluster (such as IBM Spectrum Conductor 2.1, IBM Spectrum Conductor with Spark 2.1, or IBM Spectrum LSF 10.1).

5.1.3 IBM Spectrum Symphony MapReduce, YARN, and Docker integration

IBM Spectrum Symphony Advanced Edition includes an Apache Hadoop-compatible MapReduce implementation optimized for low latency, high reliability, and policy-based resource sharing. Unlike the open source solution, which has no capability to restart failed services automatically, this function is built into the implementation of IBM Spectrum Symphony MapReduce, helping to improve reliability. IBM Spectrum Symphony Advanced Edition can also run multiple MapReduce and non-MapReduce jobs concurrently for better throughput and resource utilization.

Integrating YARN with IBM Spectrum Symphony provides a robust setup that takes advantage of the resource scheduling functionality of both YARN and IBM Spectrum Symphony resource manager. It also provides high availability for YARN resource manager entities. Service instances can be run in a Docker container. This allows applications to be packed together into a portable Docker image that can run on different platforms, isolates applications from each other and the underlying infrastructure, and provides an added layer of application protection.

This version of IBM Spectrum Symphony supports the following technology for big data integration:

- Hadoop 2.7.2
- YARN 2.7.2
- Ambari 2.2 for IBM Open Platform (IOP) BigInsights 4.2

Additionally, this version of IBM Spectrum Symphony supports IBM Spectrum Symphony native high availability (HA) in an Ambari integrated environment.

Information: For more information about IBM Spectrum Symphony, see the IBM Spectrum Sympony Overview two IBM Knowledge Center.

5.2 IBM Spectrum Symphony editions

IBM Spectrum Symphony is available in four different editions that are tailored to different business requirements:

- Developer: Build and test applications without needing a full-scale grid
- Express: The ideal solution for departmental clusters

- Standard: Enterprise-class performance and scalability
- Advanced: Ideal for distributed compute- and data-intensive applications requiring Hadoop MapReduce, or benefiting from the advanced capabilities of the Grid synchronization, recursive workload, multiple task service (MTS), multi-cluster management, Ambari integration support, and YARN integration support

Table 5-2 summarizes the features that are associated with each IBM Spectrum Symphony edition.

Table 5-2 IBM Spectrum Symphony features

Features	IBM Spectrum Symphony Edition			
	Developer	Express	Standard	Advanced
Low-latency HPC SOA	X	X	X	X
Agile service and task scheduling	X	X	X	X
Dynamic resource orchestration		X	X	X
Standard and custom reporting			X	X
Desktop, server, and virtual server harvesting capability			X	X
Data affinity				X
MapReduce framework support	X			X
Grid synchronization				X
Recursive workload				X
Multiple task service (MTS)				X
Multicluster management				X
Ambari integration support				X
YARN integration support				X

Product edition and add-on entitlement

Entitlement (or licensing) of the Express, Standard, and Advanced editions is controlled by an encrypted key in an entitlement file that enables a specific product edition and associated features using a common installation package. The entitlement configuration file is verified by the entitlement configuration library to determine which IBM Spectrum Symphony edition and supported features are enabled. The entitlement has no expiry date but can limit the number of cores, as in the case of the IBM Spectrum Symphony Express Edition. The Developer Edition does not require an entitlement configuration file.

Product add-ons are optional and serve to enhance the functionality of IBM Spectrum Symphony Standard Edition and IBM Spectrum Symphony Advanced Edition. The following add-ons are supported:

- IBM Spectrum Symphony Desktop Harvesting
- IBM Spectrum Symphony GPU Harvesting
- IBM Spectrum Symphony Server and VM Harvesting
- IBM Spectrum Symphony Co-Processor Harvesting

5.3 IBM Spectrum Symphony MultiCluster

This section describes the IBM Spectrum Symphony MultiCluster.

5.3.1 Cluster management

A *cluster* is a logical grouping of hosts that provides a distributed environment on which to run applications. IBM Spectrum Symphony manages the resources and the workload in the cluster.

When using IBM Spectrum Symphony, resources are virtualized. It dynamically and flexibly assigns resources, provisions, them and makes them available for applications to use.

IBM Spectrum Symphony can assign resources to an application on demand when the work is submitted, or assignment can be predetermined and preconfigured.

Cluster components

An IBM Spectrum Symphony cluster manages both workload and resources. IBM Spectrum Symphony maintains historical data, includes a web interface for administration and configuration, and has a command-line interface for administration.

Application

IBM Spectrum Symphony MultiCluster (MultiCluster), is a collection of resources that spans clusters that can be moved between clusters based on need. This process involves a MultiCluster master cluster with a master host, and one or more MultiCluster silo clusters with compute hosts and a management host. MultiCluster is available for clusters running IBM Spectrum Symphony Advanced Edition for IBM Spectrum Symphony workload (MapReduce applications are not supported).

The clusters that you want to join the MultiCluster must have hosts running the same version of IBM Spectrum Symphony Advanced Edition (for example, all hosts within MultiCluster can use IBM Spectrum Symphony 7.1.1). IBM Spectrum Symphony clusters running the Developer, Standard, or Express editions cannot join the MultiCluster.

Using IBM Spectrum Symphony MultiCluster, you can enable workload placement to distribute workload among clusters and applications, and repurpose hosts between clusters to satisfy peak load. MultiCluster, provides a set of **`smcadmin`** commands, and also a MultiCluster Management Console, which you can use to manage and monitor clusters and applications.

Figure 5-2 depicts an IBM Spectrum Symphony MultiCluster environment.

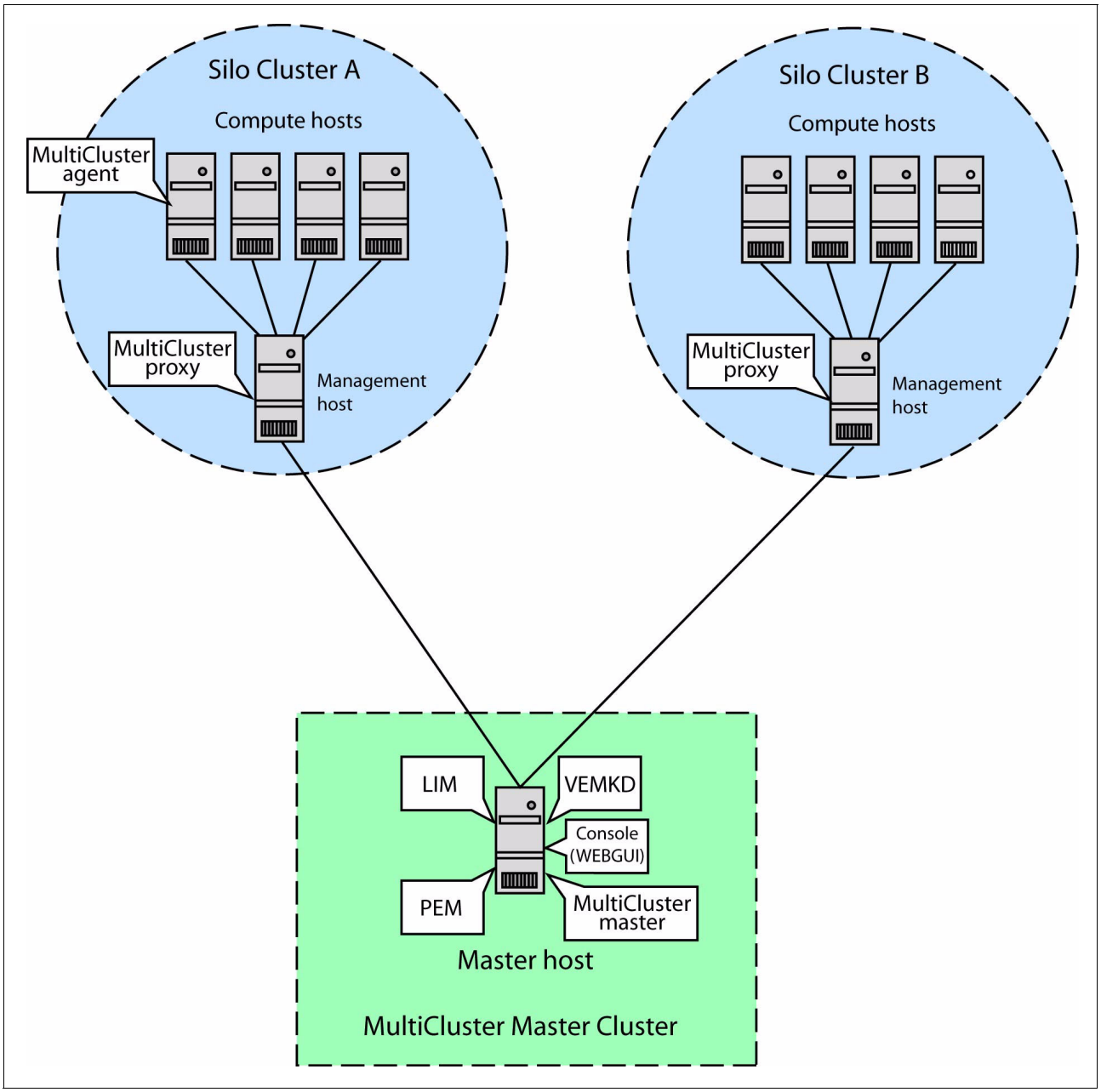


Figure 5-2 IBM Spectrum Symphony MultiCluster environment

Note: All hosts within MultiCluster (including the master cluster and silo clusters) must use physical machines, not virtual ones.

5.3.2 MultiCluster master cluster

The MultiCluster master cluster hosts the MultiCluster master services and GUI. It contains a master management host running IBM Spectrum Symphony. This master cluster is composed of software components that run outside of individual clusters to manage movement of resources between clusters and to coordinate and track them.

There is only one master host in the MultiCluster master cluster. The master candidate is a host that can become the master host if the MultiCluster master host fail. If the master host fails, one of the master candidate hosts automatically takes over the master host role.

Tip: Refer to Recovery for a failed MultiCluster master cluster for details about creating a backup MultiCluster master cluster in case the MultiCluster master cluster fails.

See IBM Knowledge Center for information about Failure recovery for MultiCluster master cluster in Symphony 7.1.

MultiCluster components

The MultiCluster master cluster is composed of the following components:

- MultiCluster master

  The MultiCluster master communicates with the silo clusters using the MultiCluster proxy on each silo cluster. For example, the MultiCluster master gathers resource information collected by each MultiCluster proxy on each silo cluster. The MultiCluster master can then accept resource repurposing requests, and invokes the MultiCluster proxies to move the resources. It also provides a cluster registration service for adding clusters to MultiCluster or removing clusters from MultiCluster.

- VEMKD

  The EGO kernel daemon (VEMKD) daemon supports host failover and is responsible for starting EGO services including the SMCP service (for the MultiCluster proxy on silo clusters) and the WEBGUI service (for running the MultiCluster Management Console).

- Load Information Manager (LIM)

  The LIM daemon starts VEMKD on the MultiCluster master host.

- MultiCluster Management Console

  The MultiCluster Management Console can be accessed through a web browser. To use the console, the minimum window resolution required is 1024 x 768. The web server runs as the WEBGUI daemon. Use the console to manage all your MultiCluster tasks; the top of the console includes links to the three main areas:

  - Workload

    Under the Workload menu, a MultiCluster user can work with the applications they have been authorized to manage, including these tasks:

    - Access an overall dashboard view of the applications within the MultiCluster.
    - View a list of applications within the MultiCluster.
    - Manage the settings for applications and workload bins. For application workload placement, you can place applications into workload bins; each bin can have rules and filters that define the workload placement for the application.

  - Resources

    Under the Resources menu, a MultiCluster administrator can manage clusters, including these tasks:

    - Access an overall dashboard view of the clusters within the MultiCluster.
    - Manage the clusters that are current members of the federation cluster (a group of IBM Spectrum Symphony clusters managed by MultiCluster), or clusters waiting to participate.
    - Manage resource plans, including repurposing hosts to move hosts between clusters and monitor progress.

- Manage a global resource view so that you have a single point of access to all the resource plans in all your clusters monitored in MultiCluster. You can view planned allocations per time interval for all consumers.

– System

Under the System menu, MultiCluster administrator manages MultiCluster settings, including these tasks:

- Manage MultiCluster operations.
- Manage workload placement settings, including enabling clusters and applications to participate in, and defining global rules for, workload placement.
- Manage dashboard display settings.
- Manage users, including creating MultiCluster accounts and assigning MultiCluster user roles.

5.3.3 MultiCluster silo clusters

You can have one MultiCluster master cluster, but one or more MultiCluster silo clusters (in the diagram, there are two: Silo Cluster A and Silo Cluster B). For each silo cluster that you want to manage, you enable a MultiCluster proxy on a management host. The SMCP service is the MultiCluster proxy, and is responsible for discovering hosts in the cluster and triggering actions to repurpose them. There is one MultiCluster proxy per cluster.

The MultiCluster agent is a daemon that runs on demand on a host being repurposed. It is responsible for stopping and starting LIM, and starting move-in and move-out scripts and reporting their results to the MultiCluster proxy.

The MultiCluster master periodically checks for disabled silo clusters and removes them from the database if they do not connect to the MultiCluster master within 120 seconds. Disabled clusters are removed from the cluster view in the MultiCluster Management Console.

Note: MultiCluster does not support the use of duplicate host names across different silo clusters.

5.3.4 MultiCluster roles

IBM Spectrum Symphony MultiCluster supports two roles: the MultiCluster administrator and the MultiCluster user.

The MultiCluster administrator has full access (that is, view and control permissions) to MultiCluster features using the MultiCluster Management Console or the **`smcadmin`** command.

A MultiCluster user, by default, does not have any view or control access. However, for application workload placement the administrator can explicitly grant users access to manage (view and control) applications based on their leaf consumers. For example, to allow a MultiCluster user to administer application ABC, the administrator edits the permissions file (called `SmcUserAppPerms.xml`) that explicitly assigns the user to application ABC and specifies the type of access (view, control, or both).

Note: The application workload placement is the only MultiCluster feature where a user can be assigned view and control access. The administrator, not the user, manages all the other features (such as creating users and roles, system settings, global (cluster) workload placement, repurposing requests, and so on).

5.4 Multidimensional schedule

Enterprise grid orchestrator (EGO) multidimensional scheduling, as the name suggests, can schedule multiple dimensions of resources. If you have experience with other resource schedulers, such as YARN or Mesos, you are familiar with multidimensional scheduling. Multidimensional scheduling is available with IBM Spectrum Symphony 7.1.2.

Heterogeneous workload and slot-based resource scheduling

Before the introduction of multidimensional scheduling, resources were abstracted as slots. With slot-based scheduling, the administrator defines the number of slots for each host. Each slot is distributed to applications according to the resource plan, where one slot can run one task. To a host with N slots, each slot is considered to have 1/N of the power of the host and has access to 1/N of every resource available on the host.

With multidimensional scheduling, when an application requests resources from a resource manager, it can specify the quantity of resources in multidimensional ways, such as `cpu=2,mem=4G`. This type of scheduling is quite different from what you are accustomed to with IBM Spectrum Symphony 6.1.1 or earlier.

The simplicity of slots-based scheduling concept works perfectly when all tasks are equally demanding for CPU power, and for all kinds of other resources. However, this simplicity can become a limitation for advanced use cases, where some tasks demand more CPU than others, some tasks demand more memory than others, and so on. A user needs to define slots to be big enough to accommodate both types of tasks, resulting in defining slots with too many resources so that they cannot be fully utilized. For quite some time, users have been asking for a way to allow more efficient use of resources for heterogeneous workload.

The dynamic service feature is a solution for heterogeneous workload. With this feature, applications can be configured to use multiple slots to run one task or use one slot to run multiple tasks. This way, a task that demands more CPU can use more slots than a task that demands less CPU, assuming memory is always enough or irrelevant.

Heterogeneous workload and multidimensional scheduling

Starting from IBM Spectrum Symphony 7.1, you can also use multidimensional scheduling, which is a superior solution as each task can be allocated with accurate resources, improving the overall resource utilization, and thus, increasing workload throughput. For example, when CPU-demanding tasks run together with memory-demanding tasks on the same host, more total tasks can be started simultaneously.

Multidimensional scheduling is highly customizable. It does not only support CPU, memory, it can also support up to five user-defined resources (such as network bandwidth, disk space, or just any resource, if the user knows how to define it).

Multidimensional scheduling has been integrated with multiple third-party resource or workload managers: YARN, Mesos, Spark, to name a few. IBM Spectrum Symphony workload is now able to share resources with these third-party workloads using resource sharing policies that guarantee SLA and by configuring resource plans that involve ownership, share ratio, and so on.

Supported distributed files systems for MapReduce, Spark, or YARN

The following is the list of supported distributed file systems for MapReduce, Spark, and Yarn.

1. IBM Spectrum Symphony supports the following distributed file systems for integrating MapReduce and Spark, as shown in Table 5-3.

Table 5-3 Supported distributed files systems for MapReduce and Spark

Application	Version	Notes
IBM Spectrum Scale-FPO	4.1.1 4.2.1	None
Apache Hadoop Distributed File System (HDFS)	2.7.2	Open-source Hadoop distribution with both MRv1 and MRv2 applications is supported in stand-alone IBM Spectrum Symphony environment. BigInsights Hadoop distribution is supported in IBM Spectrum Symphony-enabled BigInsights environment
Cloudera CDH	5.5.1	Verified with IBM Spectrum Symphony on Linux
MapReduce	3.0.2	None

2. IBM Spectrum Symphony supports the following distributed file systems for integrated YARN jobs, as shown in Table 5-4.

Table 5-4 Supported distributed files systems for YARN

File system	Version
Apache Hadoop Distributed File System (HDFS)	2.7.2

5.5 Multitenancy infractructure

Multitenancy is an architecture in which a single instance of a software application serves multiple customers. Each customer is called a *tenant*. Tenants might be given the ability to customize some parts of the application, such as color of the user interface (UI) or business rules, but they cannot customize the application's code.

Note: A *Building a Multitenant Analytics Infrastructure* demonstration video is available on YouTube (you need to be signed in with your IBM ID).

5.5.1 The narrow view of multitenancy

Note: In a multitenancy environment, multiple customers share the same application that runs on the same operating system, on the same hardware, and with the same data-storage mechanism.

Big data and analytics infrastructure silos are inefficient. IBM Spectrum Symphony helps you achieve the best results by using multitenancy.

5.5.2 Advantages and challenges

By using a shared infrastructure environment, this service reduces hardware, software, and environmental costs and at the same time maintaining a secured infrastructure through isolated LPARs and IBM comprehensive managed services. The solution offers an allocation-based consumption model that further reduces costs, so you pay for only what is allocated to you. The savings are from extending the cost of hardware and software across the entire multitenant customer environment. The service also provides dynamic capacity to meet peak workload requirements and growth as business needs change.

There are several challenges associated with this solution:

- Increasing cost of analytics
- Addressing challenges associated with the ETL (extract, transform, load) process
- Accommodating data warehouse volume growth
- Delivering needed information in a timely manner
- Ensuring that information is available when needed
- Management of the Hadoop environment

Ever-increasing expectations increase technical needs, too:

- Increased performance to support business demands
- Increased scalability to address huge and growing volumes of data
- Optimized use of existing resources for scaled performance
- Efficient data management to remove data bottlenecks
- Support for new, cloud-native application workload patterns
- Effective operational management: monitoring, alerts, diagnostics, and security

5.5.3 Multitenant designs

In general, multitenancy implies multiple unrelated users or customers of a set of services. Within a single organization, this can be multiple business units with resources and data that must remain separate for legal or compliance reasons. Most hosting companies require multitenancy as a core attribute of their business model. This might include a dedicated physical infrastructure for each hosted customer or logical segmentation of a shared infrastructure by using software-defined technologies.

In IBM Spectrum Symphony Advanced Edition, up to 300 MapReduce runtime engines (*job trackers*) can coexist and use the same infrastructure.

Users can define multiple MapReduce applications and associate them with resource consumers by “cloning” the default MapReduce application. Each application has its separate and unique job tracker called session manager (SSM). When there are multiple instances of SSMs, they are balanced on the available management nodes.

Within each application, simultaneous job management is possible because of the design that implements sophisticated scheduling of multiple sessions on the resources that are allocated for an application. This function is possible by separating the job control function (workload manager) from the resource allocation and control (Enterprise Grid Orchestrator). The new YARN, Apache Hadoop 2, has a similar feature, but the release is still in alpha stage. The stable release of Hadoop MapReduce offers only one job tracker per cluster.

Moreover, multitenancy is much more than just multiple job trackers. It is about user security, shared and controlled access to the computing resources and to the whole environment, monitoring and reporting features, and so on. These multitenancy features are addressed as they are implemented by the IBM Spectrum Symphony solution.

5.5.4 Requirements gathering

Requirements gathering helps you determine how the consumer can be aware of the hosted applications and how is the best way to request access to these hosted services.

While gathering requirements, get answers to the following questions:

- Will consumers use accounts that the host creates, or will they use accounts that they use internally to access services?
- Is one consumer allowed to be aware of other consumer's identities, or is a separation required?
- Can multiple consumers share a physical infrastructure?
- Can traffic from multiple consumers share a common network?
- Can software-defined isolation meet the requirements?
- How far into the infrastructure must authentication, authorization, and accounting be maintained for each consumer?

There are also segmentation options to consider as part of a multitenant infrastructure:

- Physical separation by customer (dedicated hosts, network, storage)
- Logical separation by customer (shared physical infrastructure with logical segmentation)
- Data separation
- Network separation (virtual LANS, or VLANs)
- Performance separation (shared infrastructure but guaranteed capacity)

5.6 IBM Spectrum Symphony Developer Edition

IBM Spectrum Symphony Developer Edition (DE) provides an environment for application developers to grid-enable, test, and run their service-oriented applications. Developers can test their client and services in their own cluster of machines before deploying to the IBM Spectrum Symphony grid.

IBM Spectrum Symphony Developer Edition provides a complete test environment, simulating the grid environment provided by IBM Spectrum Symphony. With IBM Spectrum Symphony Developer Edition, you can use two compute hosts in your cluster with no separate application manager; to use up to 5000 hosts and up to 300 separate application managers, you require IBM Spectrum Symphony.

To run IBM Spectrum Symphony workload on the grid, the application developer creates a service package and adds the service executable into the package: no additional code changes are required. The IBM Spectrum Symphony Developer Edition does not include the EGO resource management component. It does include an EGO stub to simulate basic EGO resource distribution.

IBM Spectrum Symphony Developers Edition provides the following features:

- Easy-to-use APIs and rich design patterns to seamlessly grid-enable all types of service-oriented applications with minimal changes
- The cluster management console is your window to IBM Spectrum Symphony, monitoring and controlling your test environment for IBM Spectrum Symphony and MapReduce workloads
- A command line interface (CLI) is available for monitoring the IBM Spectrum Symphony cluster and performing operations on it
- A MapReduce framework to run MapReduce applications with minimal changes. The MapReduce framework in IBM Spectrum Symphony Developer Edition provides:
  - Different modes for debugging MapReduce applications:
    - The stand-alone mode in which the entire MapReduce workflow runs in a single Java process on the local host
    - The pseudo-distributed mode in which each MapReduce daemon runs in separate Java processes on the local host
  - Support for distributed file systems, such as the open-source Apache Hadoop Distributed File System (HDFS), Cloudera's Distribution Including Apache Hadoop (CDH), Appistry Cloud IQ, and IBM Spectrum Scale
  - A command-line utility called `mrsh` that automatically sets up the environment when submitting MapReduce jobs
  - A MapReduce service class definition that you can customize to implement custom lifecycle event handlers
  - A Java class wrapper that defines buffers as data containers, enabling you to copy and view large files as a sequence of bytes

6

IBM Spectrum Conductor

As described in Chapter 1, “Introduction to IBM Spectrum Computing” on page 1, the digital world is generating tremendous amounts of data, and in an era of rapid change and increasing competition, organizations need a more intelligent infrastructure that is agile, flexible, and resilient. IBM Spectrum Conductor delivers those capabilities with robust features that enable organizations to store, analyze, and protect their data with optimum efficiency.

This chapter describes the following topics:

- IBM Spectrum Conductor
- IBM Spectrum Conductor with Spark
- IBM Conductor with Spark installation

6.1 IBM Spectrum Conductor

IBM Spectrum Conductor is an integrated application and data-optimized platform that enables organizations to achieve up to 60% faster results with new generation, cloud-native applications and open source frameworks. IBM Spectrum Conductor provides policy-based, workload-aware resource management, ensuring maximum availability of shared services with intelligent application and data lifecycle management.

IBM Spectrum Conductor maximizes resource utilization with a shared, multi-tenant infrastructure, by efficiently analyzing, accessing, and protecting data. Resource and data sharing can be significantly increased without compromising availability or security.

IBM Spectrum Conductor achieves these goals in the most cost-efficient way possible through three core capabilities as shown in Figure 6-1:

- Analyze. Workload and data-aware platform increases usage of existing hardware resources and speeds analysis
- Access. Policy-driven data placement with global access enables organizations to tackle all facets of storing data and sharing resources for distributed teams or data centers
- Protect. Enterprise-class features, such as data encryption, automatic failover, and seamless system recovery, provide enhanced resilience and protection

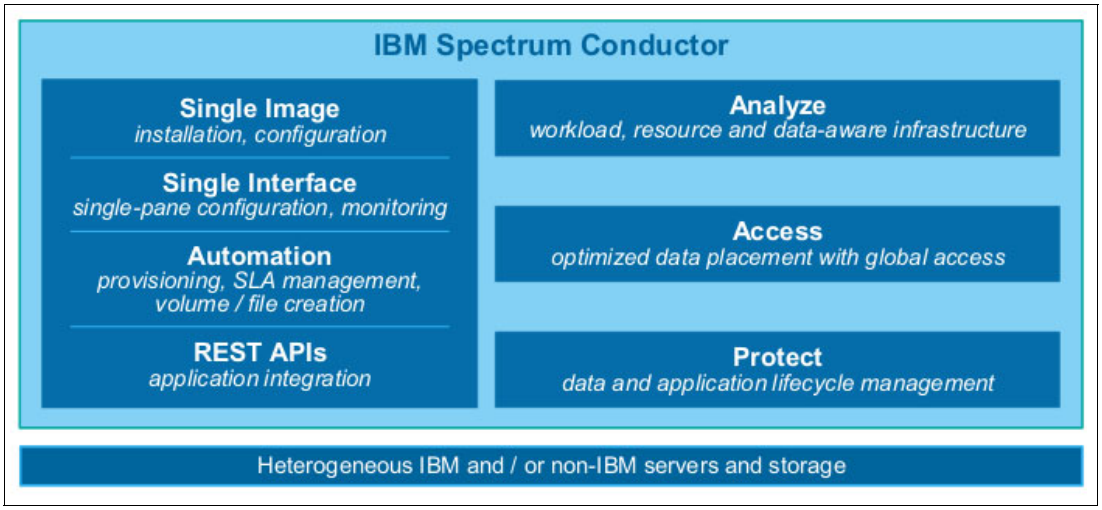


Figure 6-1 IBM Spectrum Conductor solution

6.2 IBM Spectrum Conductor with Spark

IBM Spectrum Conductor with Spark is a complete enterprise-grade, multi-tenant solution for Apache Spark. Known previously as IBM Platform Conductor for Spark, IBM Spectrum Conductor with Spark is now available as a product offering in the IBM Spectrum Conductor family, along with IBM Spectrum Conductor. Together, both products enable organizations to achieve faster results by efficiently analyzing, accessing, and protecting data.

IBM Spectrum Conductor with Spark is a multitenant solution for Apache Spark, enabling you to efficiently deploy and manage multiple Spark deployments. You can manage your Spark deployments independent of your infrastructure, or you can choose an integrated infrastructure solution, where system infrastructure and storage can be monitored and managed along with your Spark deployments.

6.2.1 IBM Spectrum Conductor with Spark value proposition

As an end-to-end solution, IBM Spectrum Conductor with Spark helps to address the today growth data challenge, eliminates resource silos that are tied to multiple Apache Spark implementations, and provides the following benefits:

- Granular and dynamic resource allocation. Improves performance, resource usage, and efficiency of Spark instance groups through a shared resource pool. Extends beyond Spark and eliminates cluster sprawl
- Multitenancy through Spark instance groups. You can run multiple instances of Spark, including different Spark versions, in a shared environment
- Accelerate result. Run Spark natively on a shared infrastructure without the dependency of Hadoop, helping reduce application wait time and accelerating time to results
- Reduce administration costs. Proven architecture at extreme scale, with enterprise class workload management, monitoring, reporting, and security capabilities
- Enterprise class solution. A tightly integrated offering that combines the IBM supported Spark distribution with workload, resource, and data management. Provides high-performance shared storage through IBM Spectrum Scale FPO technology as well
- Containerized environment. Integrates with Docker containers, enabling you to run Spark

IBM Spectrum Conductor with Spark 2.1.0 supports two primary installation scenarios to better accommodate your requirements:

- Stand-alone. If your hosts are already deployed with a supported operating system and you want to manage Spark deployments, install the stand-alone software packages. The stand-alone package installation supports a *composable* architecture for flexible installation across products in the IBM Spectrum Computing family. Core functionality is broken down into separate modules for greater flexibility, enabling you to easily install or uninstall complementary products such as IBM Spectrum Conductor 2.1.0 and IBM Spectrum Symphony 7.1.2.

  When installing IBM Spectrum Conductor with Spark as stand-alone software, you can install to a shared file system, such as IBM Spectrum Scale, for storage and high availability.

  You can save disk space with a shared file system because files are shared and maintained only in one location. You also conserve network bandwidth as files are not required to be transferred between multiple hosts in your cluster. With deployment to a shared file system, the shuffle service is disabled by default and the shared file system is used to shuffle data, instead of the shuffle service. If required, you can choose to enable the shuffle service to shuffle data across hosts.

Tip: To install to a shared file system, set the `SHARED_FS_INSTALL` environment variable during installation. Refer to Table 6-7 on page 85.

- Bare Metal. If you want to set up and manage your system infrastructure and IBM Spectrum Scale storage along with Spark deployments, install the bare metal package.

  The bare metal installation supports an integrated infrastructure and storage (along with Spark deployments found in the stand-alone software package). With this option, you can take advantage of a fully integrated solution for infrastructure management that provides cluster deployment that includes infrastructure configuration and node provisioning, automatic IBM Spectrum Scale 4.2.1 installation, and atomic Docker installation if enabled.

Important: If you choose a bare metal installation, you must review the requirements for an integrated environment before installing IBM Spectrum Conductor with Spark.

6.2.2 Conductor with Spark Components

The IBM Spectrum Conductor with Spark framework consists of the following components:

- EGO
- ASCD
- Spark distribution
- Notebook framework
- IBM Spectrum Scale FPO
- IBM Spectrum Cluster Foundation

Figure 6-2 shows the IBM Spectrum Conductor with Spark components.

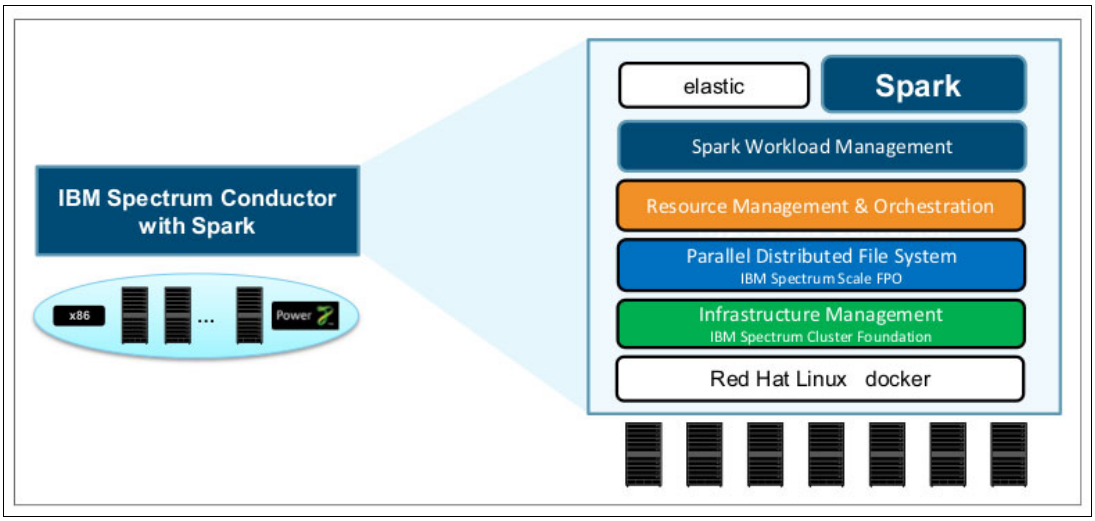


Figure 6-2 IBM Spectrum Conductor with Spark architecture

EGO

IBM Enterprise Grid Orchestrator (EGO) manages the supply of both logical and physical resources, making them available to applications, and provides the underlying system infrastructure to control and manage cluster resources. EGO also provides the underlying system infrastructure to enable multiple applications to operate within a shared multi-tenant infrastructure. EGO integrates with a variety of commercial and open-source frameworks and can coexist and add value to other resource managers like YARN or Mesos for enterprise users.

Just as an operating system running on a single machine aggregates and virtualizes physical resources and allocates them to applications, EGO performs similar functions, but across a distributed environment. EGO uses individual nodes and their operating systems as devices that makeup a large virtual supercomputer.

Agents running on the nodes provide information and execution capabilities that are aggregated into a central master process. The central master acts as the kernel of the distributed environment. Multiple clusters, each with their own master, make up the grid. Each cluster can have multiple types of workload managers running on top of the kernel. A single cluster can scale up to 10,000 nodes, though in practice the largest enterprises have clusters with 4,000 - 5,000 nodes.

EGO components

EGO contains the following main components (Figure 6-3 on page 72):

- Hosts. These are the nodes in an EGO cluster and can be divided into two groups: management hosts and compute hosts. Management hosts provide specialized services to the cluster, while compute hosts run user workload:
  - Management hosts. Provides both cluster and workload management services within the cluster, and are not expected to run workload for users. Master host, all master candidate hosts, and session manager hosts must be management hosts. Other management hosts include the host running the data loaders and data purger for the reporting feature:

    Important: Management hosts all run on the same operating system (all UNIX or all Linux).

    - Master host. The master host is the first host installed in the cluster. The resource manager (`vemkd`) for the cluster resides on this host. The master host controls the rest of the hosts in the cluster and is the interface to the clients of the cluster.
    - Master candidate. There is only one master host at a time. If the master host fails, another host automatically takes over the master host role. Hosts that can act as the master are called master candidates.
    - Session manager host. One or more management hosts run session managers. There is one session manager per available slot on a management host. There is one session manager per application.
  - Compute hosts. Compute hosts are those hosts in the cluster that provide computing resources to consumers. A cluster can contain any number of compute hosts, but must have at least one compute host:
    - CPU slots. A CPU slot is the unit used to measure compute resources. A single CPU slot can run one service instance on a compute host, or one session manager on a management host.

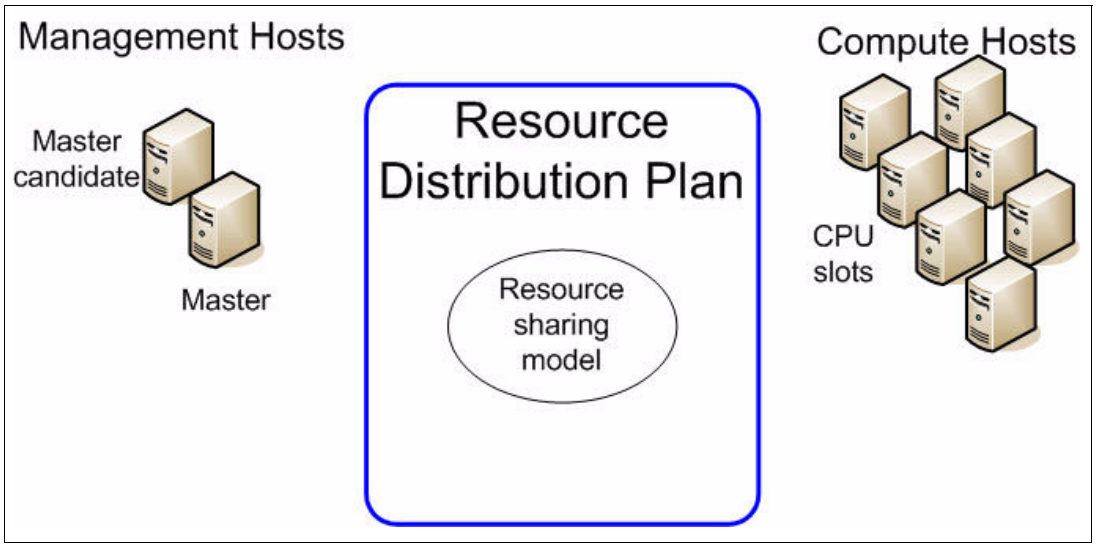


Figure 6-3 EGO main components

- Daemons. These are services in charge of running and control EGO services in the cluster:
  - vemkd. The VEM kernel daemon that runs on the master host. It starts other daemons and responds to allocation requests.
  - egosc. The v service controller requests appropriate resources from the vemkd daemon and controls service instances.
  - Process Execution Manager (PEM). PEM works for the vemkd daemon, providing execution capabilities such as starting and stopping processes or containers, and collecting and sending runtime resource usage. The EGO kernel talks to the PEM by way of the long-lived TCP connections that enable fast parallel start-up of processes.

EGO Resource management

EGO provides a full suite of services to support and manage resource orchestration in a cluster, including cluster management, configuration, and auditing of service-level plans, failover capabilities, monitoring, and data distribution.

EGO uses resource groups to organize the supply of resources. It manages the supply of resources and allocates them to different workloads according to policies. Resource groups can be static or dynamic, using attributes of hosts to define membership.

An agent running on each node collects dynamic load state information about the node from the operating system, and reports it up to a central master. Resources can also be logical entities that are independent of nodes, like software licenses or bandwidth capacity.

Only the resource requirements are considered when allocating resources. Therefore, the business services can use the resources with no interference.

Those resources are used through consumers. Consumers are organizational entities that request resources from the system and are organized hierarchically into a tree to reflect lines of business, departments, projects, and so on. Multiple workload managers can map different types of workloads into the same consumer so that resource sharing is controlled regardless of the form a workload can take.

Figure 6-4 shows the EGO component architecture for resource management.

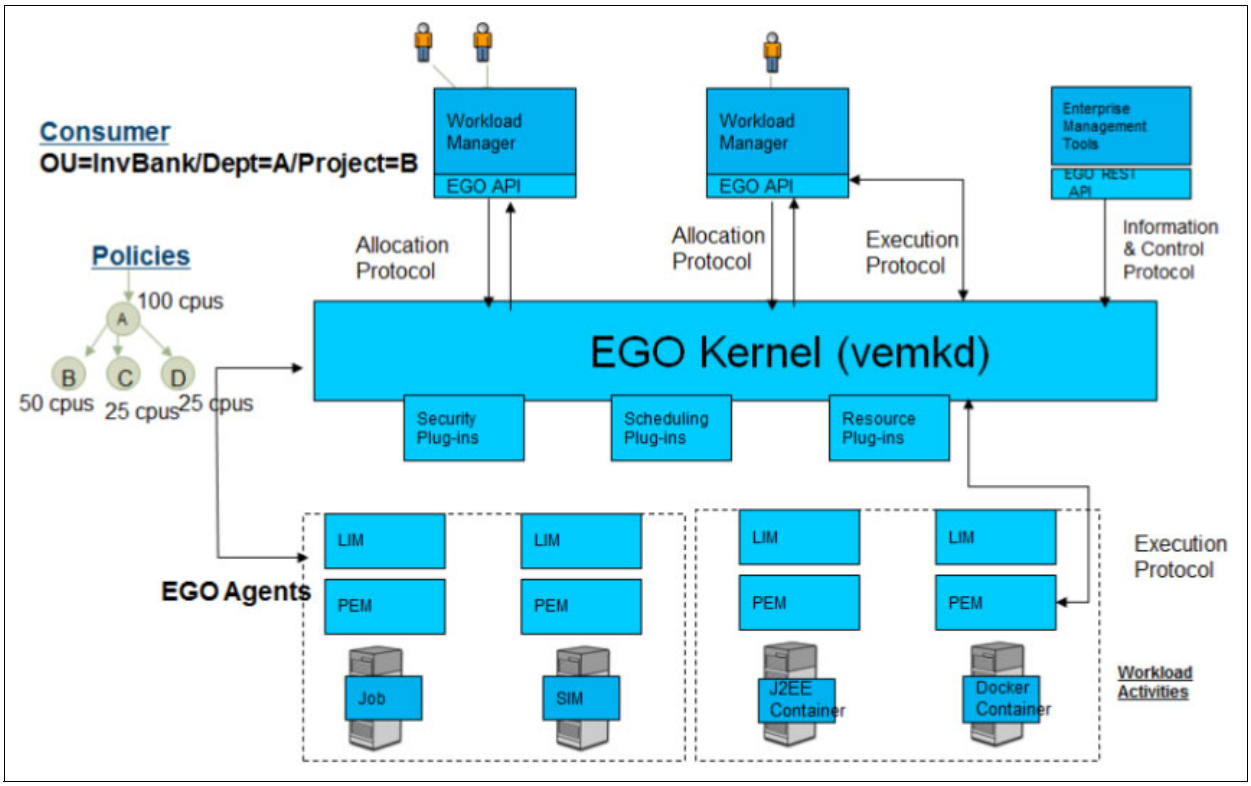


Figure 6-4 EGO architecture for resource management

EGO enables you to implement policies to the amount of resources assigned to different consumers, through the resource plans. These policies are implemented as pluggable shared libraries that allow different strategies to be implemented without affecting the framework.

The resource plan allows SLAs to be created so that each line of business can meet its objectives, and at the same time share a common set of resources. The resource requirements can accommodate multi-dimensional resource allocations where each allocation can request different amounts of physical resource types, including but not limited to processors (CPUs), cores, memory, and number of disks.

ASCD

The application service controller daemon (ASCD) manages the lifecycle of application instances. Registered as an EGO service, the ASCD acts as a Representational State Transfer (REST) application programming interface (API) layer. It enables application instances to be described as a set of inter-related long-running services in a single specification. The services associated with the application instances can then be deployed and scaled.

Spark distribution

IBM Spectrum Conductor with Spark bundles Spark 1.5.2, 1.6.1 as the built-in versions and Apache Zeppelin 0.5.6 as the built-in notebook version.You can remove these versions from the system if you no longer require them, especially if you installed newer versions of Spark or the Zeppelin notebook.

Tip: For more information about deleting versions, see IBM Knowledge Center topics regarding Removing Spark versions and Removing notebooks.

The Spark distribution includes the following components:

- Spark SQL
- Spark Streaming
- MLib Machine Learning Library
- GraphX

Figure 6-5 presents the Spark architecture.

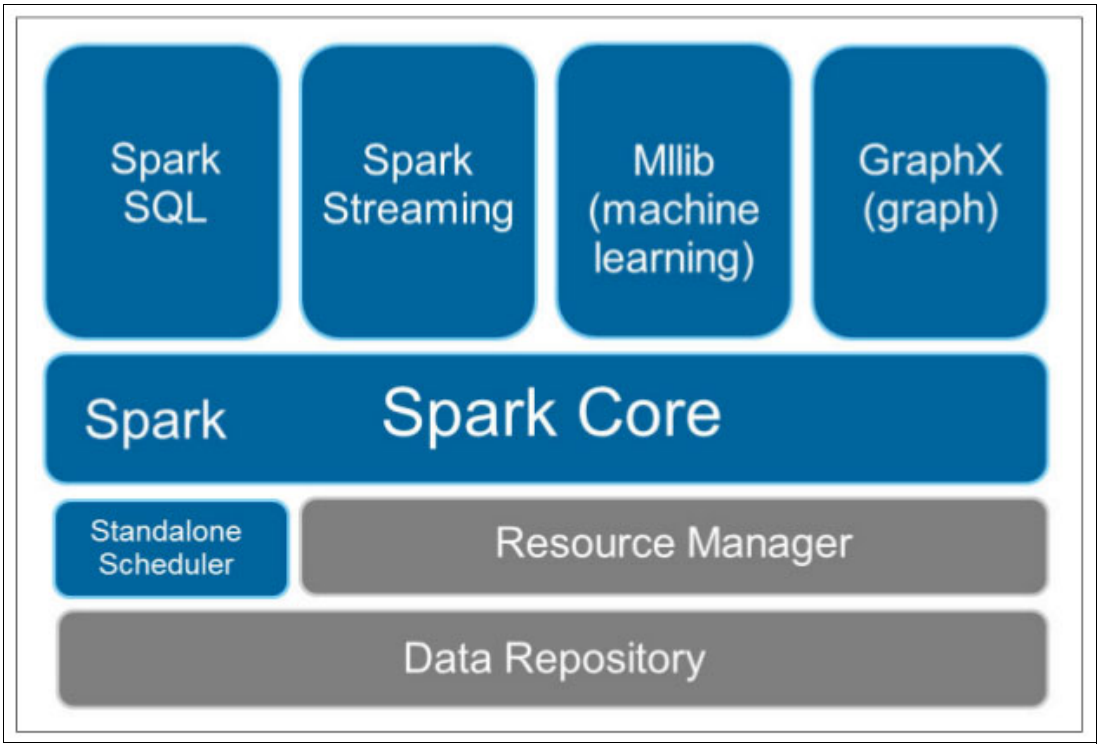


Figure 6-5 Spark software stack

Notebook framework

Notebook framework is used for data manipulation and visualization. The Elastic Stack (Elasticsearch 2.1, Logstash 2.1.1, Kibana 4.3.1, and Filebeat 1.1.1) is integrated within IBM Spectrum Conductor with Spark. Registered as system services, the Elastic Stack integration enables you to search, analyze, and visualize Spark application data for efficient monitoring.

Note: Kibana is also embedded within the management console for data visualization.

IBM Spectrum Scale FPO

IBM Spectrum Scale FPO is a POSIX-compliant storage management technology, and a more space-efficient alternative to HDFS (which is also supported). When using IBM Spectrum Scale, IBM Spectrum Conductor with Spark becomes an end-to-end IBM-supported Spark solution. This component provides the data layer.

IBM Spectrum Cluster Foundation

IBM Spectrum Cluster Foundation provides an infrastructure management layer. It is a graphical administration interface to replace the former Platform Management Console (PMC). IBM Spectrum Cluster Foundation is cover in Chapter 7, “Cluster management with IBM Spectrum Cluster Foundation” on page 103.

6.2.3 Apache Spark applications

Apache Spark applications run as independent sets of processes on a cluster that is coordinated by the Spark Context object in the driver program. The Spark Context object can connect to the cluster manager (either Spark’s own stand-alone cluster manager, Apache Mesos, Hadoop YARN, or EGO in IBM Spectrum Conductor with Spark) through the Session Scheduler, which allocates resources across applications, as shown in Figure 6-6.

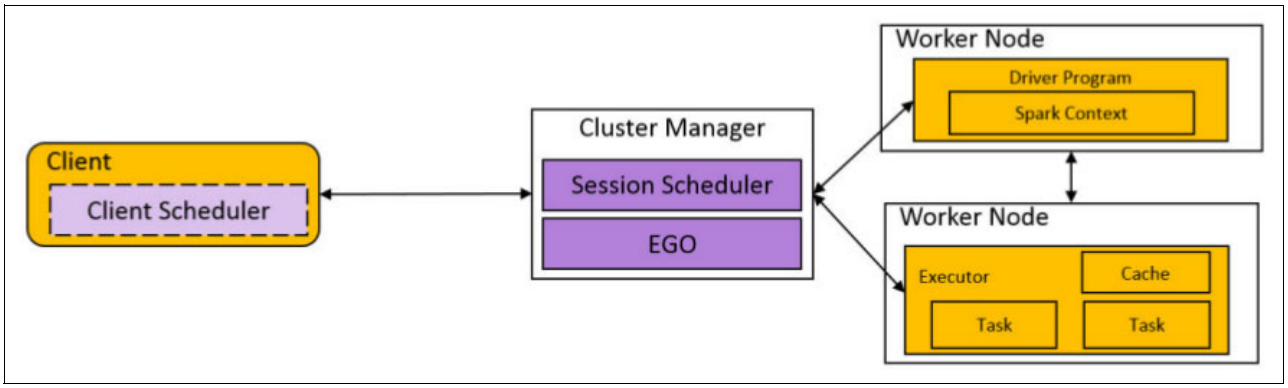


Figure 6-6 Spark integration with IBM Spectrum Computing resource manager

In IBM Spectrum Conductor with Spark, the resource orchestrator (EGO) acts as the cluster manager with Apache Spark, enabling Spark applications to benefit from resource sharing across the cluster. This provides the following benefits:

- Fine-grained scheduling. Apache Spark uses a coarse-grained or fine-grained resource scheduling policy. The Spark application requests a static number of resources and holds them in its lifecycle, which means each application gets more or fewer resources as it scales up and down. Based on the fine-grained scheduling policy, applications share resources at a fine granularity, especially when many applications are running in a cluster concurrently.
- Resource sharing and reclaim. Tenants (known as a Spark instance group) share resources that are managed in consumers and resource groups by the resource orchestrator. Therefore, some tenants can acquire more resources than the resource distribution definition by borrowing them from other applications. When the lender needs more resources, it reclaims these borrowed resources from the borrower. This can keep the cluster in high usage status, and maintain consistency between all tenants.
- Multiple resource scheduling policy (first-in first-out (FIFO)/Fairshare) for each tenant. All tenants partition the resources based on the resource allocation plan.

- Multi-tenant scheduling with Session scheduler. A consumer can run many applications concurrently, and the session scheduler can schedule resources between applications that belong to one consumer.

  Therefore, the resource scheduling is hierarchical, as shown in Figure 6-7.

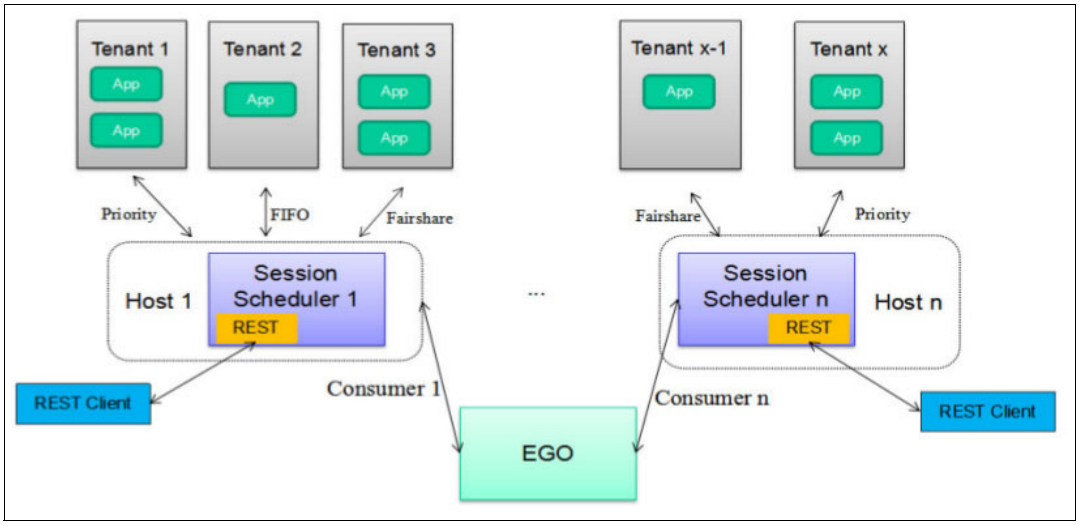


Figure 6-7 Session scheduler flexible scheduling plans

6.2.4 Spark instance group

Within IBM Spectrum Conductor with Spark, a Spark instance group is a Spark tenant; an installation of Apache Spark that can run Spark core services (Master, Shuffle, and History) and notebooks as configured. This provides multitenancy in IBM Spectrum Conductor with Spark.

Multiple Spark instance groups can be created, associating each instance group with different Spark version packages as required. A Spark instance group can be created to serve a line of business or a group member of a business organization.

A Spark instance group contains the following elements:

- Spark core services
- Spark tools and notebook
- User and applications
- Basic isolation

Multidimensional scheduling policy

The Spark instance groups can be configure for either slot-based scheduling (default) or multidimensional scheduling. The multidimensional scheduling policy increases resource utilization by enabling more granular resource allocations for applications with varying resource requirements. It can accommodate multidimensional resource allocations, where each allocation can request different amounts of physical resource types including, but not limited to CPU, cores, memory, and number of disks.

The benefits of multidimensional scheduling are:

- Multiple resource groups in one resource plan
- Named resource ownership (ownership of hosts, including partial ownership)
- Sharing of the share pool resource by priority and ratio
- Reclaim and lend grace periods

SparkCleanup service

The SparkCleanup service cleans up all Spark instance group directories in the cluster, including the Spark working directory, the logs directory used by Elastic Stack to harvest information, and the directories specified by the `spark.local.dir` parameter and the `SPARK_LOCAL_DIRS` environment variable.

Dynamic ports

When Spark instance groups are started in the cluster, the ports required for Spark and notebook services are dynamically determined based on real-time port usage on each host.

If the ports in the Spark instance group configuration are manually specified, these ports are used as the base port when attempting to start Spark services, with a maximum of 16 attempts, or as specified by the `spark.port.maxRetries` parameter.

Attention: For IPython notebooks, port 8888 is used as the base port; any other specify value is ignored.

Instance group filter

Spark instance groups can be filtered to view those associated with a specific consumer. To filter Spark instance groups:

1. Go to **Workload** → **Spark** → **Spark Instance Groups** and click the all consumers link.
2. Select the consumer whose Spark instance groups you want to view.

The Spark instance group list refreshes to show instance groups that are associated with that consumer.

6.2.5 GPU scheduling for Spark applications

Apache Spark allows accelerating applications by using graphics processor units (GPUs) effectively and transparently. This is achieved by an API in resilient distributed dataset (RDD) implemented in both CUDA or OpenCL.

IBM Spectrum Conductor interfaces with Spark scheduler to ensure that GPU resources are assigned to the applications that can use them. A stage marked with GPU is scheduled to the GPU resource group, where others are scheduled to the default resource group.

Figure 6-8 presents a high-level overview of the GPU support functionality.

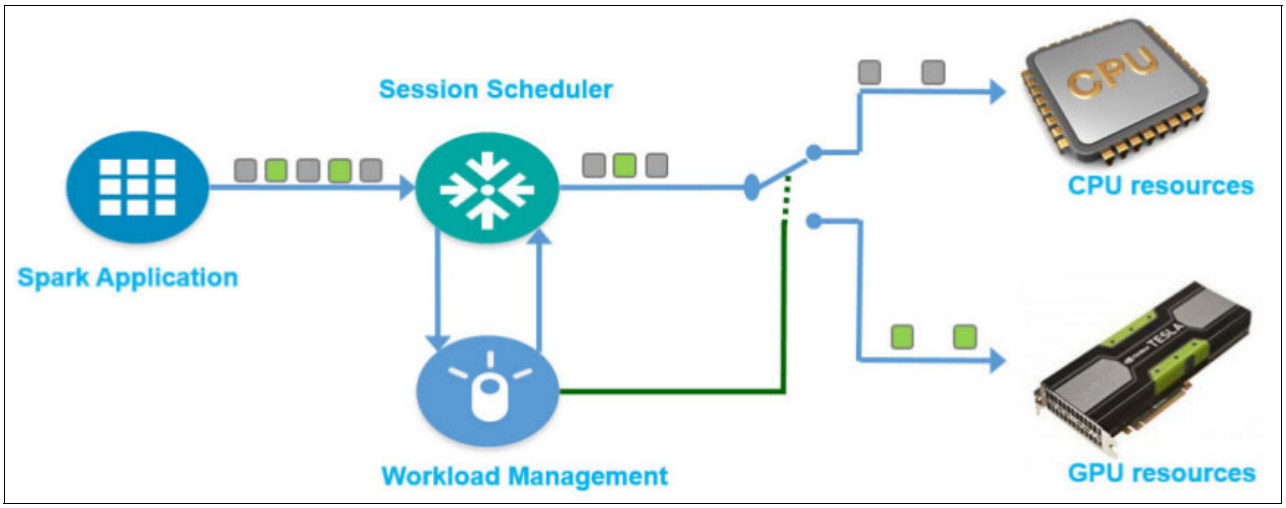


Figure 6-8 GPU resources support in IBM Spectrum Conductor with Spark

When a Spark instance group is enabled for GPU scheduling, GPU slots (in addition to CPU slots) are allocated to Spark executors in the instance group.

To support GPU scheduling, you must set up dedicated resource groups for GPU and CPU resources, enable the creation of Spark instance groups that use GPU resources, create a Spark instance group that allocates GPU slots to applications, and submit a Spark application with GPU to the Spark instance group.

6.2.6 Docker support

With IBM Spectrum Conductor with Spark, you can enable Docker support for Spark instance groups and any associated notebooks. By Dockerizing Spark instance groups, you encapsulate and run its services inside separate Docker containers.

In a bare metal installation, Docker is automatically deployed across your cluster if enabled. If not enabled during installation, then Docker must be installed and configured manually if needed.

Important: IBM Spectrum Conductor with Spark does not provide Docker images. You must provide the required Docker image.

To Dockerize a Spark instance group, there are three components that must be configured during Spark instance group creation: Spark drivers, Spark executors, and Spark instance group services, you can choose to run these components in Docker mode, Cgroup mode, or normally without any special configuration.

To Dockerize the Spark instance group, selecting the Docker mode enables components to run in their own individual Docker containers, creating an isolated environment from the host. This means that each container runs its own tasks regardless of the host environment's configuration. Figure 6-9 shows how to configure Dockerize a Spark instance group.

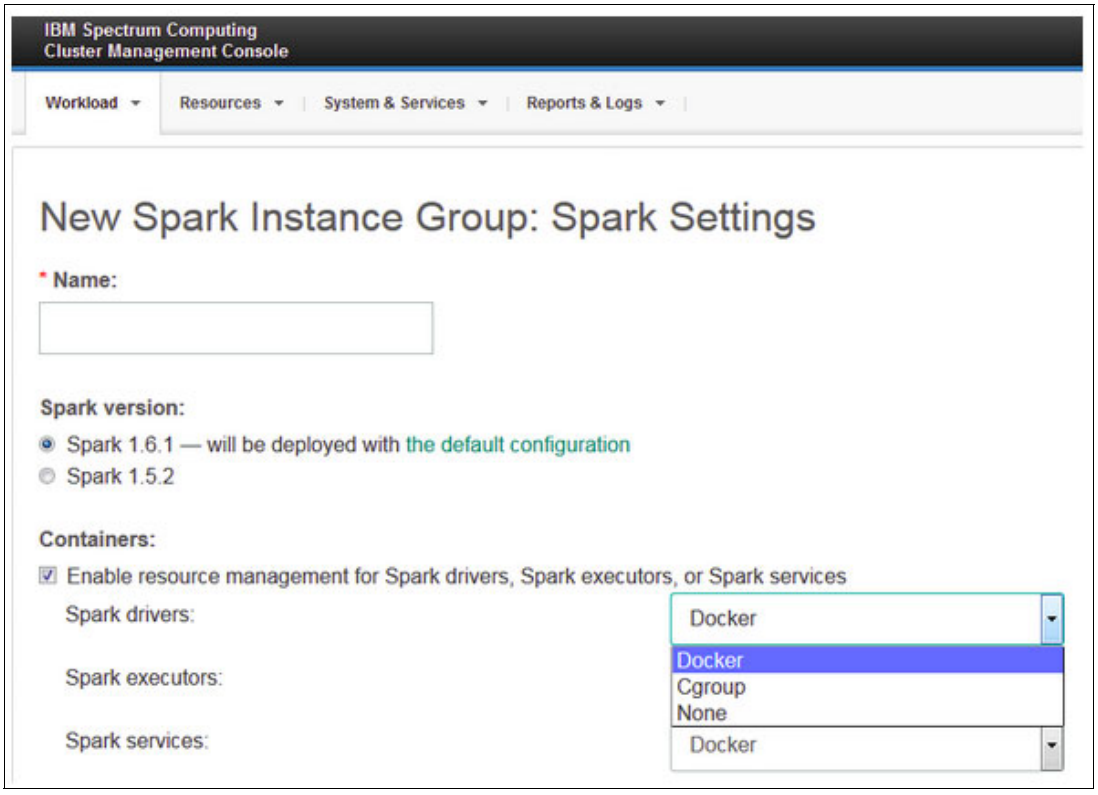


Figure 6-9 Dockerize Spark instance group

Notebooks, such as Apache Zeppelin and IPython-Jupyter, can also be Dockerized by selecting the **Run notebook in a Docker container** option, from the Notebook Management page of the management console. Figure 6-10 shows how to create Dockerized notebooks.

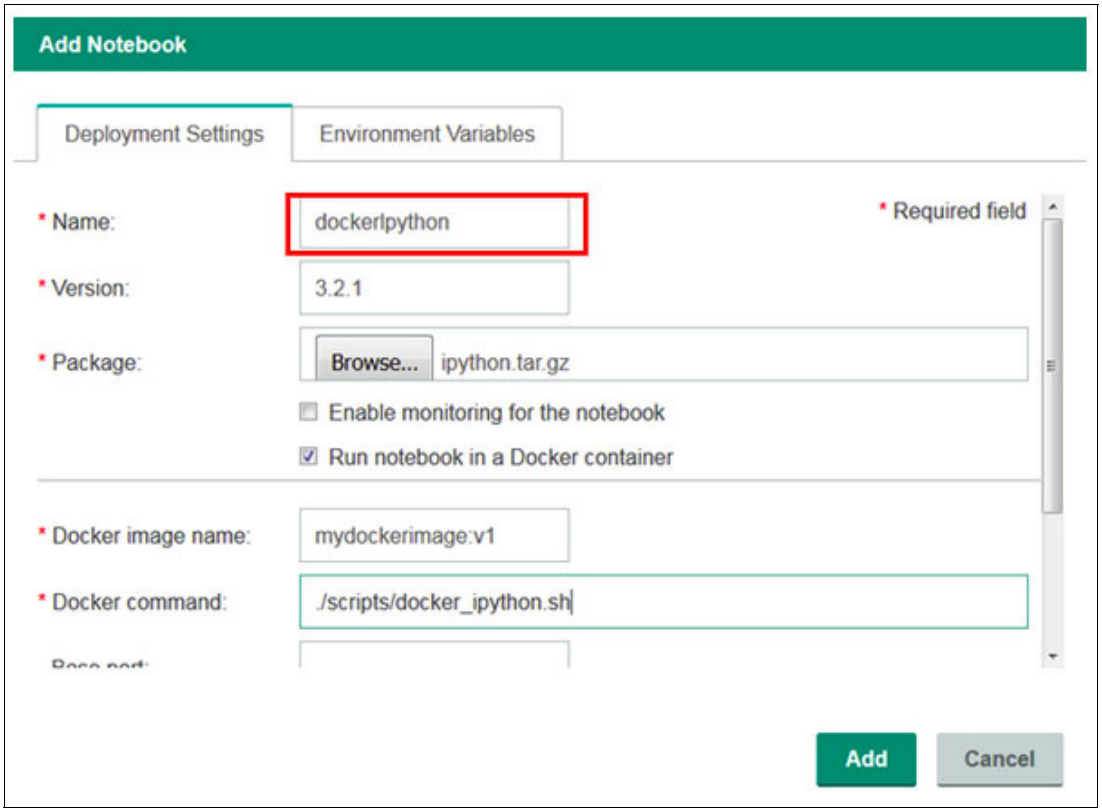


Figure 6-10 Dockerize notebook

6.3 IBM Conductor with Spark installation

IBM Spectrum Conductor with Spark is easy to install, and the following section describes the requirements and steps to install it.

6.3.1 Supported software and hardware requirements

All the supported software and hardware requirements information described in this section is regarding IBM Spectrum Conductor with Spark version 2.1.0, which is the current version available at the time this publication was written.

Hardware requirements

When planning your installation, consider hardware requirements, either for evaluation and production environments.

Table 6-1 on page 81 and Table 6-2 on page 81 list the minimum system requirements for running IBM Spectrum Conductor with Spark in both evaluation and production environments. You might require extra requirements, such as extra CPU, Cores, and RAM, depending on the application instances that will run on the hosts.

Table 6-1 Minimum hardware requirements for evaluation environment

Requirement	Management hosts	Compute hosts
Number of hosts	Without high availability: 1 or more With high availability: 3 or more	1 or more
CPU speed	>= 2.4 GHz	>= 2.4 GHz
Memory	8 GB	8 GB
Disk space to install	6 GB	6 GB
Additional disk space (application instance packages, logs, and so on)	Can be 30 GB for a large cluster	1 GB*N slots + sum of service package sizes (including dependencies)

Table 6-2 Minimum hardware requirements for production environment

Requirement	Management hosts	Compute hosts	Notes
Number of hosts	Without high availability: 1 or more With high availability: 3 or more	1 or more	For high availability purposes, a quorum of eligible master hosts is required. A loss of quorum results in an inoperable cluster and potential loss of data. The minimum number of eligible hosts required is dynamically determined as ((number_of_management_hosts/2)+1). To allow for the loss of 1 management host, your cluster must have a minimum of 3 master eligible hosts when high availability is enabled
CPU speed	>= 2.4 GHz	>= 2.4 GHz	
CPU cores	8	1 or more	Choose a modern processor with multiple cores. Common clusters use 2 - 8 core machines. When it comes to choosing between faster CPUs or more cores, it is advised to choose hosts with more cores
Memory	64 GB	32 GB	In general, the more memory your hosts have, the better performance can be achieved.
Disk space to install	6 GB	6 GB	

Requirement	Management hosts	Compute hosts	Notes
Additional disk space (application instance packages, logs, and so on)	Can be 30 GB for a large cluster	1 GB*N slots + sum of service package sizes (including dependencies) **Note:** Disk requirements depend on your application instance workload, and can be more than management host requirements. The higher the number of application instances, the more the required disk space	Disk space requirements depend on the number of Spark instance groups and the Spark applications that you run. Long running applications, such as notebooks and streaming applications, can generate huge amounts of data that is stored in Elasticsearch. What your applications log can also increase disk usage. Consider all these factors when estimating disk space requirements for your production cluster. For optimal performance, look at tuning how long to keep application monitoring data based on your needs

Important: Based on Elasticsearch heap recommendations, configure the Elasticsearch heap in IBM Spectrum Conductor with Spark to use 6~8 GB. The heap size must not be configured to be higher than the CMS garbage collector was designed for, to prevent long stop-the-world pauses (~6 - 8 GB). See IBM Knowledge Center for more information about how to change the heap size.

Supported operating systems

As mentioned in 6.2.1, “IBM Spectrum Conductor with Spark value proposition” on page 69, IBM Spectrum Conductor with Spark supports two primary scenarios:

- The hosts in your infrastructure are already set up and you want to install IBM Spectrum Conductor with Spark (stand-alone scenario).
- You want to set up your infrastructure, provision bare metal nodes, and then automatically install IBM Spectrum Conductor with Spark (bare metal scenario).

Table 6-3 shows the supported operating systems for both scenarios.

Table 6-3 Supported operating systems

Platform	Operating system
Linux 64-bit	► Red Hat Enterprise Linux (RHEL) 6.4 or later ► Red Hat Enterprise Linux (RHEL) 7 or later up to 7.2 ► SUSE Linux Enterprise Server 11 SP2 and SP3 ► SUSE Linux Enterprise Server 12 ► Ubuntu 16.04 LTS
Linux on POWER 64-bit Little Endian (LE)	► Red Hat Enterprise Linux (RHEL) 7.1 and 7.2 ► SUSE Linux Enterprise Server 12 ► Ubuntu 16.04 LTS

Supported file systems for high availability

IBM Spectrum Conductor with Spark supports the following file systems for high availability (HA) as shown in Table 6-4.

Table 6-4 Supported file systems for HA

File system	Version
NFS	4
IBM Spectrum Scale	4.1.1 4.2.1

Supported databases

IBM Spectrum Conductor with Spark supports several databases for reporting purposes as shown in Table 6-5, including a built-in IBM Derby database for demonstration and testing purposes. The Derby database is not certified for production use.

Table 6-5 Supported databases

Database	Version
Oracle Enterprise Edition	11g 12c
IBM DB2®	10.5

Supported web browsers

IBM Spectrum Conductor with Spark is not supported on Windows. You can, however, access the IBM Spectrum Conductor with Spark cluster from Windows hosts using the browser.

Table 6-6 shows the supported web browsers.

Table 6-6 Supported browsers

Platform	Supported browsers	Notes
Windows 64-bit	Internet Explorer 10	On Internet Explorer 10, you must enable TLS 1.2 (from Internet Options > Advanced Settings) to access the cluster management console. You also need to disable the Compatibility View.
	Internet Explorer 11	Although the cluster management console supports disabling of the auto-complete feature, Internet Explorer 11 (and later versions) and Firefox 30 (and later versions) do not support disabling auto-complete.
	Firefox 45 Firefox 46 Firefox ESR 38 Firefox ESR 45	Kibana (embedded within the cluster management console) is supported only on the current versions of Firefox and Firefox ESR.

Platform	Supported browsers	Notes
Linux 64-bit	Firefox 45 Firefox 46 Firefox ESR 38 Firefox ESR 45	Although the cluster management console supports disabling of the auto-complete feature, Firefox 30 and later versions do not support disabling auto-complete. Kibana (embedded within the cluster management console) is supported only on the current versions of Firefox and Firefox ESR.

6.3.2 Installation procedure

IBM Spectrum Conductor with Spark installation is an easy procedure. Use the following roadmap (shown in Figure 6-11) to guide you through the process of installing and configuring IBM Spectrum Conductor with Spark as stand-alone software.

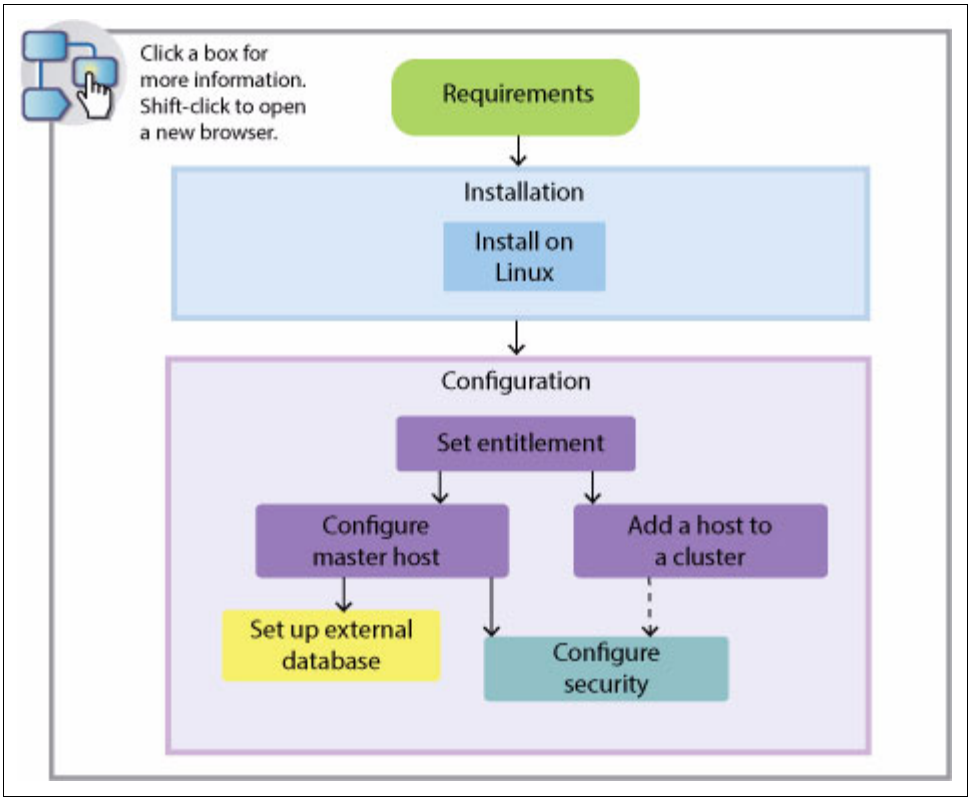


Figure 6-11 Conductor with Spark installation guide

Before installing

Before you install it, check the following requirements:

1. All hosts in your cluster must use fully qualified domain names (FQDN). The names of the hosts must use a valid domain name server (DNS) so that you can resolve the IP address by the domain name, and be able to find the domain name by IP address.
2. All hosts in your cluster must have the same clock setting.

3. Install Python 2.7.5, 2.7.6, 2.7.7, or 3.x on all your hosts.
4. Install OpenSSL 1.0.1 or higher on all your hosts.
5. Install rsh 0.17 in all your hosts.
6. You must install cURL 7.28 or higher for Elastic Stack on all management hosts, and all hosts that will be used to run notebooks.
7. Depending on your environment, make the following adjustments:
   - For RHEL 7, install the net-tools package.
   - For Ubuntu, install the 32-bit compatibility libraries (`lib32z1`) by using the aptitude **`install lib32z1`** command.
   - For SUSE Linux Enterprise Server, set the environment variable **`LANG`** to your locale, such as `en_US`. Do not set to `C`, `c`, or `POSIX`.
8. `In the /etc/security/limits.conf file,` configure the required limits for the maximum number of processors (`nprocs`) and the maximum number of files open (`nofile`) to `65535`. Without this limit, services hang or enter the Error state on cluster startup.
9. Disable SELinux.
10. Your hosts require the **`setfacl`** function from the ACL utility library to display Spark applications in the cluster management console. If this library has been removed or is not installed with your operating system, install it before installing IBM Spectrum Conductor with Spark.
11. Your hosts require the **`gettext`** library to provide globalization support for translated product messages. Without this library, you can encounter a `gettext.sh: file not found` or `gettext: command not found error during installation`. Typically, this library is installed with your operating system; however, if it has been removed or is not installed, install it before installing IBM Spectrum Conductor with Spark.
12. If you cannot install with root permissions on all hosts, use the cluster administrator as the installation account.
13. Install and configure IBM Spectrum Scale.

Cluster properties

Before installing a management or compute host, first define the cluster properties by setting the environmental variables shown in Table 6-7.

Table 6-7 Cluster properties

Option	Description	Requirement level	Notes
CLUSTERADMIN	Set to any valid operating user account, which then owns all installation files	Mandatory	Default value is root.
CLUSTERNAME	Set the name of the cluster	Optional	You cannot change the cluster name after it has been installed. Default cluster name is cluster1. Do not use a valid host name as the cluster name.

Option	Description	Requirement level	Notes
BASEPORT	The cluster uses seven consecutive ports from the base port to assign to the main daemons	Optional	Before installation, make sure that the seven consecutive ports are not in use, this will cause conflicts and EGO will not start properly. The default port number is 7869.
SHARED_FS_INSTALL	When using a shared file system (such as IBM Spectrum Scale) for shared storage and high availability, install IBM Spectrum Conductor with Spark to the shared file system	Mandatory when installing to a shared file system	No need to be set when installing to a local file system.
DSC_MOUNT_POINT	If you want to enable the Data Service Controller (DSC)	Optional	IBM Spectrum Scale must be already installed.
EGOCOMPUTEHOST	Installs the necessary packages for IBM Spectrum Conductor with Spark to run on the compute host	Mandatory when installing on a compute host	
DISABLESSL	If you do not want to enable SSL communication for your web servers, set this option with “Y” value	Optional	SSL is enabled by default
DERBY_DB_HOST	Non-production Derby database, used for reporting	Optional	Set DERBY_DB_HOST whit the master host or another management host that serves as the database host.

Example 6-1 shows the cluster properties on the management host.

Example 6-1 Cluster properties on the management host

```
[root@st6a04-rh-sym1 /]# export CLUSTERADMIN=root
[root@st6a04-rh-sym1 ~]# export BASEPORT=17869
[root@st6a04-rh-sym1 ~]# export DSC_MOUNT_POINT=/gpfs
[root@st6a04-rh-sym1 ~]# export DERBY_DB_HOST=st6a04-rh-sym1
[root@st6a04-rh-sym1 ~]# export DISABLESSL=Y
```

Example 6-2 shows the cluster properties on the compute host.

Example 6-2 Cluster properties on the compute host

```
[root@st6a04-rh-sym3 ~]# export CLUSTERADMIN=root
[root@st6a04-rh-sym3 ~]# export BASEPORT=17869
[root@st6a04-rh-sym3 ~]# export EGOCOMPUTEHOST=Y
[root@st6a04-rh-sym3 ~]# export DISABLESSL=Y
```

If you want to modify the Elastic Stack configuration, set the environment variables as shown in Table 6-8. If you do not set these environment variables, the default values are used.

Table 6-8 Modify the Elastic Stack configuration

Option	Description	Notes
ELK_ESHTTP_PORT	Specifies the HTTP port used by the indexer and Kibana services for communication with Elasticsearch and on which the Elasticsearch RESTful APIs are accessible.	The default port number is 9200.
ELK_ESSYSCOMM_PORT	Specifies the port used for communication within the Elasticsearch cluster.	The default port number is 9300.
ELK_LOGSHIPPER_PORT	Specifies the port on which the indexer service monitors input.	The default port number is 5043.
ELK_KBGUI_PORT	Specifies the port number on which Kibana is accessible.	The default port number is 5601.
ELK_HARVEST_LOCATION	Specifies the directory under which ELK logs are stored. The Elastic Stack uses the log files under this directory to harvest information and support queries.	The default directory is /var/tmp/elk_logs/.

Important: Ensure that the environment variables are the same on all hosts. You cannot change these values after installation.

Install a management host

Management hosts provide specialized services to the cluster. The first management host that you install becomes your master host.

The sections that follow explain the process required to install the management host.

Run the IBM Spectrum Conductor with Spark installation file

After the cluster properties are set, navigate to the directory where you upload the IBM Spectrum Conductor with Spark binary file and run it, as shown in Example 6-3.

Example 6-3 Run the installation file

```
[root@st6a04-rh-sym1 conductor]# ./conductorspark2.1.0.0_ppc64le.bin
Extracting files... done.
Installation for IBM Spectrum Conductor with Spark 2.1.0

IBM Spectrum Conductor with Spark License Agreement
International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, CLICKING ON
AN "ACCEPT" BUTTON, OR OTHERWISE USING THE PROGRAM,
LICENSEE AGREES TO THE TERMS OF THIS AGREEMENT. IF YOU ARE
ACCEPTING THESE TERMS ON BEHALF OF LICENSEE, YOU REPRESENT
```

```
AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND LICENSEE
TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

* DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, CLICK ON AN
"ACCEPT" BUTTON, OR USE THE PROGRAM; AND

* PROMPTLY RETURN THE UNUSED MEDIA, DOCUMENTATION, AND

Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
1
The following commands will install the .rpm packages on the host.
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egocore-3.4.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egojre-1.8.0.3.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egowlp-8.5.5.9.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egorest-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egomgmt-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egoelastic-1.0.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egogpfsmonitor-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  ascd-2.1.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  conductorsparkcore-2.1.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  conductorsparkmgmt-2.1.0.0.rpm
Do you want to continue?(Y/N)y

Start install...
Preparing...                          ################################# [100%]

The installation will be processed using the following settings:
Workload Execution Mode (WEM): Advanced
Cluster Administrator: root
Cluster Name: cluster1
Installation Directory: /opt/ibm/spectrumcomputing
Connection Base Port: 17869

Updating / installing...
   1:egocore-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO core package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egojre-1.8.0.3-408454            ################################# [100%]

Successfully installed the IBM Java runtime environment package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egowlp-8.5.5.9-408454            ################################# [100%]

Successfully installed the IBM WebSphere Application Server Liberty Profile
package.
Preparing...                          ################################# [100%]
```

```
Updating / installing...
   1:egorest-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO RESTful API server package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egomgmt-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO management package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egoelastic-1.0.0.0-1             ################################# [100%]

Elastic package is successfully installed.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egogpfsmonitor-3.4.0.0-408454    ################################# [100%]

Successfully installed the monitoring package for IBM Spectrum Scale.
Preparing...                          ################################# [100%]

Updating / installing...
   1:ascd-2.1.0.0-408454              ################################# [100%]

Successfully installed IBM Spectrum Computing family: ascd package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:conductorsparkcore-2.1.0.0-408454################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark core
package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:conductorsparkmgmt-2.1.0.0-408454################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark
management package.
```

Source the environment

After the installation is complete, you need to source the environment. This procedure depends on the shell interpreter currently been used in the system; either BASH or CSH.

If BASH is running, run the following command to source the environment:

```
. $EGO_TOP/profile.platform
```

If CSH is running, run the following command to source the environment:

```
source $EGO_TOP/cshrc.platform
```

In this case, `$EGO_TOP` is the path to your installation directory, and `/opt/ibm/spectrumcomputing` is the default directory, as shown in Example 6-4.

Example 6-4 Source the environment

```
[root@st6a04-rh-sym1 conductor]# echo $0
-bash
[root@st6a04-rh-sym1 conductor]# echo $EGO_TOP
/opt/ibm/spectrumcomputing/
[root@st6a04-rh-sym1 conductor]# . /opt/ibm/spectrumcomputing/profile.platform
```

Tip: To avoid the need to source the environment every time you log in the system, add the source command to the `~/.bashrc` file.

Entitling IBM Conductor with Spark

Entitlement (licensing) is required only on the master host. To do so, complete the following steps:

1. Join the master host to the cluster using its FQDN
2. Set the entitlement, specifying the entitlement file path, as shown in Example 6-5.

Example 6-5 Conductor with Spark entitling

```
[root@st6a04-rh-sym1 conductor]# hostname -f
st6a04-rh-sym1.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 conductor]# egoconfig join st6a04-rh-sym1.pok.stglabs.ibm.com
You are about to create a new cluster with this host as the master host.  Do you
want to continue? [y/n]y
A new cluster <cluster1> has been created. The host
<st6a04-rh-sym1.pok.stglabs.ibm.com> is the master host.
[root@st6a04-rh-sym1 conductor]# egoconfig
setentitlementconductor_spark_entitlement.dat
To complete the installation, restart the cluster using the egosh ego start
command. (If you already started the cluster, stop all services and restart the
cluster.)
Then, log on to the management console as cluster administrator and run the
cluster configuration wizard when prompted. You must complete this step to ensure
correct cluster configurations for IBM Spectrum Symphony, IBM Spectrum Conductor
and IBM Spectrum Conductor with Spark.
The system successfully set the specified entitlement.
```

Start and logon

After the Conductor with Spark is entitled, complete the following steps:

1. Start the cluster, as shown in Example 6-6 on page 90.
2. Log on as cluster administrator using `Admin` as the user and password.

   After your are logged on, the cluster and the entitlement information can be verified.

Important: If your firewall is running, you must open the corresponding ports before start the cluster.

Example 6-6 Start the cluster

```
[root@st6a04-rh-sym1 conductor]# egosh ego start
Start up LIM on <st6a04-rh-sym1> ...... done
```

```
[root@st6a04-rh-sym1 conductor]# egosh user logon -u Admin -x Admin
Logged on successfully
[root@st6a04-rh-sym1 /]# egosh ego info
Cluster name            : cluster1
EGO master host name    : st6a04-rh-sym1.pok.stglabs.ibm.com
EGO master version      : 3.4
[root@st6a04-rh-sym1 /]#
[root@st6a04-rh-sym1 conductor]# egosh entitlement info
IBM Spectrum Conductor with Spark   : Entitled
EGO                                 : Entitled
```

Tip: At the `$EGO_TOP/kernel/conf/ego.conf` configuration file, you can verify the EGO daemons port numbers.

3. If a failure occurs when trying to start the cluster, review the `lim.log.`*`host_name`* and *`v`*`emkd.log.`*`host_name`* logs, both at the `$EGO_TOP/kernel/log/` directory. See Example 6-7.

Example 6-7 Cluster manager logs

```
[root@st6a04-rh-sym1 /]# cd $EGO_TOP/kernel/log
[root@st6a04-rh-sym1 log]# ls
elim.nvidia.log.st6a04-rh-sym1 melim.log.st6a04-rh-sym1
vemkd.log.st6a04-rh-sym1 elim.sa.log.st6a04-rh-sym1
pem.log.st6a04-rh-sym1 wsm.log.st6a04-rh-sym1 lim.log.st6a04-rh-sym1
pim.log.st6a04-rh-sym1
[root@st6a04-rh-sym1 log]#
```

4. You can verify that all of the EGO services are running with the **`service`** command, as shown in Example 6-8.

Example 6-8 V*erify the EGOservices*

```
[root@st6a04-rh-sym1 ~]# egosh service list
SERVICE  STATE    ALLOC CONSUMER RGROUP RESOURCE SLOTS SEQ_NO INST_STATE ACTI
GPFSmon* DEFINED        /GPFSmo* Manag*
elk-ela* STARTED  89    /Manage* Manag* st6a04-* 1     1      RUN        364
                                       st6a04-* 1     3      RUN        340
                                       st6a04-* 1     2      RUN        339
purger   STARTED  90    /Manage* Manag* st6a04-* 1     1      RUN        350
elk-kib* STARTED  99    /Manage* Manag* st6a04-* 1     1      RUN        351
elk-ind* STARTED  100   /Manage* Manag* st6a04-* 1     1      RUN        352
WEBGUI   STARTED  91    /Manage* Manag* st6a04-* 1     1      RUN        353
plc      STARTED  92    /Manage* Manag* st6a04-* 1     1      RUN        354
derbydb  DEFINED        /Manage* Manag*
elk-shi* STARTED  102   /Cluste* Inter* st6a04-* 1     6      RUN        363
                                       st6a04-* 1     3      RUN        359
                                       st6a04-* 1     4      RUN        360
                                       st6a04-* 1     5      RUN        361
                                       st6a04-* 1     1      RUN        357
                                       st6a04-* 1     2      RUN        358
Service* STARTED  93    /Manage* Manag* st6a04-* 1     1      RUN        355
ExecPro* STARTED  94    /Cluste* Inter* st6a04-* 1     1      RUN        362
                                       st6a04-* 1     4      RUN        336
                                       st6a04-* 1     5      RUN        345
                                       st6a04-* 1     6      RUN        346
```

```
                                        st6a04-* 1     2       RUN         344
                                        st6a04-* 1     3       RUN         335
ascd      STARTED  95    /Manage* Manag* st6a04-* 1     1       RUN         347
WebServ*  STARTED  98    /Manage* Manag* st6a04-* 1     1       RUN         356
REST      STARTED  96    /Manage* Manag* st6a04-* 1     1       RUN         348
EGOYARN   DEFINED        /YARN/R* Inter*
RS        STARTED  97    /Manage* Manag* st6a04-* 1     1       RUN         349
SPARKSS   DEFINED        /Cluste* Inter*
```

For easier cluster management, you can use the Conductor cluster management console, to access it. Use `Admin` as the username and password:

- With SSL enabled by default (`https://`*`host_name`*`:8443`)
- With SSL disabled (`http://`*`host_name`*`:8080`)

Figure 6-12 shows the cluster management console main page.

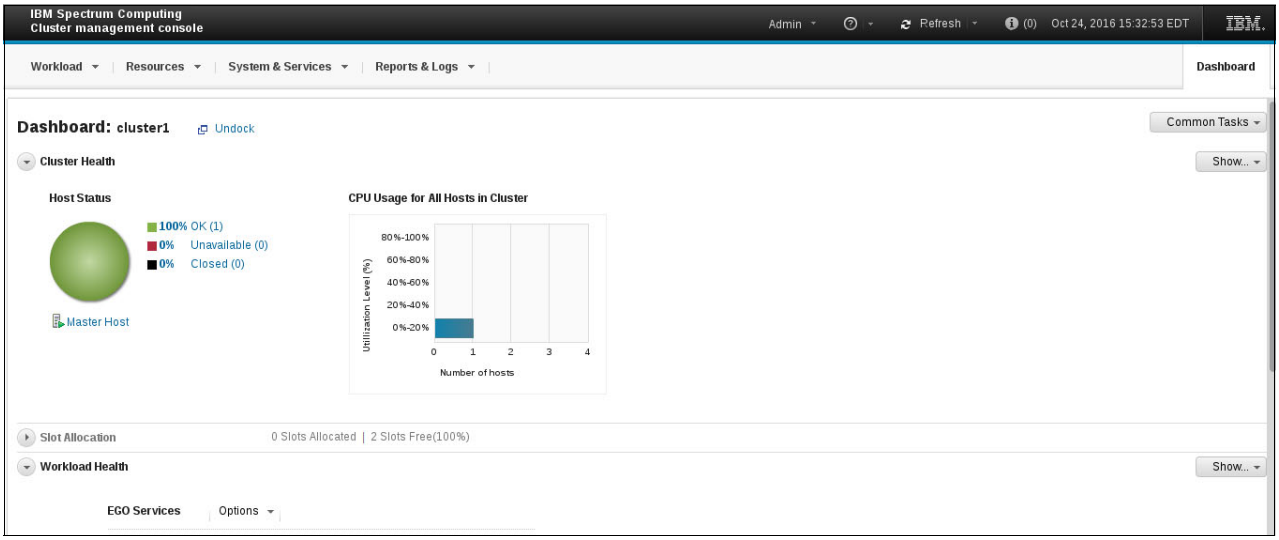


Figure 6-12 Cluster management console

Add a management node

Adding management nodes enables you to configure a host failover, so the system services fail over to a different management host if the master host fails. The installation steps are described as follows.

Stop the elk-elasticsearch

Before adding a management host to the cluster, stop the `elk-elasticsearch` service from the master host, as shown in Example 6-9.

Example 6-9 Stop elk-elasticsearch

```
[root@st6a04-rh-sym1 ~]# egosh service stop elk-elasticsearch
Service <elk-elasticsearch> has been stopped successfully
[root@st6a04-rh-sym1 ~]#
```

Install a management host

Install IBM Spectrum Conductor with Spark on the management host following the procedure discussed in “Install a management host” on page 87.

Note: The entitlement is not required on the new management host.

Join the node to the cluster

After the IBM Spectrum Conductor with Spark is installed, join the new management node to the cluster using the master host name, identified by its FQDN as shown in Example 6-10.

Example 6-10 Join a new management host

```
[root@st6a04-rh-sym2 conductor]# egoconfig join st6a04-rh-sym1.pok.stglabs.ibm.com
You are about to join this host to a cluster with master host
st6a04-rh-sym1.pok.stglabs.ibm.com.  Do you want to continue? [y/n]y
The host st6a04-rh-sym2 has joined the cluster cluster1.
```

Important: Source the environment before running the **`join`** command.

Start, log on, and verify

To start and verify, complete the following steps:

1. Start up the cluster and log on as cluster administrator (as shown in Example 6-11) using `Admin` as the user and password.

   Example 6-11 Start and log in the new management node

   ```
   [root@st6a04-rh-sym2 conductor]# egosh ego start
   Start up LIM on <st6a04-rh-sym2> ...... done
   [root@st6a04-rh-sym2 conductor]#egosh user logon -u Admin -x Admin
   Cluster name            : cluster1
   EGO master host name    : st6a04-rh-sym1
   EGO master version      : 3.4
   ```

   Important: If your firewall is running, you must open the corresponding ports before start the cluster.

2. After your are started and logged on, the join procedure can be verified, as shown in Example 6-12.

Example 6-12 Verify the join procedure

```
[root@st6a04-rh-sym2 conductor]# egosh resource list
NAME     status       mem    swp    tmp   ut    it    pg   r1m  r15s  r15m  ls
st6a04-* ok         2516M  4094M    29G   1%  2714     0     0   0.6   0.1   0
st6a04-* ok         2544M  4094M    29G   3%     1     0     0     0   0.1   1
```

Figure 6-13 shows the new management host added to the cluster.

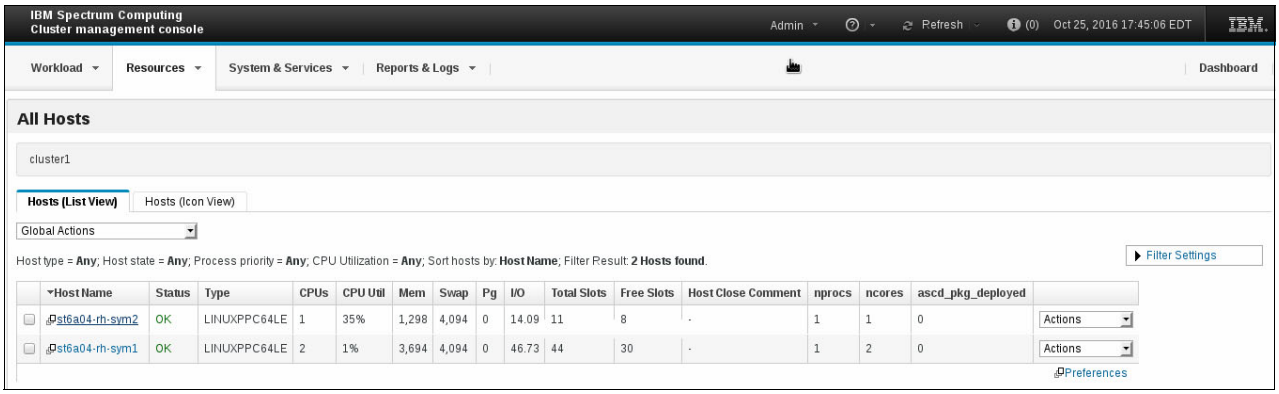


Figure 6-13 New management host in the cluster

Start the elk-elasticsearch service

When the management host is added, start the elk-elasticsearch service from the master host, as shown in Example 6-13.

Example 6-13 Start elk-elasticsearch service

```
root@st6a04-rh-sym1 ~]# egosh service start elk-elasticsearch
Starting service <elk-elasticsearch>. Run <service view> to ensure startup
[root@st6a04-rh-sym1 ~]# egosh service list | grep elk*
elk-ela* STARTED  81    /Manage* Manag* st6a04-* 1     1       RUN         294
elk-kib* STARTED  82    /Manage* Manag* st6a04-* 1     1       RUN         296
elk-ind* STARTED  83    /Manage* Manag* st6a04-* 1     1       RUN         297
elk-shi* STARTED  84    /Cluste* Inter* st6a04-* 1     3       RUN         300
```

Figure 6-14 shows the EGO services current running from the cluster management console.

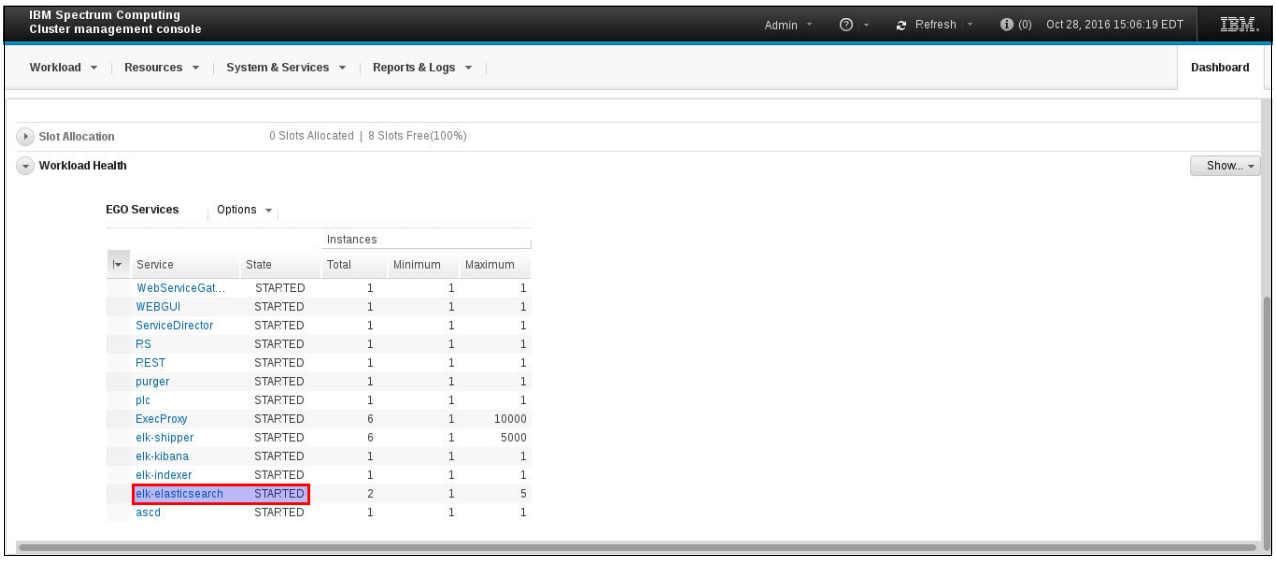


Figure 6-14 Verifying the elk-elasticsearch service

Setting up the master host failover

After installing your management hosts, set up the master host failover by configuring a shared file location and defining the order in which management hosts take over as the master if the master host fails. The management hosts that are designated to become the master when failover occurs are known as master candidate hosts.

When setting up failover for the master host, specify a shared location for configuration information and common files that management hosts look up. This shared location must always be available for failover purposes.

To set up the master host failover, use the following steps.

Shut down the EGO services

Before you begin, you must have joined the host to the cluster by running **egoconfig join master_host** in the management host you that want to be a master candidate:

1. Log on to the management hosts as the cluster administrator.
2. Stop all the services, as shown in Example 6-14.
3. Shut down the cluster as shown in Example 6-15.

Example 6-14 Stop ego services

```
[root@st6a04-rh-sym2 ~]# egosh user logon -u Admin -x Admin
Logged on successfully
[root@st6a04-rh-sym2 ~]# egosh service stop all
<all> will be considered as the option to stop all the services,
and all the existing services will be stopped
Service <GPFSmonitor> has been stopped successfully
Service <elk-shipper> has been stopped successfully
Service <elk-kibana> has been stopped successfully
Service <elk-indexer> has been stopped successfully
Service <elk-elasticsearch> has been stopped successfully
Service <purger> has been stopped successfully
Service <plc> has been stopped successfully
Service <derbydb> has been stopped successfully
Service <WEBGUI> has been stopped successfully
Service <ascd> has been stopped successfully
Service <SPARKSS> has been stopped successfully
Service <EGOYARN> has been stopped successfully
Service <REST> has been stopped successfully
Service <ExecProxy> has been stopped successfully
Service <ServiceDirector> has been stopped successfully
Service <RS> has been stopped successfully
Service <WebServiceGateway> has been stopped successfully
```

Example 6-15 Shut down the cluster

```
[root@st6a04-rh-sym2 ~]# egosh ego shutdown all
Do you really want to shut down LIMs on all hosts? [y/n] y
Shut down LIM on <st6a04-rh-sym2> ..... done
Shut down LIM on <st6a04-rh-sym1> ..... done
[root@st6a04-rh-sym2 ~]#
```

Specify a shared directory

Specify a shared location for configuration information and common files that management hosts look up. This shared location must always be available for failover purposes.

If you set the `SHARED_FS_INSTALL=Y` environment variable to install your cluster to a shared file system, specify a directory on the shared file system, as shown in Example 6-16.

When you run the **`egoconfig mghost shared_directory`** command on a host, the host is added to the `ManagementHosts` resource group.

Example 6-16 Specify the shared directory

```
[root@st6a04-rh-sym2 ~]# egoconfig mghost /gpfs
This host will use configuration files on a shared directory. Make sure that /gpfs
is a shared directory. Do you want to continue? [y/n]y
Warning: stop all cluster services managed by EGO before you run egoconfig. Do you
want to continue? [y/n]y
Warning: you must stop the cluster before you run egoconfig. Do you want to
continue? [y/n]y
The shared configuration directory is /gpfs/kernel/conf. You must reset your
environment before you can run any more EGO commands. Source the environment
/opt/ibm/spectrumcomputing/cshrc.platform or
/opt/ibm/spectrumcomputing/profile.platform again.
[root@st6a04-rh-sym2 ~]#
```

Start the cluster

Reset your environment on this session to use the shared directory and start the cluster, as shown in Example 6-17.

Example 6-17 Start the cluster

```
[root@st6a04-rh-sym2 ~]# . $EGO_TOP/profile.platform
[root@st6a04-rh-sym2 ~]#
[root@st6a04-rh-sym2 ~]# egosh ego start all
Do you really want to start up LIM on all hosts ? [y/n]y
Start up LIM on <st6a04-rh-sym2> ...... done
Start up LIM on <st6a04-rh-sym1> ...... done
```

Configure the order of failover hosts

After specifying a shared directory for management hosts, set the order in which management hosts take over as the master when the master host fails. By default, the list includes just one host: The master host.

When you specify additional candidates to enable failover, the master host is the first host in the list. When the master host becomes unavailable, the next host becomes the master, and so on down the list.

To configure the failover hosts order, complete the following steps:

1. Log on to the cluster management console
2. Select **System & Services** → **Cluster** → **Master and Failover** (Figure 6-15 on page 97).
3. In the available hosts list, select the management hosts that are not already configured as master candidates and click **Add** (Figure 6-16 on page 97).
4. After the management host is added as master candidate, click **Apply**.

If you installed to a shared file system, you need to synchronize the host list between the local `ego.conf` file and the shared `ego.conf` file.

5. On each of your compute hosts, edit the `ego.conf` configuration file at `$EGO_TOP/kernel/conf/`. Set the **`EGO_GET_CONF`** parameter to `LIM` and save your changes, as shown in Example 6-18.

Example 6-18 EGO_GET_CONF value

```
[root@st6a04-rh-sym3 ~]# cat $EGO_TOP/kernel/conf/ego.conf | grep EGO_GET_CONF
EGO_GET_CONF=LIM
[root@st6a04-rh-sym3 ~]#
```

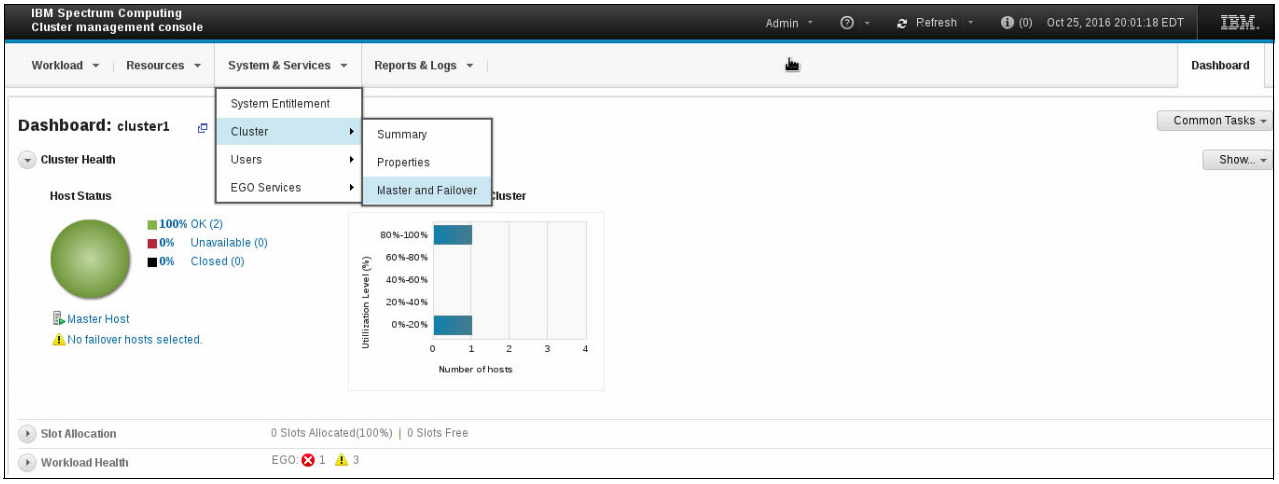


Figure 6-15 Master and failover tab

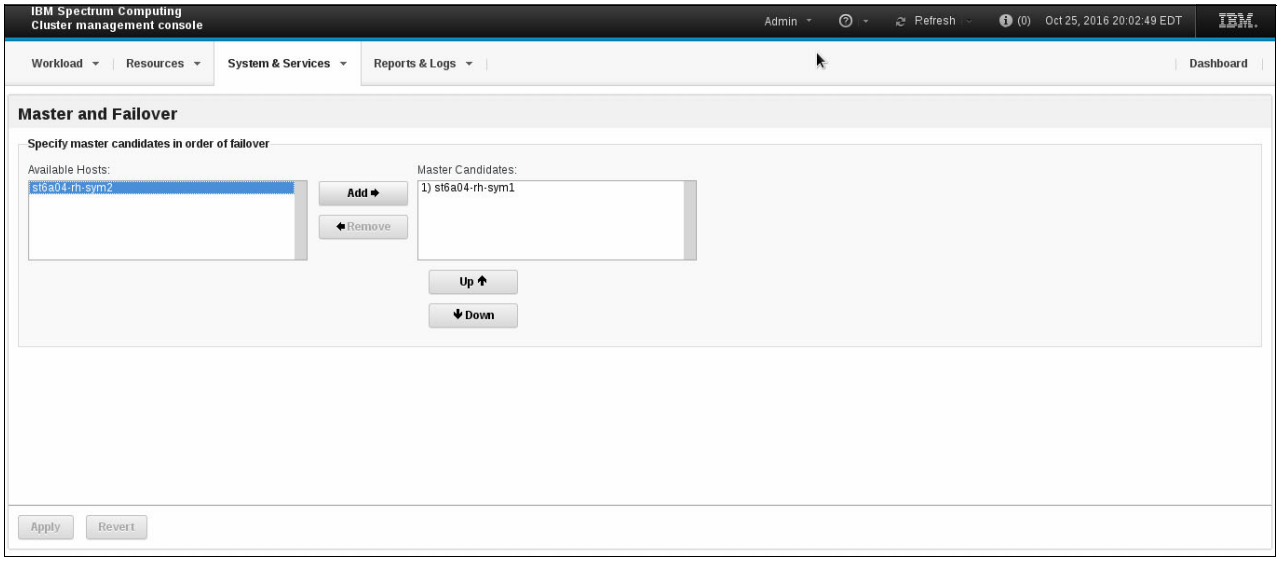


Figure 6-16 Add the master candidate

Tip: Use *Shift* or *Ctrl* to select more than one host at a time.

6. To change the failover order, use the *Up* and *Down* buttons. The master host is at the top of the list. If the master becomes unavailable, the host that takes over is next. If those two hosts are unavailable, the host that takes over is third.
7. You can also remove any hosts from the candidate list by selecting the host name and clicking **Remove**. The master host (which is the management host at the top of the list by default) cannot be removed. If you want to remove that host, you must move it down in the list so that it is not the master, and restart your cluster.

Figure 6-17 shows the host failover configuration in the cluster.

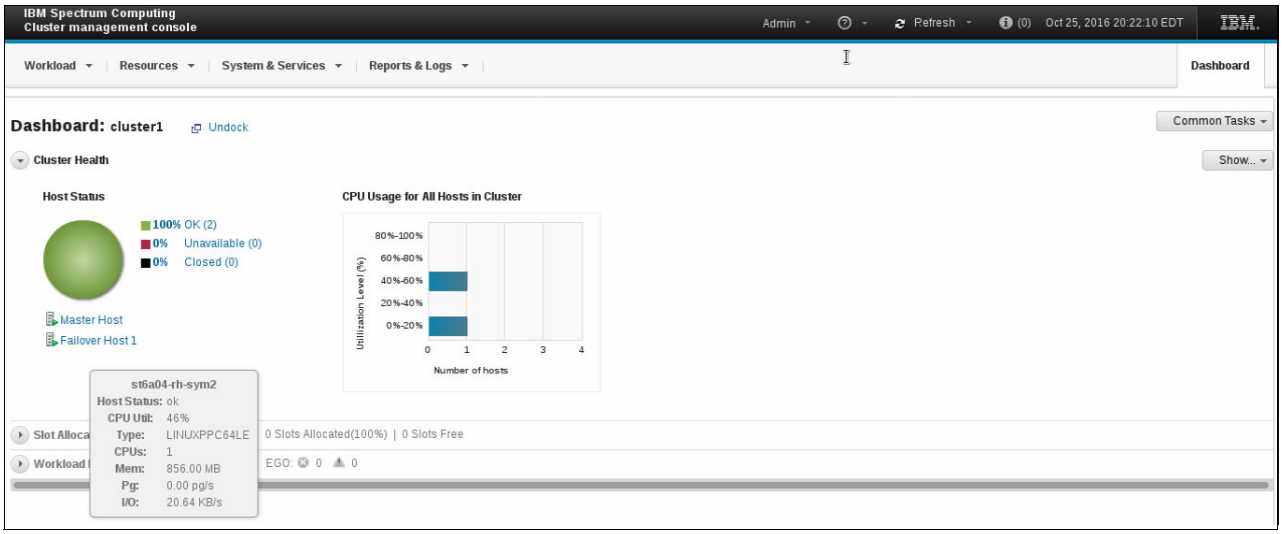


Figure 6-17 Cluster host failover

Install a compute host

The compute host provides computing resources to consumers in the EGO cluster. Before you install the IBM Conductor with Spark on a compute host, the cluster properties settings must be defined (see Table 6-7 on page 85). When done, complete the four steps as follows.

Important: The `EGOCOMPUTEHOST` environmental variable must be set to **"Y"** before running the installation. See Example 6-2 on page 86.

Run the IBM Spectrum Conductor with Spark installation file

Navigate to the directory where you uploaded the IBM Spectrum Conductor with Spark binary file and run it. See Example 6-19.

Example 6-19 IBM Spectrum Conductor with Spark installation

```
[root@st6a04-rh-sym3 conductor]# ./conductorspark2.1.0.0_ppc64le.bin
Extracting files... done.
Installation for IBM Spectrum Conductor with Spark 2.1.0

IBM Spectrum Conductor with Spark License Agreement
International Program License Agreement

Part 1 - General Terms
BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, CLICKING ON
AN "ACCEPT" BUTTON, OR OTHERWISE USING THE PROGRAM,
LICENSEE AGREES TO THE TERMS OF THIS AGREEMENT. IF YOU ARE
ACCEPTING THESE TERMS ON BEHALF OF LICENSEE, YOU REPRESENT
```

```
AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND LICENSEE
TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

* DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, CLICK ON AN
"ACCEPT" BUTTON, OR USE THE PROGRAM; AND

* PROMPTLY RETURN THE UNUSED MEDIA, DOCUMENTATION, AND

Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
1
The following commands will install the .rpm packages on the host.
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egocore-3.4.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egojre-1.8.0.3.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egoelastic-1.0.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egowlp-8.5.5.9.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egorest-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egomgmt-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egoelastic-1.0.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egogpfsmonitor-3.4.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  ascd-2.1.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  conductorsparkcore-2.1.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  conductorsparkmgmt-2.1.0.0.rpm
Do you want to continue?(Y/N)y
Start install...
Preparing...                          ################################# [100%]

The installation will be processed using the following settings:
Workload Execution Mode (WEM): Advanced
Cluster Administrator: root
Cluster Name: cluster1
Installation Directory: /opt/ibm/spectrumcomputing
Connection Base Port: 17869

Updating / installing...
   1:egocore-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO core package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egojre-1.8.0.3-408454            ################################# [100%]
Updating / installing...
   1:egowlp-8.5.5.9-408454            ################################# [100%]

Successfully installed the IBM WebSphere Application Server Liberty Profile
package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egorest-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO RESTful API server package.
```

```
Preparing...                          ################################# [100%]

Updating / installing...
   1:egomgmt-3.4.0.0-408454           ################################# [100%]

Successfully installed the EGO management package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egoelastic-1.0.0.0-1             ################################# [100%]

Elastic package is successfully installed.
Preparing...                          ################################# [100%]

Updating / installing...
   1:egogpfsmonitor-3.4.0.0-408454    ################################# [100%]

Successfully installed the monitoring package for IBM Spectrum Scale.
Preparing...                          ################################# [100%]

Updating / installing...
   1:ascd-2.1.0.0-408454              ################################# [100%]

Successfully installed IBM Spectrum Computing family: ascd package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:conductorsparkcore-2.1.0.0-408454################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark core
package.
Preparing...                          ################################# [100%]

Updating / installing...
   1:conductorsparkmgmt-2.1.0.0-408454################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark
management package.
```

Source the environment

After the installation is completed, you need to source the environment. This procedure depends on the shell interpreter currently been used in the system; either BASH or CSH.

If BASH is running, run the following command to source the environment:

```
. $EGO_TOP/profile.platform
```

If CSH is running, run the following command to source the environment:

```
source $EGO_TOP/cshrc.platform
```

In this case, `$EGO_TOP` is the path to your installation directory and `/opt/ibm/spectrumcomputing` is the default directory.

Join the host to the cluster

Join the new compute node to the cluster using the master host name, identified by its FQDN, as shown in Example 6-20.

Example 6-20 Join the compute node

```
[root@st6a04-rh-sym3 conductor]# egoconfig join st6a04-rh-sym1.pok.stglabs.ibm.com
You are about to join this host to a cluster with master host
st6a04-rh-sym1.pok.stglabs.ibm.com.  Do you want to continue? [y/n]y
The host st6a04-rh-sym3 has joined the cluster cluster1.
[root@st6a04-rh-sym3 conductor]#
```

Start, log in, and verify

After the new compute node is successfully joined to the cluster, complete the following steps:

1. Start the EGO services.
2. Log on with `Admin` as the username and password, as shown in Example 6-21.
3. Verify that the new compute node is part of the cluster as shown in Example 6-22.

Example 6-21 Start EGO services and log on

```
[root@st6a04-rh-sym3 conductor]# egosh ego start
Start up LIM on <st6a04-rh-sym3> ...... done
[root@st6a04-rh-sym3 conductor]# egosh user logon -u Admin -x Admin
Logged on successfully
[root@st6a04-rh-sym3 conductor]#
```

Example 6-22 Verify the new compute node

```
[root@st6a04-rh-sym3 conductor]# egosh ego info
Cluster name            : cluster1
EGO master host name    : st6a04-rh-sym1
EGO master version      : 3.4
[root@st6a04-rh-sym3 conductor]# egosh resource list
NAME     status       mem    swp    tmp   ut    it    pg   r1m  r15s  r15m  ls
st6a04-* ok         1701M  4094M    29G   1%     7     0   0.3   0.1   0.3   0
st6a04-* ok          263M  4086M    32G  40%   908     0   3.4   2.5   3.4   1
st6a04-* ok         1412M  4094M    38G   5%     0     0     0   0.2   0.1   1
st6a04-* ok         1416M  4094M    38G   5%     0     0     0   0.3   0.1   1
[root@st6a04-rh-sym3 conductor]#
```

7

Cluster management with IBM Spectrum Cluster Foundation

This chapter describes the cluster management challenges and how you can overcome them with a simple and intuitive interface.

For a better understanding, this chapter is divided into the following topics:

- High-performance computing and big data cluster management
- IBM Spectrum Cluster Foundation
- IBM Spectrum Cluster Foundation Community Edition

7.1 High-performance computing and big data cluster management

The new information technology paradigm is based on scale-out environments. The massive amount of data and calculations that need to be done that are not price/performance friendly on a scale up system since it requires real large machine to run the workload. Flexible clusters are the base of high-performance computing for a long time and deliver the level of performance that is required for computational applications. The data-intensive applications that are now in vogue, are using the same concept to make data analysis possible at commercial scale referred in the industry as *big data*.

At the same time, that this approach to the compute and data challenge problems and hardware advancements, such as new processors and interconnect technologies, help addressing the feasibility of the projects, they have made clustered environments more complex and difficult to manage.

With growing complexity of the compute environments and the growth of multiple clusters aligned to business needs, today's clusters demand an easy-to-use cluster management tool.

The response from IBM Spectrum Computing Family to these challenges that arose from this scale-out approach to computational and data analysis is IBM Spectrum Cluster Foundation. It is the offering that automates the self-service assembly of multiple heterogeneous high-performance computing (HPC) and analytics clusters (big data) on shared compute infrastructure.

IBM Spectrum Cluster Foundation creates a secure multi-tenant analytics and HPC cloud for users running technical computing and analytics workloads to dynamically create clusters, grids, and HPC clouds on demand, consolidate a scattered cluster infrastructure, increase hardware utilization, gain access to larger cluster infrastructures, and rapidly deploy multiple heterogeneous HPC environments.

7.2 IBM Spectrum Cluster Foundation

IBM Spectrum Cluster Foundation is easy-to-use, powerful cluster management software for technical computing users. It delivers a comprehensive set of functions to help manage hardware and software from the infrastructure level. It automates the deployment of the operating system and software components, and complex activities, such as application cluster creation and maintenance of a system.

With a centralized user interface where system administrators can manage a complex cluster as a single system, it offers the flexibility for users to add customized features that are based on specific requirements of their environment. It also provides script templates for easy software deployment, and can set up and enable a multi-tenant, multi-cluster environment.

IBM Spectrum Cluster Foundation manages the provisioning of multiple multi-tenant analytics and technical computing clusters in a self-service and flexible fashion. It provides secure multi-tenancy with access controls, and resource limits to enable sharing. Based on assigned user roles, it provides rapid self-service provisioning of heterogeneous high-performance computing environments and gets the clusters that you need.

A community edition of IBM Spectrum Cluster Foundation is offered. The community edition is a complementary offering that has a limitation where only one cluster can be deployed. This edition is described in 7.3, “IBM Spectrum Cluster Foundation Community Edition” on page 123.

7.2.1 Supported software and Hardware requirements

All the information at this Chapter is regarding IBM Spectrum Cluster Foundation Version 4.2.2 that is the current available.

Hardware Requirements

Before you can install IBM Spectrum Cluster Foundation Version, verify that your environment meets the minimum hardware requirements for the management nodes and compute nodes. Verify the requirements at Table 7-1.

Table 7-1 Minimum Hardware Requirements

Requirement	Management node	Stateful compute	Stateless compute
Disk	100 GB	40 GB	-
Memory	4 GB	1 GB	8 GB for x86 and PowerBE compute nodes or 32 GB for PowerLE compute nodes
Network	1 Ethernet	1 Ethernet	1 Ethernet

Tested Hardware

IBM Spectrum Cluster Foundation Version 4.2.2 is tested on the following hardware:

- Lenovo System x3550 M4
- Lenovo System x3650 M4
- Lenovo System x3750 M4
- Lenovo System x3850 X5
- IBM Flex System® x220
- IBM Flex System x240
- IBM System x iDataPlex dx360 M4
- IBM NeXtScale nx360 M4
- IBM NeXtScale nx360 M5
- IBM Power 8
- IBM Power Systems S812LC
- IBM Power Systems S822LC

Note: For other hardware not listed under the tested hardware, including generic IPMI-based x86-64 hardware and Power hardware, contact IBM Support.

Supported software for management nodes

The supported operating systems for the management nodes can be seen in Table 7-2.

Table 7-2 Operating system compatibility for each supported hardware

x86	PowerNV LE	PowerKVM VM LE	Power Big Endian
Red Hat Enterprise Linux (RHEL) 7.2	-	RHEL 7.2	RHEL 7.2
RHEL 6.8	-		-
SUSE Linux Enterprise Server 12	-		-
SUSE Linux ES 11.3	-		-
-	-	Ubuntu 14.04.04	-

Supported software for compute nodes

The same installed software compatibility for management nodes is supported on the compute nodes.

Note: For cross-platform or cross-operating systems installations, supported features and templates are ready for use.

7.2.2 Quick and easy installation

IBM Spectrum Cluster Foundation installation is quick and easy. The interactive installer prompts for your management settings, and the installation sequence mounts the IBM Spectrum Cluster Foundation installation ISO and runs the installation. The installation copies, sets up, and installs the operating system and all dependent packages. It sets up the PostgreSQL database and completes the postinstallation, which includes configuring services. For more installation settings, a custom installation is also available.

Before installation

Before the installation, complete the following steps:

1. Check 7.2.1, “Supported software and Hardware requirements” on page 105.
2. Obtain a copy of the operating system.
3. Decide on a partitioning layout. The suggested partitioning layout is as follows:
   a. Ensure that the `/opt` partition has a least 4 GB
   b. Ensure that the `/var` partition has at least 40 GB
   c. Ensure that the `/install` partition has at least 40 GB
4. Configure at least one static network interface.
5. Use a fully qualified domain name (FQDN) for the management node.
6. The `/home` directory must be writable.
7. Install and configure the operating system.
8. Ensure that the `net-snmp-perl` package is installed on the management node. If not, you must install it manually.
9. Ensure that the NetworkManager service is stopped.

To stop the NetworkManager service, run the following command:

```
#systemctl stop NetworkManager
#systemctl disable NetworkManager
```

10. Ensure that SELinux is disabled. To disable SELinux, do the following:
    a. Edit the `/etc/selinux/config` file to set `SELINUX=disabled`.
    b. Reboot the management node.
11. Disable firewalld (installed by default in RHEL 7):

```
#systemctl firewalld stop
#systemctl disable firewalld
```

The following steps were tested on Cluster Foundation 4.2.2 and RHEL 7.2.

The installer contains the files and directories shown in Figure 7-1.

```
# ls -la
total 28
drwxr-xr-x. 9 root root 4096 Oct 19 11:13 .
drwxr-xr-x. 4 root root 4096 Oct 25 15:03 ..
drwxr-xr-x. 2 root root 4096 Jul 10 23:13 docs
drwxr-xr-x. 2 root root   65 Jul 10 23:13 ego
drwxr-xr-x. 2 root root   56 Jul 10 23:13 entitlement
drwxr-xr-x. 2 root root 4096 Jul 10 23:13 EULA
drwxr-xr-x. 5 root root   96 Jul 10 23:13 install
-r-xr-xr-x. 1 root root  281 Jul 10 23:13 install.desktop
-r-xr-xr-x. 1 root root   70 Jul 10 23:13 install.sh
drwxr-xr-x. 2 root root    6 Jul 10 23:13 kits
drwxr-xr-x. 7 root root  103 Jul 10 23:13 packages
-r--r--r--. 1 root root 1974 Jul 10 23:13 TRANS.TBL
```

Figure 7-1 ISO file content for Cluster Foundation 4.2.2

Start the installation

The Cluster Foundation installation process consist of four steps, which are described in the following sections. To install Cluster Foundation, run the `install.sh` file that is located on the root directory of the installation ISO, as shown in Example 7-1.

Example 7-1 Start the installation process

```
#./install.sh

  ================================================================
  Welcome to the IBM Spectrum Cluster Foundation 4.2.2 Installation
  ================================================================

  The complete IBM Spectrum Cluster Foundation 4.2.2 installation includes the
following:
  1. License Agreement
  2. Management node pre-checking
  3. Specify installation settings
  4. Installation

  Press ENTER to continue the installation or CTRL-C to quit the installation.
```

Accept license agreement

Example 7-2 shows the step to accept the license agreement.

Example 7-2 Accept license Agreement

```
 ===============================================================
  Step 1 of 4: License Agreement
  ===============================================================

  International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, CLICKING ON
AN "ACCEPT" BUTTON, OR OTHERWISE USING THE PROGRAM,
LICENSEE AGREES TO THE TERMS OF THIS AGREEMENT. IF YOU ARE
ACCEPTING THESE TERMS ON BEHALF OF LICENSEE, YOU REPRESENT
AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND LICENSEE
TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

* DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, CLICK ON AN
"ACCEPT" BUTTON, OR USE THE PROGRAM; AND

* PROMPTLY RETURN THE UNUSED MEDIA, DOCUMENTATION, AND

Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
```

Prerequisite check

In this step, the installation program checks if the prerequisites are met, as shown in Example 7-3.

The first warning in Example 7-3 refers to the firewalld service.

By default, RHEL 7.x uses firewalld. The IBM Spectrum Cluster Foundation installation program disables firewalld and uses the iptables service. If any firewall rules are needed for security reasons, those rules can be re-created in iptables after the installation of IBM Spectrum Cluster Foundation.

The last warning is because of the disk partitioning: The `/install` and `/var` directories are in the same file system, which is no problem for a testing environment.

Example 7-3 Step 2 node prerequisite checking

```
 ===============================================================
  Step 2 of 4: Management node pre-checking
  ===============================================================

  Checking hardware architecture...                      [  OK  ]
  Checking OS compatibility...                           [  OK  ]
  Checking free memory...                                [  OK  ]
  Checking firewall...                                   [WARNING]
```

```
  WARNING: By default, RHEL 7 and higher use firewalld to manage firewall
settings. IBM Spectrum Cluster Foundation will disable firewalld and use the
iptables service ins
tead to manage firewall settings. If needed, make sure to take the necessary steps
to reconfigure any firewall settings after installation.
  Do you want to continue? (Y/N) [N]: Y
  Checking if SELinux is disabled...                        [  OK  ]
  Checking if Auto Update is disabled...                    [  OK  ]
  Checking if NetworkManager is disabled...                 [  OK  ]
  Checking if Web Services are disabled...                  [  OK  ]
  Checking if PostgreSQL is disabled...                     [  OK  ]
  Checking if chronyd is disabled...                        [  OK  ]
  Checking for DNS service...                               [  OK  ]
  Checking for DHCP service...                              [  OK  ]
  Checking for available ports...                           [  OK  ]
  Checking management node name...                          [  OK  ]
  Checking static NIC...                                    [  OK  ]
  Probing DNS settings...                                   [  OK  ]
  Probing language and locale settings...                   [  OK  ]
  Checking home directory (/home) ...                       [  OK  ]
  Checking mount point for depot (/install) directory...    [  OK  ]
  Checking required free disk space for opt directory...    [  OK  ]
  Checking required free disk space for var directory...    [WARNING]
  WARNING: Both the install and var directory are located on the same partition.
  Make sure that the partition has at least 80 GB of free disk space.
```

Choose the installation method

This step is for choosing the preferred method of installation.

IBM Spectrum Cluster Foundation can be installed using an interactive installer in one of two methods: The quick installation method and the custom installation method. The quick installation method sets up basic options with default values. The custom installation method provides more installation options and enables the administrator to specify additional system configurations, as shown in Example 7-4.

This example shows the Custom installation Method “option 2” and the default options for the installation process except for the Network and other requirements like the iso image file used to provision the VMs.

Example 7-4 Installation method

```
 ===============================================================
  Step 3 of 4: Specify installation settings
  ===============================================================


  Select the installation method from the following options:
       1) Quick Installation
       2) Custom Installation
  Enter your selection [1]:  2

  Select a mount point for the depot (/install) directory from the following
options:
       1) Mount point: '/' Free space: '39 GB'
  Enter your selection [1]:
```

```
  The OS version must be the same as the OS version on the management node.
  From the following options, select where to install the OS from:
       1) CD/DVD drive
       2) ISO image or mount point
  Enter your selection [1]:  2
  Enter the path to the first ISO image or mount point:
 /home/pkgs/rhel-server-7.2-ppc64le-dvd.iso

  Select a network interface for the provisioning network from the following
options:
       1) Interface: eth0, IP: 9.47.76.96, Netmask: 255.255.240.0
       2) Interface: eth1, IP: 10.0.0.11, Netmask: 255.255.255.0
  Enter your selection [1]:  2

  Enter IP address range used for provisioning compute nodes
[10.0.0.3-10.0.0.200]: 10.0.0.20-10.0.0.200

  Do you want to provision compute nodes with node discovery? (Y/N) [Y]:

  Enter a temporary IP address range to be used for provisioning compute nodes
  by node discovery. This range cannot overlap the range specified for the
  provisioning compute nodes. [10.0.0.201-10.0.0.254]:

  The management node is connected to the public network by:
       1) Interface: eth0, IP: 9.47.76.96, Netmask: 255.255.240.0
       2) It is not connected to the public network
  Enter your selection [1]:
  Enable IBM Spectrum Cluster Foundation specific rules for the management node
firewall to the public interface? (Y/N) [Y]:

  Enable NAT forwarding on the management node for all compute nodes? (Y/N) [Y]:

  Enable a FSP network that uses the default provisioning template (Y/N) [N]: N


  Enter a domain name for the provisioning network [private.dns.zone]:
 itso.dns.zone

  Set a fully qualified domain name as the host name for managed compute nodes?
(Y/N) [N]:

  Set a domain name for the public network (Y/N) [Y]:

  Enter a domain name for the public network [example.com]:  pok.stglabs.ibm.com

  Enter the IP addresses of extra name servers that are separated by commas
[9.12.16.2,9.0.128.50]:

  Enter NTP server [pool.ntp.org]:

  Synchronizing management node with the time server...   [WARNING]
  WARNING: Synchronization unsuccessful with the time server 'pool.ntp.org'.

  Do you want to export the home directory on the management node
```

```
  and use it for all compute nodes? (Y/N) [Y]:

  Do you want to change the root password for compute nodes and the
  default password for the IBM Spectrum Cluster Foundation database? (Y/N) [Y]:
  Enter the IBM Spectrum Cluster Foundation database administrator password
[pcmdbadm]:
  Enter the password again:
  Enter the root account password for all compute nodes [Cluster]:
  Enter the password again:

  ================================================================
   IBM Spectrum Cluster Foundation Installation Summary
  ================================================================

  You have selected the following installation settings:

  Provision network domain:                itso.dns.zone
  Provision network interface:             eth1, 10.0.0.0/255.255.255.0
  Public network domain:                   pok.stglabs.ibm.com
  Public network interface:                eth0, 9.47.64.0/255.255.240.0
  Enable FQDN host name:                   No
  Depot (/install) directory mount point:  /
  OS media:
                              /home/pkgs/rhel-server-7.2-ppc64le-dvd.iso
  Disable OS check:                        No
  Network Interface:                       eth1
  eth1 IP address range for baremetal compute nodes:10.0.0.20-10.0.0.200
  eth1 IP address range for node discovery:10.0.0.201-10.0.0.254
  Enable firewall:                         Yes
  Enable NAT forwarding:                   Yes
  NTP server:                              pool.ntp.org
  Name servers:                            9.12.16.2,9.0.128.50
  Database administrator password:         ************
  Compute node root password:              ************
  Export home directory:                   Yes
  Enable Default BMC network with:         Public Network
  Default BMC Hardware Profile:            IBM_PowerNV

  ================================================================

  Note: To copy the OS from the OS DVD, you must insert the first
  OS DVD into the DVD drive before begining the installation.

  To modify any of the above settings, press "99" to go back
  to "Step 3: Specify installation settings", or press "1"
  to begin the installation.
  1
```

Installation

After the three previous steps are completed, the install process starts after the “1” option is selected, or there is a last choice to change something by pressing “99” to go back. The installation process output is shown in Example 7-5. All information about the installation process is logged to the `/opt/pcm/log/pcm-installer.log` file for future reference.

Example 7-5 Last step, installation Step

```
================================================================
  Step 4 of 4: Installation
  ================================================================

  Copying IBM Spectrum Cluster Foundation core packages...    [  OK  ]

  Copying IBM Spectrum Cluster Foundation kits...             [  OK  ]

  Adding OS from media '/home/pkgs/rhel-server-7.2-ppc64le-dvd.iso'...
  * Verifying that the OS distribution, architecture,
    and version are supported...                           [  OK  ]

  * Detected OS: [rhel 7 ppc64le]                          [  OK  ]
  * Copying OS media. This can take a few minutes...       [  OK  ]
  Successfully added operating system.

  Preparing the bootstrap environment...                   [  OK  ]

  Installing packages. This can take several minutes...    [  OK  ]
  Initializing IBM Spectrum Cluster Foundation configuration. [  OK  ]
  Installing kits. This can take several minutes...        [  OK  ]

  Running the pcmconfig script to complete the installation.
  Setting up hosts/resolv files:                           [  OK  ]
  Setting up firewall:                                     [  OK  ]
  Setting up IBM Spectrum Cluster Foundation https:           [  OK  ]
  Setting up dhcpd:                                        [  OK  ]
  Setting up named:                                        [  OK  ]
  Setting up shared NFS export:                            [  OK  ]
  Setting up ntpd:                                         [  OK  ]
  Setting up IBM Spectrum Cluster Foundation services:        [  OK  ]
  Setting up message of the day:                           [  OK  ]
  Setting up IBM Spectrum Cluster Foundation documentation:   [  OK  ]
  Setting up IBM Spectrum Cluster Foundation configuration:   [  OK  ]
  Updating management node:                                [  OK  ]
  Build stateless image:                                   [  OK  ]
  Setting up cross arch provisioning:                      [  OK  ]

  The IBM Spectrum Cluster Foundation installation is complete.
  Installation log can be found here: /opt/pcm/log/pcm-installer.log.

  Run the 'source /opt/pcm/bin/pcmenv.sh' command to configure environment
variables for this session. This is not required for new login sessions.

  To get started with IBM Spectrum Cluster Foundation 4.2.2, using your web
browser, you can access the Web Portal at https://9.47.76.96:8443 with the 'root'
user on the management node.
```

Verifying the installation and accessing the Web GUI

To verify that the installation is working correctly, log in to the management node as a root user and complete the following tasks:

1. Source IBM Spectrum Cluster Foundation environment variables:

```
# . /opt/pcm/bin/pcmenv.sh
```

2. Check that the PostgreSQL database server is running:

```
# service postgresql status
Redirecting to /bin/systemctl status  postgresql.service
? postgresql.service - PostgreSQL database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; vendor
preset: disabled)
   Active: active (running) since Wed 2016-10-19 15:20:17 EDT; 6 days ago
 Main PID: 22453 (postgres)
   CGroup: /system.slice/postgresql.service
         ·· 4003 postgres: xcatadm xcatdb 10.0.0.12(60739) idle
         ··16499 postgres: xcatadm xcatdb 10.0.0.12(59996) idle
         ....... << output lines omitted >>
         ··19085 postgres: xcatadm xcatdb 10.0.0.12(37257) idle
         ··22453 /usr/bin/postgres -D /var/lib/pgsql/data -p 5432
         ··22454 postgres: logger process
         ··22456 postgres: checkpointer process
         ··22457 postgres: writer process
         ··22458 postgres: wal writer process
         ··22459 postgres: autovacuum launcher process
         ··22460 postgres: stats collector process
```

3. Check that the IBM Spectrum Cluster Foundation services are running:

```
#service xcatd status
sxcatd service is running

# service pcm status
Cluster name            : PCM
EGO master host name    : st6a04-rh72-02
EGO master version      : 3.4
SERVICE  STATE    ALLOC CONSUMER RGROUP RESOURCE SLOTS SEQ_NO INST_STATE ACTI
PCM-PUR* STARTED  7     /Manage* Manag* st6a04-* 1     1      RUN        6
PTC      STARTED  8     /Manage* Manag* st6a04-* 1     1      RUN        7
PCM-PLC  STARTED  9     /Manage* Manag* st6a04-* 1     1      RUN        8
PCM-WEB  STARTED  12    /Manage* Manag* st6a04-* 1     1      RUN        11
elk-shi* STARTED  6     /Cluste* Inter* st6a04-* 1     1      RUN        5
elk-kib* STARTED  3     /Manage* Manag* st6a04-* 1     1      RUN        2
elk-ind* STARTED  4     /Manage* Manag* st6a04-* 1     1      RUN        3
elk-ela* STARTED  2     /Manage* Manag* st6a04-* 1     1      RUN        1
RULE-EN* STARTED  10    /Manage* Manag* st6a04-* 1     1      RUN        9
PCMD     STARTED  11    /Manage* Manag* st6a04-* 1     1      RUN        10
ACTIVEMQ STARTED  5     /Manage* Manag* st6a04-* 1     1      RUN        4
```

4. Log in to the Web Portal.

5. Open a supported web browser. Table 7-3 shows the supported browsers for IBM Cluster Foundation version 4.2.2.

   Go to https://<*mgtnode-IP:8443*>, where <*mgtnode-IP*> is the real management node IP address. If you are connected to a public network, you can also navigate to https://<*mgtnode-hostname*>:8443, where <*mgtnode-hostname*> is the real management node hostname.

Table 7-3 Supported browsers

Browser	Version
Internet Explorer on Windows	Internet Explorer 11 Internet Explorer 10
Firefox on Windows	Firefox ESR 38 Firefox ESR 25
Firefox on RHEL	Firefox 38 (firefox-38.3.0-2.el7_1)
Firefox on SUSE Enterprise Linux Server	Firefox ESR 41

6. Log in as a root user, as shown in Figure 7-2. The root user has administrative privileges and maps to the operating system root user. After you log in, the Resource Dashboard is displayed in the Web Portal.

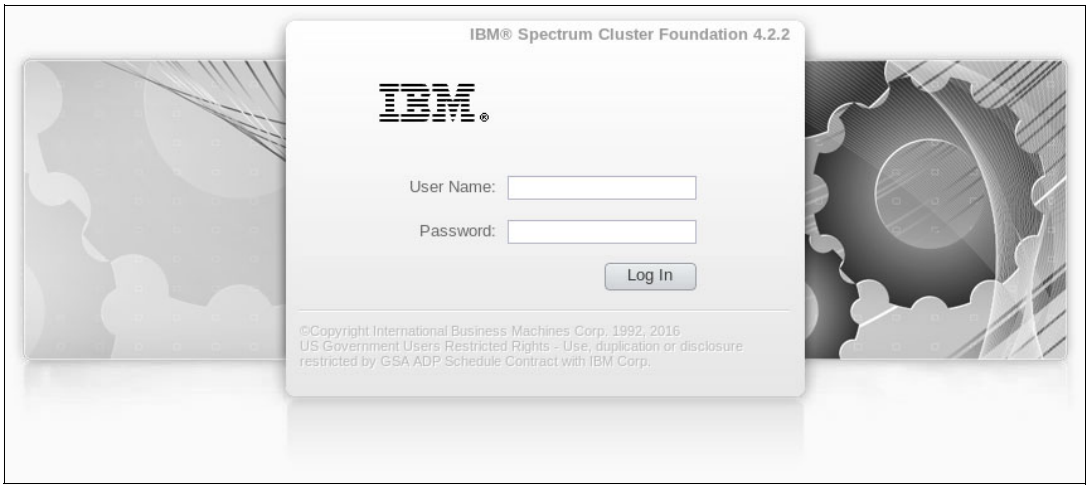


Figure 7-2 IBM Spectrum Cluster Foundation Log In dialog box

7. After login, Figure 7-3 shows the Resource Dashboard is displayed in the Web Portal.

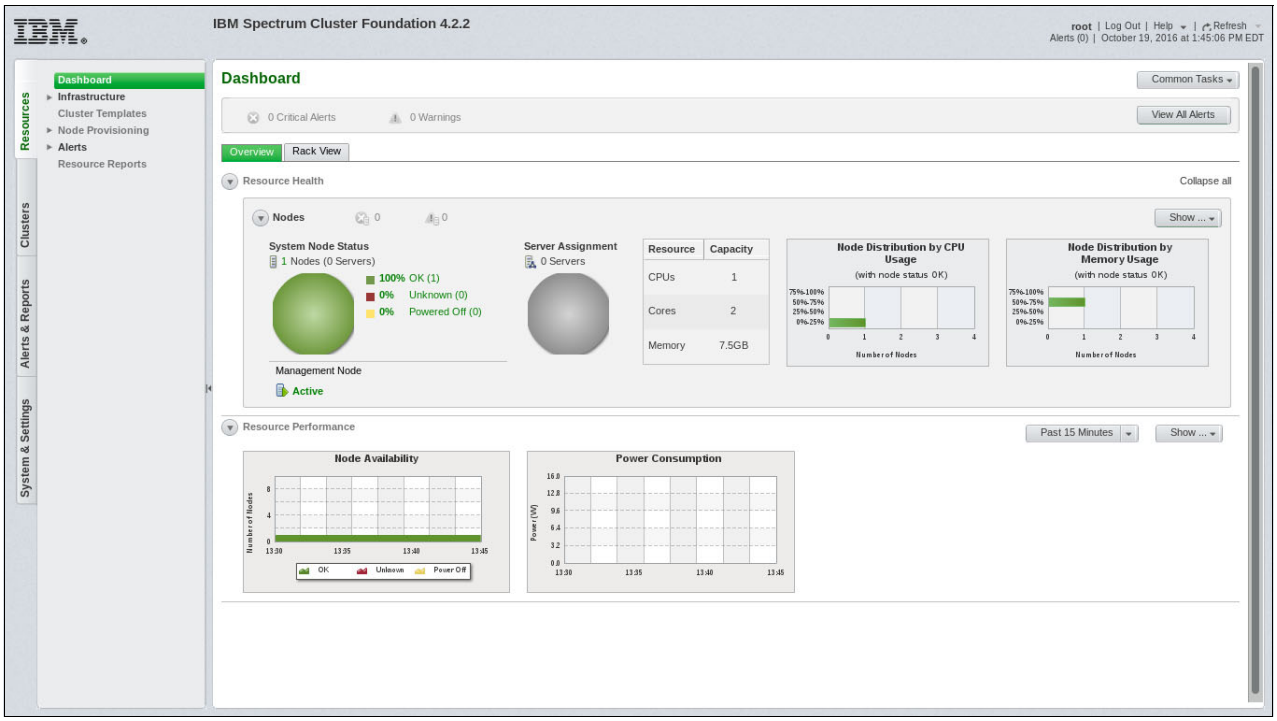


Figure 7-3 IBM Spectrum Cluster Foundation Dashboard

> **Note:** For more information about the IBM Cluster Foundation installation, see IBM Knowledge Center Installing IBM Spectrum Cluster Foundation V4.2.2.

7.2.3 Cluster management

IBM Spectrum Cluster Foundation makes it easy to deploy and manage a technical computing cluster. It provides a robust set of cluster management capabilities that includes cluster provisioning, cluster monitoring, and node management.

All the functions that are required to operate a cluster are installed concurrently and are tightly integrated allowing system administrators to manage a complex cluster as a single system. The administrator can automate deployment of operating system and software components, manage node provisioning and perform cluster maintenance.

7.2.4 High availability

High availability minimizes cluster downtime. IBM Spectrum Cluster Foundation has a tool to set up, configure, and manage high availability.

> **Note:** Enabling high availability is a nonreversible operation. After you enable high availability, it cannot be disabled.

A primary management node is the first management node installed, and it is the first management node to assume the active node role after high availability is enabled. It stores all cluster data, and acts as the active management node for workload management.

A secondary management node is the second management node that is installed and configured for the high-availability cluster. By default, it runs as the active management node after the initial high availability enablement.

Active and standby are the current states of the primary and secondary management nodes. An active management node is the management node that is currently running all workload management services. A standby management node is the management node that is waiting to run all services if a failover occurs. A failover is the process where services migrate from the active management node to the standby management if a disruption occurs on the active management node. After the services are migrated, the standby management node becomes the new active management node.

The high availability manager is a service that runs on both management nodes. It monitors the heartbeat signal and controls all services using the HA service agent. If any service that is controlled by the HA manager goes down, the HA manager restarts that service. If the active management node goes down, the HA manager detects that an error has occurred and migrates all controlled services to the standby management node. IBM Spectrum Cluster Foundation uses EGO service controller (EGOSC) as the HA manager.

Setting up high availability

This section shows the steps required to configure the IBM Spectrum Cluster Foundation high availability environment, refer to Table 7-4 for a summary.

Note: Setting up a high availability environment on an existing cluster, can temporarily impact the operation of your cluster. To minimize the impact to your cluster and end users, schedule the setup of your high availability environment.

Table 7-4 High availability environment roadmap

Actions	Description
Ensure that the high availability requirements are met	Requirements for setting up a share storage device and a secondary management node must be met
Preparing high availability	Set up the secondary management node with an operating system and IBM Spectrum Cluster Foundation installation
Enable an IBM Spectrum Cluster Foundation high availability environment	Set up IBM Spectrum Cluster Foundation high availability on the primary and secondary management nodes
Complete the high availability enablement	After high availability is enabled setup up the compute nodes
Verify IBM Spectrum Cluster Foundation high availability	Ensure that IBM Spectrum Cluster Foundation high availability is running correctly on the primary and secondary management nodes
Troubleshooting enablement problems	Troubleshooting problems that occurred during an IBM Spectrum Cluster Foundation high availability environment setup

High availability requirements

You must make sure that these requirements are met before you set up high availability in IBM Spectrum Cluster Foundation.

Management node requirements

Requirements for the primary management node and the secondary management node in a high availability environment:

- The management nodes must have the same or similar hardware requirements.
- The management nodes must have the same partition layout.

After you prepare the secondary management node, you can ensure that the secondary node uses the same partition schema as the primary management node. Use **`df -h`** and **`fdisk -l`** to check the partition layout. If the secondary node has a different partition layout, reinstall the operating system with the same partition layout:

- The management nodes must use the same network settings.
- The management nodes must use the same network interface to connect to the provision and public networks.

Ensure that the same network interfaces are defined for the primary and secondary management nodes. On each management node, issue the **`ifconfig`** command to check that the network settings are the same. Additionally, ensure that the IP address of same network interface is in the same subnet. If not, reconfigure the network interfaces on the secondary management node according to your network plan.

The management nodes must be configured with the same time, time zone, and current date.

Virtual network requirements

Virtual network information is needed to configure and enable high availability. Collect the following high availability information:

- Virtual management node name
- Virtual IP address for public network
- Virtual IP address for provision network
- Shared directory for user home
- Shared directory for system work data

Note: In a high availability environment, all IP addresses (management nodes IP addresses and virtual IP address) are in the IP address range of your network. To ensure that all IP addresses are in the IP address range of your network, you can use sequential IP addresses. Sequential IP addresses can help avoid any issues as shown in Table 7-5.

Table 7-5 Example: Sequential IP addresses

Network	IP Address range	Primary management node	Secondary management node	Virtual IP address
public	192.168.0.3-192.168.0.200	192.168.0.3	192.168.0.4	192.168.0.5
provision	172.20.7.3-172.20.7.200	172.20.7.3	172.20.7.4	172.20.7.5

The actual node network configuration can be listed using the **`lsdef xCat`** command, as shown in Example 7-6.

Example 7-6 Using lsdef xCat command to show the network definition

```
# lsdef -t network -l
Object name: provision
    domain=pok.stglabs.ibm.com
    dynamicrange=10.0.0.201-10.0.0.254
    gateway=<xcatmaster>
```

```
    mask=255.255.255.0
    mgtifname=eth1
    net=10.0.0.0
    staticrange=10.0.0.3-10.0.0.90
    staticrangeincrement=1
Object name: public
    domain=pok.stglabs.ibm.com
    gateway=9.47.79.254
    mask=255.255.240.0
    mgtifname=eth0
    net=9.47.64.0
    staticrange=9.47.76.96-9.47.76.254
    staticrangeincrement=1
    tftpserver=9.47.76.96
```

Shared storage requirements

IBM Spectrum Cluster Foundation supports either an NFS or IBM Spectrum Scale as shared storage. For a list of supported versions, see IBM Knowledge Center IBM Spectrum Cluster Foundation V4.2.2 Supported software.

By default, two shared directories are required in a high availability environment; one to store user data and one to store system work data. In a high availability environment, all shared file systems must be accessible by the provision network for both the management nodes and compute nodes.

The following shared file systems must already be created on your shared storage server before you set up and enable a high availability environment:

- Shared directory for system work data

  The minimum available shared disk space that is required is 40 GB. Required disk space varies based on the cluster usage.

  The read, write, and run permissions must be enabled for the operating system root user and the IBM Spectrum Cluster Foundation administrator. By default, the IBM Spectrum Cluster Foundation administrator is pcmadmin.

- Shared directory for user data (/home)

  Ensure that there is enough disk space for your data in your `/home` directory. The minimum available shared disk space that is required is 4 GB, and it varies based on the disk space requirements for each user and the total user number. If not provided, the user data is stored together with system work data.

  The read and write permissions must be enabled for all users.

Additionally, the following shared file system requirements must be met:

- The shared file systems cannot be one of the management nodes.
- The shared file system is specific for and only used for the high availability environment. This ensures that no single point of failure (SPOF) errors occur.
- If the IP address of the shared storage server is in the network IP address range that is managed by IBM Spectrum Cluster Foundation, it must be added as an unmanaged device to the cluster to avoid any IP address errors.
- If using an external NAS or NFS server to host the shared directories that are needed for high availability, the following parameters must be specified in the exports entries:

  `rw,sync,no_root_squash,fsid=`*num*

where *num* is an integer and is different for each shared directory.

For example, to create a shared `data` and a shared `home` directory on an external NFS server, use the following commands:

```
#mkdir -p /export/data
#mkdir -p /export/home
```

Next, modify the `/etc/exports` file on the external NFS server:

```
/export/ 172.20.7.0/24(rw,sync,no_root_squash,fsid=0)
```

Note: If you are using two different file systems to create the directories, ensure that the **`fsid`** parameter is set for each export entry. For example:

```
/export/data 172.20.7.0/24(rw,sync,no_root_squash,fsid=3)
/export/home 172.20.7.0/24(rw,sync,no_root_squash,fsid=4)
```

Prepare for high availability

This section describes the steps required to set up high availability.

Before you begin

Ensure that all high availability requirements are met and a shared file system is created on a shared storage server.

To prepare a high availability environment, set the secondary management node with the same operating system and IBM Spectrum Cluster Foundation version as on the primary management node. After the secondary management node is set up, the necessary SSH connections and configuration must be made between the primary management node and the secondary management node. Both management nodes must use the same network and must be connected to the same network interface.

1. Install the operating system on the secondary node.
2. Ensure that the time and time zone is the same on the primary and secondary management nodes.
3. Install IBM Spectrum Cluster Foundation on the secondary node. You must use the same IBM Spectrum Cluster Foundation ISO file as you used for the management node. You can complete the installation using the installer or the silent installation.

   Note: The installer includes an interactive display where you can specify your installation options, make sure to use the same installation options as the primary management node. Installation options for the primary management node are found in the installation log file (`/opt/pcm/log/pcm-installer.log`) on the primary management node.

4. Verify that the management nodes can access the share file systems:
   a. If you are using NFS as shared storage issue the **`showmount -e`** ***`nfs-server-ip`*** command, where *`nfs-server-ip`* is the IP address of the NFS server that connects to the provision network.
   b. If you are using IBM Spectrum Scale as shared storage on the management node run the **`mmgetstate -a`** command. The GPFS state of management node and secondary management node is in Active state as shown in Example 7-7.

   Example 7-7 Get state of the IBM Spectrum Scale cluster

   ```
   # mmgetstate -a
   ```

```
Node number  Node name        GPFS state
-----------------------------------------
       1      compute001       active
       2      compute002       active
       3      compute000       active
       4      management node         active
       5      secondary management node   active
```

5. Add the secondary management node entry to the /etc/hosts file on the primary management node. Ensure that the failover node name can be resolved to the secondary management node provision IP address. Run the command as follows on the primary management node:

   ```
   #echo "secondary-node-provision-ip secondary-node-name" >> /etc/hosts
   #ping secondary-node-name
   ```

   where secondary-node-provision-ip is the provision IP address of the secondary node and secondary-node-name is the name of the secondary node.

   For example: #echo "192.168.1.4 backupmn" >> /etc/hosts

6. Back up and configure a passwordless SSH connection between the primary management node and the secondary node:

   a. Back up the SSH key on the secondary node:

      ```
      ssh secondary-node-name cp –rf /root/.ssh /root/.ssh.PCMHA
      ```

   b. Configure passwordless SSH between the management node and the secondary node:

      ```
      cat /root/.ssh/id_rsa.pub > /root/.ssh/authorized_keys
      scp –r  /root/.ssh/* secondary-node-name:/root/.ssh
      ```

      where secondary-node-provision-ip is the provision IP address of the secondary node and secondary-node-name is the name of the secondary node.

7. *Only for NFS*: Prepare the compute nodes.

NFS Steps:

These steps are used for provisioned compute nodes that you do not want to reprovision:

1. Shut down services on the compute nodes:

   ```
   updatenode __Managed 'KIT_PCM_setupego shutdown -f'
   ```

2. Unmount and remove the /home and /share mount points on the compute nodes:

   ```
   # updatenode __Managed 'mountnfs del'
   # xdsh __Managed ‘umount /home’
   # xdsh __Managed ‘umount /share’
   ```

Enable a high availability environment

This section describes how to enable the high availability environment.

Before you begin

To prepare to enable an HA environment, complete the following steps:

1. Ensure that the secondary management node is installed and set up correctly.

2. Ensure that SSH connections are configured and network settings are correct between the primary management node and the secondary management node.

About this task

You can set up the high availability environment using the high availability management tool (`pcmhatool`). The tool defines and sets up a high availability environment between the management nodes using a predefined high availability definition file.

Note: The high availability management tool (`pcmhatool`) supports Bash shell only.

Procedure

To enable a high availability environment, complete the following steps:

1. Define a high availability definition file according to your high availability settings, including virtual name, virtual IP address, and shared storage, as shown in Example 7-8. The high availability definition file example `ha.info.example` is in the `/opt/pcm/share/examples/HA` directory.

Example 7-8 High availability definition file for Cluster foundation configuration

```
st6a04-rh-cf:
   nicips.eth1:0=10.0.0.4
   nicips.eth0:0=9.47.76.106
   sharefs_type= gpfs
   sharefs_mntp.work =/gpfs
```

Note: For more information, see IBM Knowledge Center IBM Spectrum Cluster Foundation V4.2.2 topic regarding the High availability definition file.

2. Set up a high availability environment.

   Setup can take several minutes to synchronize data to share storage. Ensure that the share storage server is always available. Issue the following command on the primary management node:

   `#pcmhatool config -i` *`ha-definition-file`* `-s` *`secondary-management-node`*

   where *`ha-definition-file`* is the high availability definition file shown in Example 7-8, and *`secondary-management-node`* is the name of the secondary management node.

See Appendix A, “IBM Cluster Foundation high availability configuration example” on page 189 for an output sample of the **`pcmhatool`** command.

Usage Notes:

1. During a high availability enablement, some of the services start on the standby management node instead of the active management node. After a few minutes, they switch to the active management node.
2. If the management node crashes during the high availability environment setup, rerun the **`pcmhatool`** command and specify the same options. Running this command again cleans up the incomplete environment and starts the high availability enablement again.
3. You can find the enablement log file (`pcmhatool.log`) in the `/opt/pcm/log` directory. This log file includes details and results about the high availability environment setup.

Verifying the High Availability configuration

After the high availability enablement is complete, verify that the IBM Spectrum Cluster Foundation high availability environment is set up correctly:

1. Log on to the management node as a root user.
2. Source IBM Spectrum Cluster Foundation environment variables:

   ```
   # . /opt/pcm/bin/pcmenv.sh
   ```

3. Check that IBM Spectrum Cluster Foundation high availability is configured using the **pcmhatool** command, as detailed in Figure 7-4.

```
# pcmhatool info
   Configuring status: OK
   ================================================================
   High availability group members: st6a04-rh-cf01, st6a04-rh-cf02
   Virtual node name: st6a04-rh-cf
   Virtual IP address: eth1:0; 10.0.0.4
   Virtual IP address: eth0:0; 9.47.76.106
   Shared work directory on: /gpfs
```

Figure 7-4 pcmhatool command

4. Check that IBM Spectrum Cluster Foundation services are running. All services must be in state STARTED as shown in Figure 7-5 on page 122.

```
# service pcm status

Cluster name            : PCM
EGO master host name    : st6a04-rh-cf01
EGO master version      : 3.4
SERVICE  STATE    ALLOC CONSUMER RGROUP RESOURCE SLOTS SEQ_NO INST_STATE ACTI
PCM-PLC  STARTED  33    /Manage* Manag* st6a04-* 1     1      RUN        82
PCM-WEB  STARTED  31    /Manage* Manag* st6a04-* 1     1      RUN        81
PTC      STARTED  34    /Manage* Manag* st6a04-* 1     1      RUN        83
PCMD     STARTED  36    /Manage* Manag* st6a04-* 1     1      RUN        85
PCM-PUR* STARTED  35    /Manage* Manag* st6a04-* 1     1      RUN        84
elk-ela* STARTED  24    /Manage* Manag* st6a04-* 1     1      RUN        90
                                       st6a04-* 1     2      RUN        75
elk-kib* STARTED  26    /Manage* Manag* st6a04-* 1     1      RUN        87
elk-ind* STARTED  27    /Manage* Manag* st6a04-* 1     1      RUN        91
                                       st6a04-* 1     2      RUN        74
RULE-EN* STARTED  37    /Manage* Manag* st6a04-* 1     1      RUN        86
elk-shi* STARTED  28    /Cluste* Inter* st6a04-* 1     1      RUN        89
                                       st6a04-* 1     2      RUN        73
ACTIVEMQ STARTED  32    /Manage* Manag* st6a04-* 1     1      RUN        77
PCMDB    STARTED  29    /Manage* Manag* st6a04-* 1     1      RUN        78
PCMHA    STARTED  25    /Manage* Manag* st6a04-* 1     1      RUN        88
XCAT     STARTED  30    /Manage* Manag* st6a04-* 1     1      RUN        80
```

Figure 7-5 List status of pcm services

5. Log in to the Web Portal.

a. Open a supported web browser. Refer to the Release Notes for a list of supported web browsers.

b. Go to http://<*mgtnode-virtual-IP*>:8080, where <*mgtnode-virtual-IP*> is the management node virtual IP address. If you are connected to a public network, you can also navigate to http://<*mgtnode-virtual-hostname*>:8080, where <*mgtnode-virtual-hostname*> is the virtual management node hostname.

   If HTTPS is enabled, go to https://<*mgtnode-virtual-IP*>:8443 or https://<*mgtnode-virtual-hostname*>:8443 to log in to the web portal.

c. Log in as a root user. The root user has administrative privileges and maps to the operating system root user.

d. After you log in, the Resource Dashboard is displayed in the Web Portal. Under the Cluster Health option, both management nodes are listed, as shown in Figure 7-6.

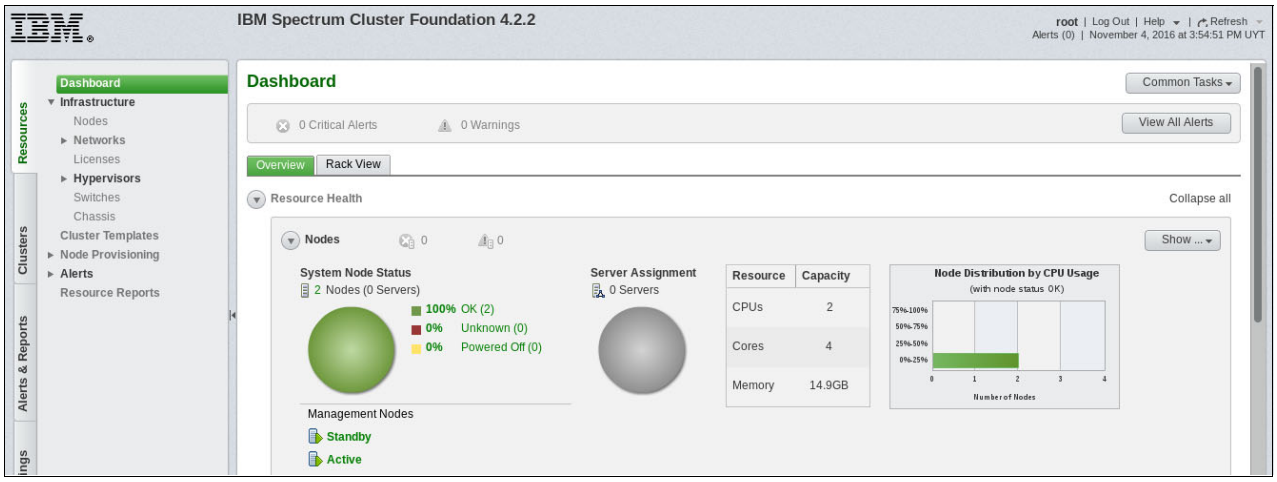


Figure 7-6 IBM Spectrum Cluster Foundation Dashboard HA configuration

7.3 IBM Spectrum Cluster Foundation Community Edition

IBM Spectrum Cluster Foundation Community Edition is a no-charge infrastructure lifecycle management solution providing systems deployment, monitoring, and advanced software management capabilities. You can download this solution from the IBM Spectrum Cluster Foundation Community Edition website.

Featuring a web-based user interface, IBM Spectrum Cluster Foundation allows system administrators to manage complex clusters as a single system from anywhere. All the functions required to operate a cluster are installed at one time and are tightly integrated.

The software offers highly scalable cluster management with support for clusters containing up to 2,500 nodes. It can provision the operating system together with specific software components, delivering both flexibility and ease of use.

IBM Spectrum Cluster Foundation supports the ability to build out mixed computing environments, allowing organizations to take advantage of the current IBM POWER8 and x86 64-bit hardware together. Using IBM POWER solutions, you have the choice to use PowerNV nodes and PowerKVM hypervisors for the underlying nodes.

8

Use case scenarios

This chapter discusses case scenarios using IBM Spectrum Computing solutions to solve common business problems. This chapter installs the products, but also works on publicly available data and tutorials to show how to improve HPC, grid computing, and cloud ready applications using a single environment and sharing resources across all these applications. For these use cases, our environment is composed of IBM Linux little endian (LE) for Power Systems Servers, although it is possible to create the scenario on an x86 cluster too.

This chapter describes the following topics:

- Creating a multi-tenant Spark and cloud-ready application environment
- Spark and static data processing use case
- Spark and streaming data processing use case
- Multi-head cluster for improved multi-tenancy
- Multi-head possibilities
- Adding IBM Spectrum Symphony to the existing IBM Spectrum Conductor cluster
- Adding MongoDB to the IBM Spectrum Conductor

8.1 Creating a multi-tenant Spark and cloud-ready application environment

IBM Spectrum Conductor with Spark is a multi-tenant solution for Apache Spark. It enables you to efficiently deploy and manage multiple Spark deployments. With this capability IBM Spectrum Conductor provides multi tenancy through Spark instance groups, which are akin to the Spark notion of *tenant*. You can run multiple instances of Spark, including different Spark versions, in a shared environment. It improves the performance and efficiency using granular and dynamic resource allocation for Spark instance groups that share a resource pool.

Integrating with Docker, IBM Spectrum Conductor provides flexible and efficient data management for shared storage and high availability by connecting to existing storage infrastructure, such as NFS mounts to a file system or IBM Spectrum Scale. IBM Spectrum Scale specifically is a cluster file system that provides concurrent access to a single file system or set of file systems from multiple hosts. It enables high-performance access to this common set of data to support a scale-out solution and provide a high-availability platform.

The next section shows you how to install Spark using a previous installed IBM Spectrum Scale file system for a single name space across all nodes in the cluster.

8.1.1 Installing IBM Spectrum Conductor with Spark

Just for testing and demonstration purposes, this section shows how to install the IBM Spectrum Conductor with Spark version 2.2 on six LPARs using Red Hat Enterprise Linux 7.2 Little Endian on an IBM Power Systems environment. The requirements for a cluster production environment can vary depending on the needs of each business. For the minimum requirements for running IBM Spectrum Conductor with Spark, see IBM Knowledge Center System requirements and recommendations.

Now, follow the installation steps found at IBM Knowledge Center.

The intention of this section is to have a small “cookbook” to have you ready to start analyzing data and perform basic data analysis using different tools.

Confirming prerequisites

The first step to a successful installation is to ensure that all prerequisites are met before you begin:

1. Ensure that your hosts respond to the fully qualified domain name (FQDN).
2. Ensure that you have a valid cluster admin user if not using root (our case uses egoadmin).
3. Date and time must be the same on all systems (use ntp to enforce clock sync).
4. Install required packages (cURL; Python 2.7.5, 2.7.6, 2.7.7, or 3.x; OpenSSL 1.0.1 or higher; gettext; bind-utils; iproute; acl; and net-tools) if not already installed.
5. Set the number of processes (nproc) and the number of open files to 65,535 for the cluster admin user.

Complete the following steps:

1. Example 8-1 on page 127 shows how to confirm that all servers have the correct FQDN and are known by this name.

Example 8-1 Checking fully qualified domain name setting and resolution

```
[root@st6a04-rh-sym1 /]# hostname -f
st6a04-rh-sym1.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# ssh sym2 hostname -f
st6a04-rh-sym2.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# ssh sym3 hostname -f
st6a04-rh-sym3.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# ssh sym4 hostname -f
st6a04-rh-sym4.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# ssh sym5 hostname -f
st6a04-rh-sym5.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# ssh sym6 hostname -f
st6a04-rh-sym6.pok.stglabs.ibm.com
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.98
10.0.0.98       st6a04-rh-sym1.pok.stglabs.ibm.com st6a04-rh-sym1 sym1
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.99
10.0.0.99       st6a04-rh-sym2.pok.stglabs.ibm.com st6a04-rh-sym2 sym2
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.100
10.0.0.100      st6a04-rh-sym3.pok.stglabs.ibm.com st6a04-rh-sym3 sym3
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.101
10.0.0.101      st6a04-rh-sym4.pok.stglabs.ibm.com st6a04-rh-sym4 sym4
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.102
10.0.0.102      st6a04-rh-sym5.pok.stglabs.ibm.com st6a04-rh-sym5 sym5
[root@st6a04-rh-sym1 /]# getent hosts 10.0.0.103
10.0.0.103      st6a04-rh-sym6.pok.stglabs.ibm.com st6a04-rh-sym6 sym6
[root@st6a04-rh-sym1 /]#
```

2. Example 8-2 shows how to confirm that you have the correct user and the ID is the same on all servers because we are using a local user. A central user configuration with active directory (AD) or LDAP can be used.

Example 8-2 Checking the existence of correct users on all hosts of the clusters

```
[root@st6a04-rh-sym1 /]# id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
[root@st6a04-rh-sym1 /]# ssh sym2 id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
[root@st6a04-rh-sym1 /]# ssh sym3 id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
[root@st6a04-rh-sym1 /]# ssh sym4 id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
[root@st6a04-rh-sym1 /]# ssh sym5 id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
[root@st6a04-rh-sym1 /]# ssh sym6 id egoadmin
uid=1000(egoadmin) gid=1001(egoadmin) groups=1001(egoadmin)
```

3. Now check that all required packages and the Network Time Protocol daemon (ntpd) are installed to configure it later, as shown in Example 8-3.

Example 8-3 Installing software prerequisites

```
[root@st6a04-rh-sym1 /]# yum install curl python openssl gettext bind-utils
iproute acl net-tools ntp
Loaded plugins: product-id, search-disabled-repos, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
```

```
Package curl-7.29.0-25.el7.ppc64le already installed and current version
Package python-2.7.5-34.el7.ppc64le already installed and current version
Package 1:openssl-1.0.1e-42.el7_1.9.ppc64le already installed and current
version
Package gettext-0.18.2.1-4.el7.ppc64le already installed and current version
Package iproute-3.10.0-54.el7.ppc64le already installed and current version
Package acl-2.2.51-12.el7.ppc64le already installed and current version
Package net-tools-2.0-0.17.20131004git.el7.ppc64le already installed and
current version
Package ntp-4.2.6p5-22.el7.ppc64le already installed and current version

Resolving Dependencies
--> Running transaction check
---> Package bind-utils.ppc64le 32:9.9.4-29.el7 will be installed
--> Processing Dependency: bind-libs = 32:9.9.4-29.el7 for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: libbind9.so.90()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: libdns.so.100()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: libisc.so.95()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: libisccc.so.90()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: libisccfg.so.90()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Processing Dependency: liblwres.so.90()(64bit) for package:
32:bind-utils-9.9.4-29.el7.ppc64le
--> Running transaction check
---> Package bind-libs.ppc64le 32:9.9.4-29.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package      Arch       Version          Repository             Size
================================================================================
Installing:
 bind-utils   ppc64le    32:9.9.4-29.el7  xCAT-rhels7.2-path0    199 k
Installing for dependencies:
 bind-libs    ppc64le    32:9.9.4-29.el7  xCAT-rhels7.2-path0    974 k

Transaction Summary
================================================================================
Install  1 Package (+1 Dependent package)

‘Total download size: 1.1 M
Installed size: 3.6 M
Is this ok [y/d/N]: y
Downloading packages:
(1/2): bind-utils-9.9.4-29.el7.ppc64le.rpm         |           199 kB
   00:00:00
(2/2): bind-libs-9.9.4-29.el7.ppc64le.rpm          |           974 kB
00:00:00
--------------------------------------------------------------------------------
```

```
Total
10 MB/s | 1.1 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 32:bind-libs-9.9.4-29.el7.ppc64le 1/2
  Installing : 32:bind-utils-9.9.4-29.el7.ppc64le 2/2
  Verifying  : 32:bind-libs-9.9.4-29.el7.ppc64le 1/2
  Verifying  : 32:bind-utils-9.9.4-29.el7.ppc64le 2/2

Installed:
  bind-utils.ppc64le 32:9.9.4-29.el7

Dependency Installed:
  bind-libs.ppc64le 32:9.9.4-29.el7

Complete!
```

4. Repeat this process on all servers so you clear all dependencies on all servers. Also configure NTP to connect to our NTP server.

   The last prerequisite step is to check the number of open files and processes are adequate. You must have the correct limits at the `/etc/security/limits.conf`, file as shown in Example 8-4.

Example 8-4 Confirming cluster admin limits

```
[root@st6a04-rh-sym1 ~]# grep egoadmin /etc/security/limits.conf
egoadmin   soft    nproc     65535
egoadmin   hard    nproc     65535
egoadmin   soft    nofile    65535
egoadmin   hard    nofile    65535
[root@st6a04-rh-sym1 ~]#
```

Management host installation

After all prerequisites are confirmed, set the installation variables for the installation:

1. The cluster administrative user is `egoadmin`, so you need to set `CLUSTERADMIN=egoadmin`. We set `CLUSTERNAME=itsocluster` to define the cluster name.

   Also, because for this installation we do not have a proper certificate from a trusted certificate authority (CA), we disable SSL using `DISABLESSL=Y`, as shown in Example 8-5.

Example 8-5 Setting initial installation variables

```
[root@st6a04-rh-sym1 ~]# export CLUSTERADMIN=egoadmin
[root@st6a04-rh-sym1 ~]# export CLUSTERNAME=itsocluster
[root@st6a04-rh-sym1 ~]# export DISABLESSL=Y
[root@st6a04-rh-sym1 ~]#
```

Note: The default configuration is to have SSL enabled for extra security. With this configuration, a certificate is generated during installation for proof of concept (POC) purposes. For a production environment, you are required a proper certificate from a trusted certificate authority (CA). When opening it on your browser, check that you have installed the certificate (rather than accept it).

Installing the certificate installs the certificate for use by the cluster management console and the RESTful web servers that require the same certificate. SSL is highly advised when integrity and confidentiality of data transmission is essential.

2. Other optional variables are available, such as setting the proof of concept database (Derby). Set `DERBY_DB_HOST` to the hostname of the server that you want to use the Derby database on.

Note: You cannot use the Derby database for production clusters. To produce regular reports from a production cluster, you must configure an external production database after installation. To configure a database for a production environment, see IBM Knowledge Center Setting up an external database for production.

3. To use Yarn, you need to point Java with the variable `JAVA_HOME` and `HADOOP_YARN_HOME` to a supported Hadoop distribution, as shown in Example 8-6.

Example 8-6 Setting additional installation variables

```
[root@st6a04-rh-sym1 ~]# export DERBY_DB_HOST=st6a04-rh-sym1
[root@st6a04-rh-sym1 ~]# export JAVA_HOME=/usr/bin/java
[root@st6a04-rh-sym1 ~]# export HADOOP_YARN_HOME=/opt/hadoop-2.7.3
[root@st6a04-rh-sym1 ~]#
```

4. Now start the installation process with the IBM Spectrum Conductor with Spark. Run the binary files and the installation process begins, as shown in Example 8-7. Read the License Agreement and press 1 if you agree to start installation process.

Example 8-7 Starting IBM Spectrum Conductor with Spark installation

```
[root@st6a04-rh-sym1 conduct]# ./cws-2.2.0.0_ppc64le.bin
Extracting files... done.
Installation for IBM Spectrum Conductor with Spark 2.2.0

IBM Spectrum Conductor with Spark License Agreement
International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, CLICKING ON
AN "ACCEPT" BUTTON, OR OTHERWISE USING THE PROGRAM,
LICENSEE AGREES TO THE TERMS OF THIS AGREEMENT. IF YOU ARE
ACCEPTING THESE TERMS ON BEHALF OF LICENSEE, YOU REPRESENT
AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND LICENSEE
TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

* DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, CLICK ON AN
"ACCEPT" BUTTON, OR USE THE PROGRAM; AND

* PROMPTLY RETURN THE UNUSED MEDIA, DOCUMENTATION, AND
```

```
Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
1
```

5. The installer shows the packages and the prefix of the installation path, as shown in Example 8-8. If the information is correct press Y and Enter.

Example 8-8 The installer showing the packages and the prefix of the installation

```
The following commands will install the .rpm packages on the host.
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egocore-3.5.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egojre-8.0.3.20.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force egowlp-8.5.5.9.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egorest-3.5.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egomgmt-3.5.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egoyarn-3.5.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egoelastic-1.2.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egogpfsmonitor-3.5.0.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force ascd-2.2.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force conductormgmt-2.2.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
conductorsparkcore-2.2.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
conductorsparkmgmt-2.2.0.0.rpm
Do you want to continue?(Y/N)
```

6. The install process begins and the variables settings you selected are shown in Example 8-9.

Example 8-9 Install process on the management host

```
Start install...
Preparing...                          ################################# [100%]

.
.
.

The installation will be processed using the following settings:
Workload Execution Mode (WEM): Advanced
Cluster Administrator: egoadmin
Cluster Name: itsocluster
Installation Directory: /opt/ibm/spectrumcomputing
Connection Base Port: 7869
Updating / installing...
   1:egocore-3.5.0.0-433344           ################################# [100%]
```

```
Successfully installed the EGO core package.
Preparing...                          ################################# [100%]

.
.
.

Updating / installing...
   1:conductorsparkmgmt-2.2.0.0-433344################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark
management package.
```

Compute host installation

After installing your manager host, you can now install the initial compute hosts you will use in your cluster. The installation process is similar to installing the manager host, but you have one more variable to set before starting the installation process.

Note: All software, user creation, and configuration prerequisites apply to compute hosts too. Repeat all of the steps shown from Example 8-1 on page 127 to Example 8-4 on page 129.

To define the node that you are installing is a compute host, set `EGOCOMPUTEHOST=Y` (yes), as shown in Example 8-10.

Example 8-10 Installing IBM Conductor with Spark on a compute host at a local file system

```
[root@st6a04-rh-sym2 conduct]# export CLUSTERADMIN=egoadmin
[root@st6a04-rh-sym2 conduct]# export CLUSTERNAME=itsocluster
[root@st6a04-rh-sym2 conduct]# export JAVA_HOME=/usr/bin/java
[root@st6a04-rh-sym2 conduct]# export HADOOP_YARN_HOME=/opt/hadoop-2.7.3
[root@st6a04-rh-sym2 conduct]# export EGOCOMPUTEHOST=Y
[root@st6a04-rh-sym2 conduct]#./cws-2.2.0.0_ppc64le.bin
Extracting files... done.
Installation for IBM Spectrum Conductor with Spark 2.2.0

IBM Spectrum Conductor with Spark License Agreement
International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, CLICKING ON
AN "ACCEPT" BUTTON, OR OTHERWISE USING THE PROGRAM,
LICENSEE AGREES TO THE TERMS OF THIS AGREEMENT. IF YOU ARE
ACCEPTING THESE TERMS ON BEHALF OF LICENSEE, YOU REPRESENT
AND WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND LICENSEE
TO THESE TERMS. IF YOU DO NOT AGREE TO THESE TERMS,

* DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, CLICK ON AN
"ACCEPT" BUTTON, OR USE THE PROGRAM; AND

* PROMPTLY RETURN THE UNUSED MEDIA, DOCUMENTATION, AND

Press Enter to continue viewing the license agreement, or
```

```
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
```

Compute host installation

As you completed on the master host, if you read and accept the license, press 1 and Enter to continue with the installation and confirm that the path to install it is correct, as shown in Example 8-11.

Example 8-11 Confirm installation path

```
1
The following commands will install the .rpm packages on the host.
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force egocore-3.5.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force egojre-8.0.3.20.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force egowlp-8.5.5.9.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force egoyarn-3.5.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
egoelastic-1.2.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force ascd-2.2.0.0.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  --force
conductorsparkcore-2.2.0.0.rpm
Do you want to continue?(Y/N)Y
```

The installation process begins and the variables settings you selected are shown in Example 8-12.

Example 8-12 Installation process on a compute host

```
Start install...
Preparing...                          ################################# [100%]
.
.
.

The installation will be processed using the following settings:
Workload Execution Mode (WEM): Advanced
Cluster Administrator: egoadmin
Cluster Name: itsocluster
Installation Directory: /opt/ibm/spectrumcomputing
Connection Base Port: 7869
Updating / installing...
   1:egocore-3.5.0.0-433344           ################################# [100%]

Successfully installed the EGO core package.
.
.
.
Preparing...                          ################################# [100%]

Updating / installing...
   1:conductorsparkcore-2.2.0.0-433344################################# [100%]

Successfully installed IBM Spectrum Computing family: Conductor with Spark core package.
[root@st6a04-rh-sym2 conduct]#
```

Note that the message shown at the end of the installation process shows that the core package has been installed successfully.

Note: Install all computing hosts using this method. You can create an automated script to perform this task and distribute across the hosts.

Configuring Conductor with Spark

If you want to configure the admin user to have root privileges to start and stop Conductor with Spark, you can configure sudoers to do that with the helper `egosetsudoers.sh`. Also, Conductor with Spark can be set to start with the initialization of the operating system using the helper `egosetrc.sh`.

1. To use this helper you need to source the `profile.platform` first (if using the bash shell for csh, use `cshrc.platform`). Both configurations are shown in Example 8-13.

Example 8-13 Enabling sudo for admin user and auto-start for conductor

```
[root@st6a04-rh-sym1 ~]# . /opt/ibm/spectrumcomputing/profile.platform
[root@st6a04-rh-sym1 ~]#
[root@st6a04-rh-sym1 ~]# egosetsudoers.sh
egosetsudoers succeeds
[root@st6a04-rh-sym1 ~]# egosetrc.sh
Egosetrc successful
[root@st6a04-rh-sym1 ~]#
```

Note: Remember to perform this activity on all nodes of the clusters, manager, and compute.

Now you are ready to continue the tasks with the admin user defined. You can configure the initial cluster joining the masterhost as the client and setting the entitlement for Conductor. Remember to source the profile.

2. To create the new cluster, issue the **`egoconfig join <masterhost>`** command. Then set the entitlement with the **`egoconfig setentitlement <entitlement file>`** command. The output from these steps is shown in Example 8-14.

Example 8-14 Creating the cluster and setting entitlement

```
[root@st6a04-rh-sym1 ~]# su - egoadmin
Last login: Fri Mar 10 09:58:17 EST 2017 on pts/0
[egoadmin@st6a04-rh-sym1 ~]$ . /opt/ibm/spectrumcomputing/profile.platform
[egoadmin@st6a04-rh-sym1 ~]$ egoconfig join st6a04-rh-sym1
You are about to create a new cluster with this host as the master host.  Do
you want to continue? [y/n]y
A new cluster <itsocluster> has been created. The host <st6a04-rh-sym1> is the
master host.
You should run <egoconfig setentitlement "entitlementfile"> before using the
cluster.
[egoadmin@st6a04-rh-sym1 ~]$ egoconfig setentitlement
/gpfs/pkgs/conduct/cws_entitlement.dat
The system successfully set the specified entitlement.
```

Note: To ease administration exchange for the SSH keys between master and client nodes for the admin user, remember to import all host keys so that SSH does not prompt for host verification, as shown in Example 8-15 on page 135.

3. Now you can start your cluster, as shown in Example 8-15.

Example 8-15 Starting the cluster

```
[egoadmin@st6a04-rh-sym1 ~]$ egosh ego start all
Do you really want to start up LIM on all hosts ? [y/n]y
Start up LIM on <st6a04-rh-sym1.pok.stglabs.ibm.com> ..... The authenticity of
host 'st6a04-rh-sym1.pok.stglabs.ibm.com (10.0.0.98)' can't be established.
ECDSA key fingerprint is 2c:38:3f:7d:fb:75:40:e0:40:98:8d:bd:79:76:0e:30.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'st6a04-rh-sym1.pok.stglabs.ibm.com,10.0.0.98'
(ECDSA) to the list of known hosts.
done
[egoadmin@st6a04-rh-sym1 ~]$
```

4. Now log on using the graphical user interface. If you do not know if the web interface is up, then log on using your text interface by issuing the **egosh client view GUIURL_1** command, as shown in Example 8-16.

Example 8-16 Finding the URL for the GUI

```
[egoadmin@st6a04-rh-sym1 ~]$ egosh user logon
user account: Admin
password:
Logged on successfully

[egoadmin@st6a04-rh-sym1 ~]$ egosh client view GUIURL_1
-------------------------------------------------------------
CLIENT NAME: GUIURL_1
DESCRIPTION: http://st6a04-rh-sym1.pok.stglabs.ibm.com:8080/platform
TTL        : 0
LOCATION   : 47974@10.0.0.98
USER       : Admin

CHANNEL INFORMATION:
CHANNEL             STATE
13                  CONNECTED
[egoadmin@st6a04-rh-sym1 ~]
```

Note: The administrative user is Admin and the initial password is Admin for both command line and Graphical User Interface login.

The logon page has the user and password automatically, as shown in Figure 8-1.

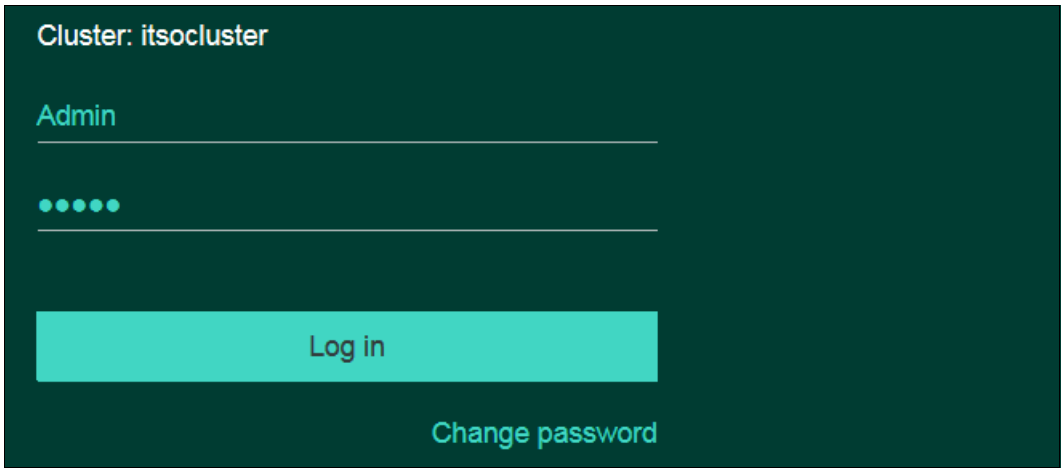


Figure 8-1 IBM Spectrum Conductor Login page

As you log in, you can see a dashboard containing the cluster health information. Note that Figure 8-2 has no clients that have been added. Only one host is seen by the cluster.

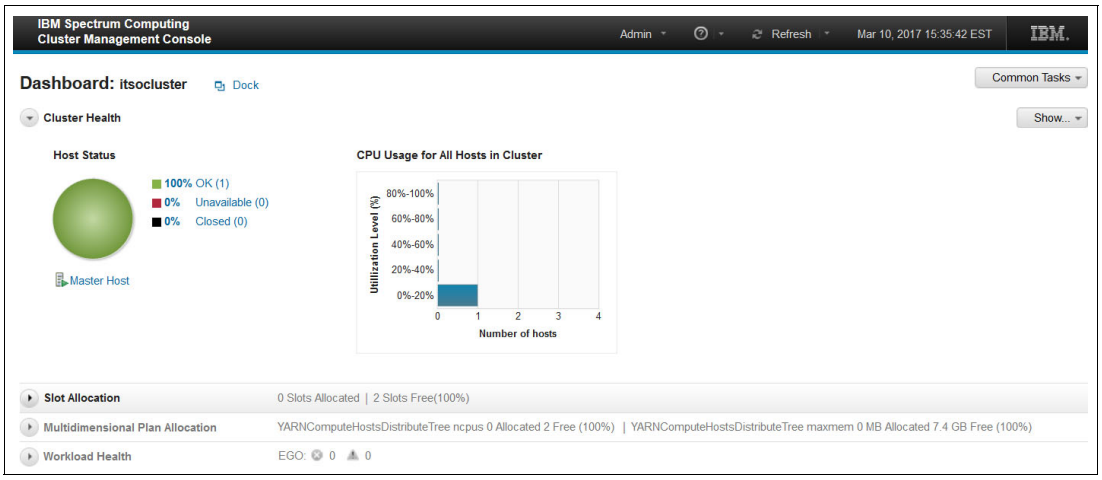


Figure 8-2 IBM Spectrum Conductor with Spark Dashboard

5. Now join compute nodes to the cluster using the source platform profile and issuing the **egoconfig join <master host>** command. Then start the cluster in your compute node, as shown in Example 8-17.

Example 8-17 Adding compute node to the cluster and starting the process

```
[egoadmin@st6a04-rh-sym1 ~]$ ssh st6a04-rh-sym2
Last login: Fri Mar 10 15:45:13 2017 from st6a04-rh-sym1.pok.stglabs.ibm.com
[egoadmin@st6a04-rh-sym2 ~]$ . /opt/ibm/spectrumcomputing/profile.platform
[egoadmin@st6a04-rh-sym2 ~]$ egoconfig join st6a04-rh-sym1
You are about to join this host to a cluster with master host st6a04-rh-sym1.
Do you want to continue? [y/n]y
The host st6a04-rh-sym2 has joined the cluster itsocluster.
[egoadmin@st6a04-rh-sym2 ~]$ egosh ego start
Start up LIM on <st6a04-rh-sym2.pok.stglabs.ibm.com> ...... done
[egoadmin@st6a04-rh-sym2 ~]$
```

Note: Repeat the process shown in Example 8-17 on page 136 on all of your compute nodes. You can create a script to ease the process.

All our hosts joined the cluster. You can see six hosts in the cluster, as shown in Figure 8-3.

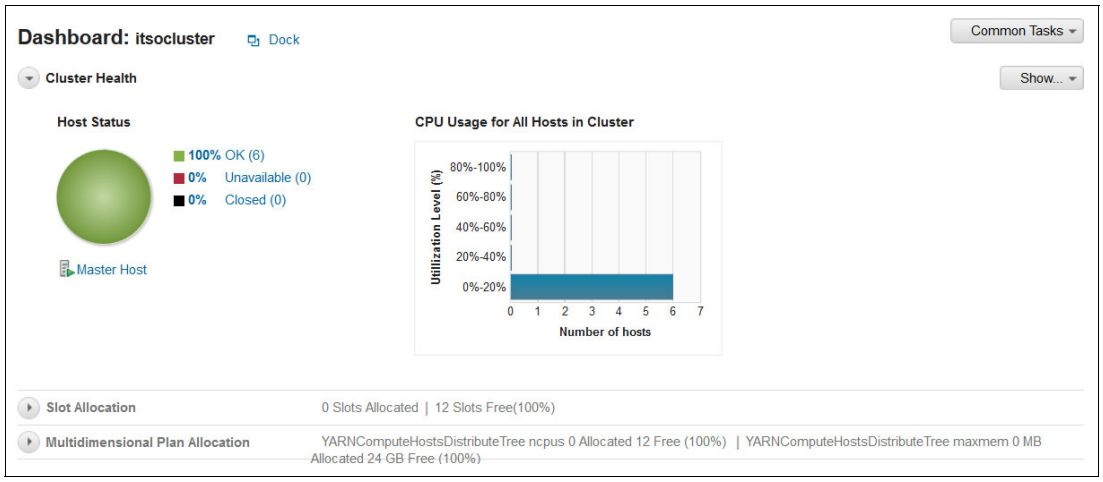


Figure 8-3 Dashboard with all hosts joined the cluster

8.1.2 Creating Spark instances and defining a notebook for a user

With the cluster fully operational, you can now create spark instances.

Note: Check that you are connecting to the management GUI using the name, not the IP address, otherwise you get an HTTP error 401.

1. To create a Spark Instance, select **Workload** → **Spark** → **Spark Instance group**, as shown in Figure 8-4, which brings you to the Spark Instance Group management page.

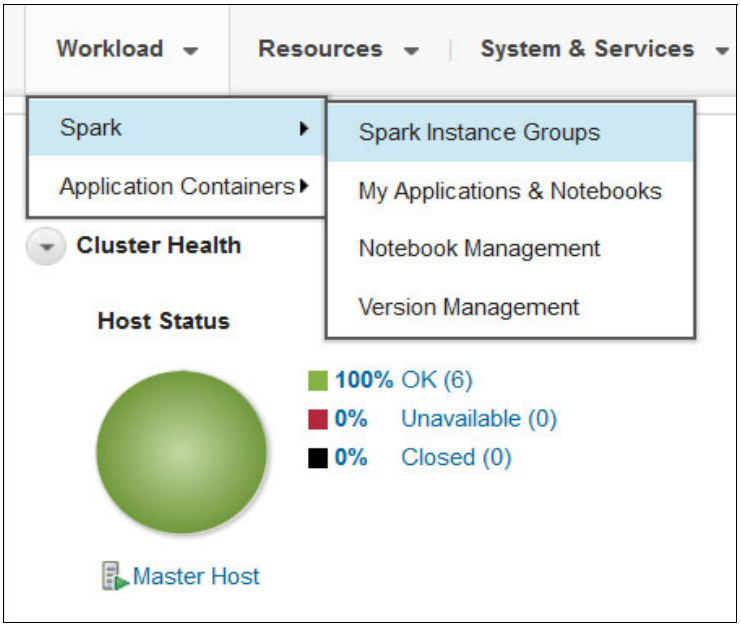


Figure 8-4 Spark Instance Groups Management

2. You see the *Spark Instance Groups for all consumers* page, as shown in Figure 8-5. You can now create a new Spark Instance, which is used for our case scenario.

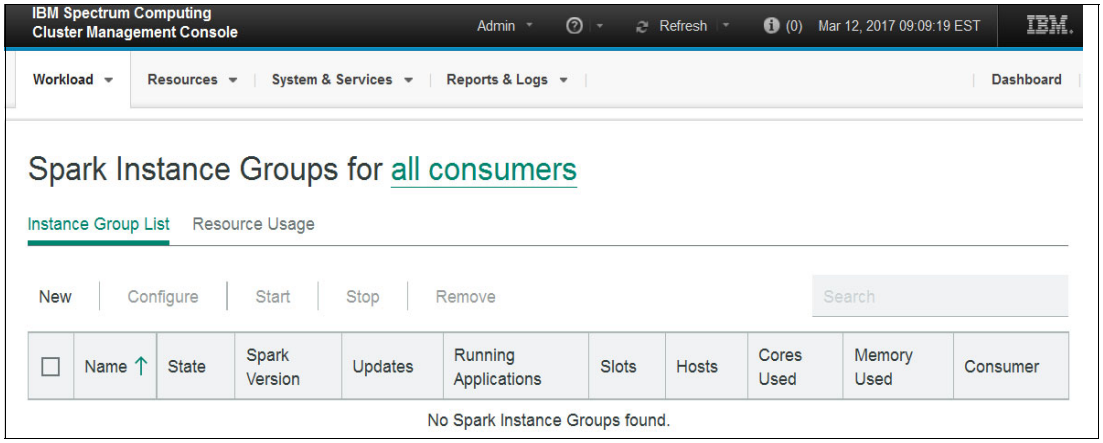


Figure 8-5 Spark Instance Groups for all consumers page

3. Click **New** to open the page for creating the new Spark Instance. For this example, use the Zeppelin Notebook that is supported with Spark V1.6.1 (not on V2.0.1). Our scenario uses the egoadmin user and name with the instance as `spark1` because we want to deploy the instance in the home directory of the egoadmin as `/home/egoadmin/spark1` shown in Figure 8-6. We do not change anything else in this page.

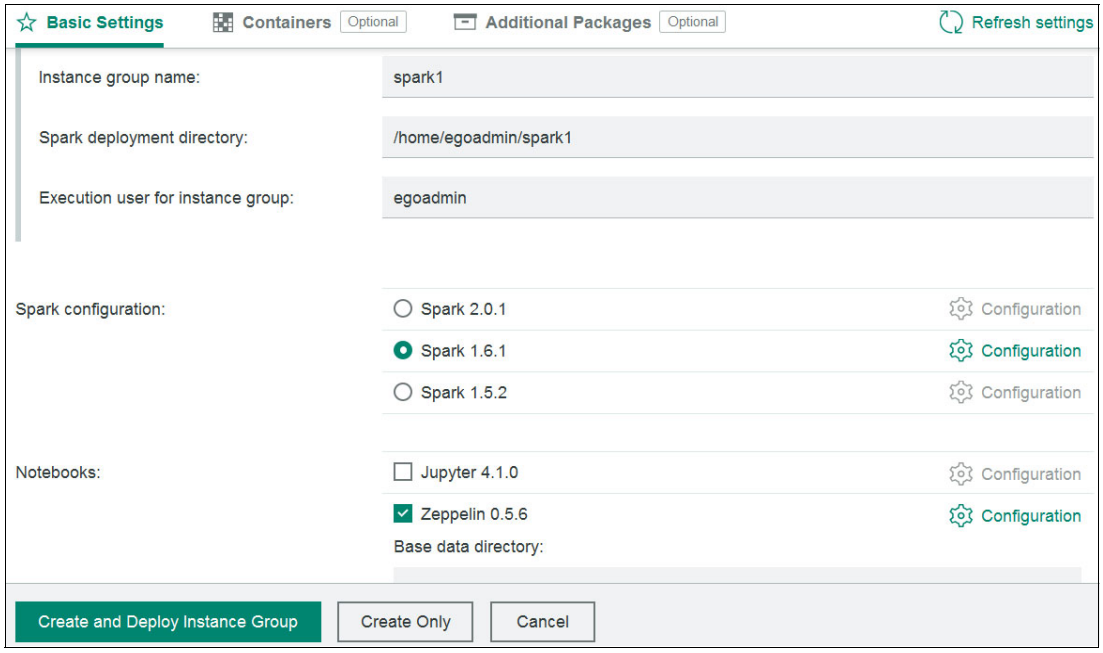


Figure 8-6 Create Spark Instance Page

4. Then click the **Create and Deploy Instance Group** button and Spark is deployed. You see the pane with a button to continue to the newly created Instance group after the deploy is started as shown in Figure 8-7.

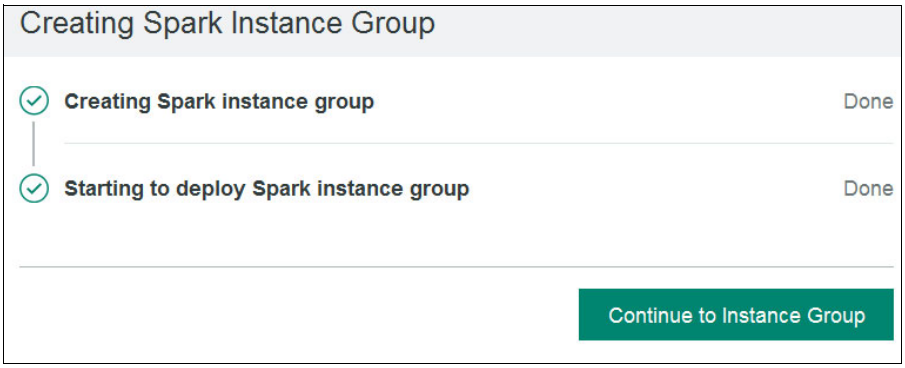


Figure 8-7 Creating Spark Instance Group

5. Click the **Continue to Instance Group**. This can take some time. After a successful deployment, you see a **Ready** status on your Spark Instance as shown in Figure 8-8.

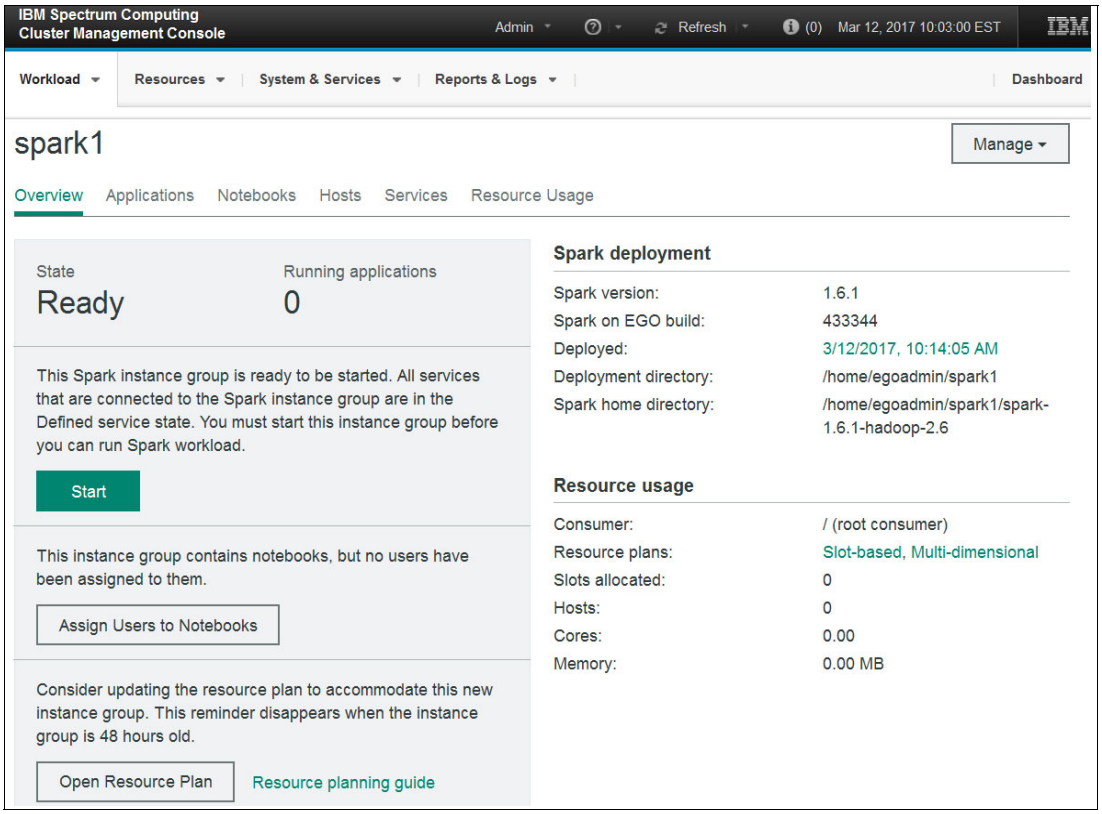


Figure 8-8 Spark Instance Pane

Note: If the page does not update, click the **Refresh** button on the top dark gray bar. You can configure an automatic refresh on the drop-down menu located to the right of the **Refresh** button.

6. Now that your Spark instance is ready, you can start it. Click the **Start** button. After the **State** changes to `Started`, create a notebook to analyze data. Click the **Notebooks** tab to see the Notebook pane, as shown in Figure 8-9.

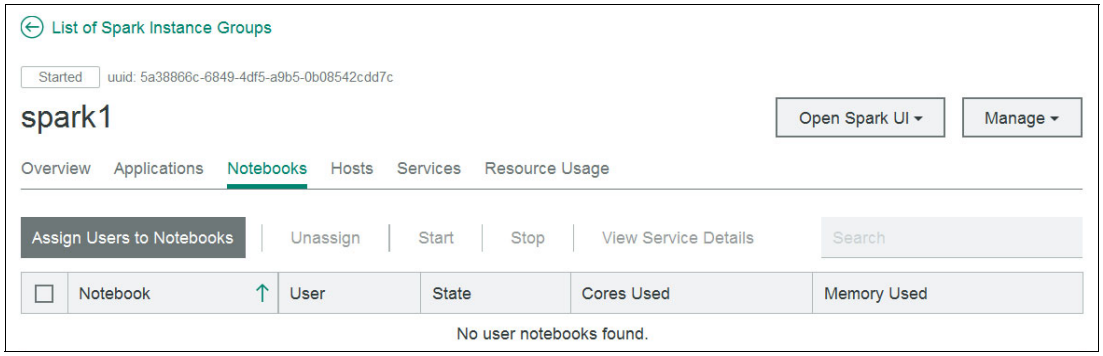


Figure 8-9 Notebooks Panel

7. Click the **Assign Users to Notebooks** button to open a new panel to create a notebook for the user. For demonstration purposes, our scenario uses the `Admin` user. Select the user and click Assign, as shown in Figure 8-10.

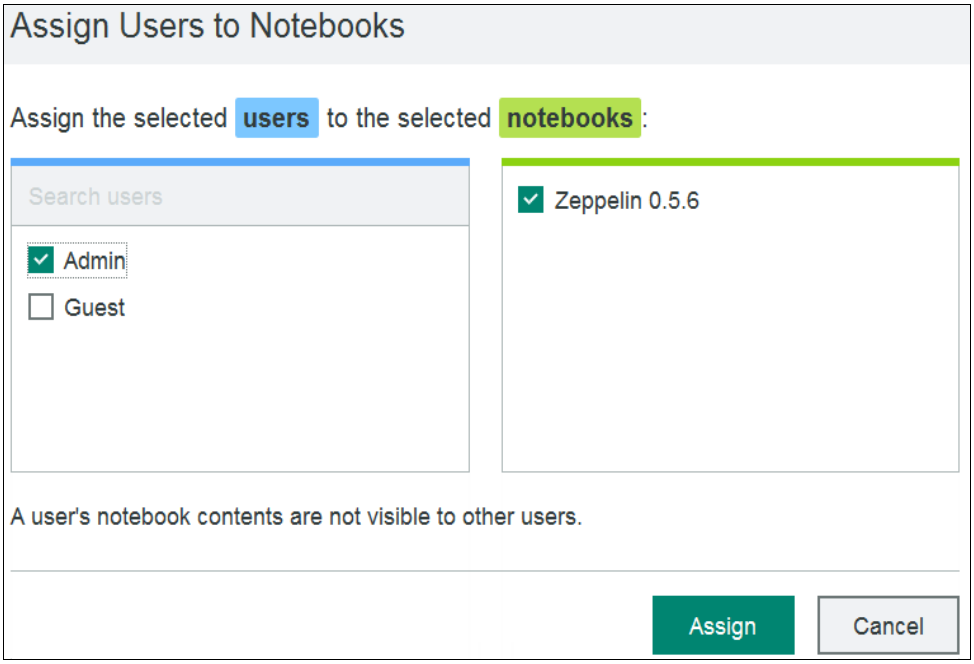


Figure 8-10 Assign user to a notebook

8. After the assignment and the Notebook shows `Started`, as shown in Figure 8-11, select the notebook and click **View Service Details**.

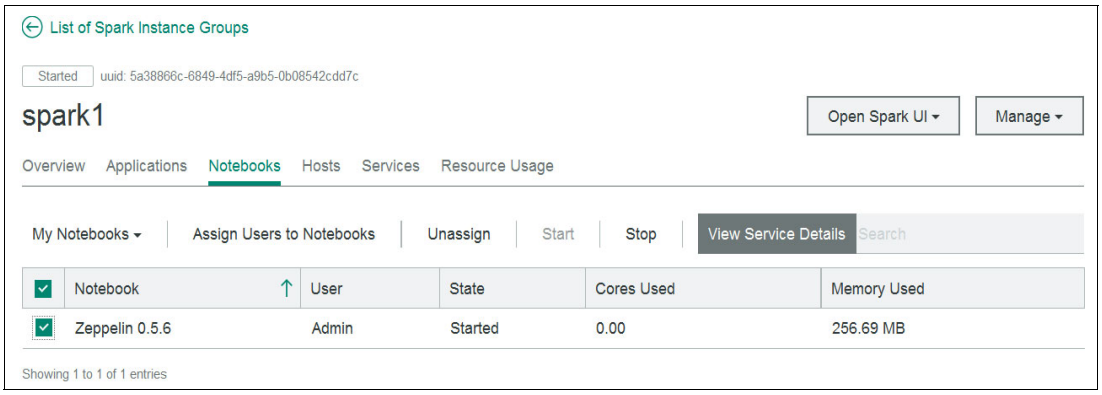


Figure 8-11 Notebook started

9. On the Service Details pane, click the Instances tab to see where the notebook can be Accessed, as shown in Figure 8-12. In this screen, you are able to see if the notebook is available and connect to it.

Figure 8-12 Notebook Instance information

Note: RESTful APIs are available to create an external tool to automate creation notebook assignments and forward the information to the user to access the notebook if you prefer not to use the GUI. Showing how to automate using the RESTful APIs is out of scope of this book, but go to IBM Knowledge Center RESTful APIs for more information.

10. In our case, the notebook is available on the `st6a04-rh-sym3` host. We now point our browser to the location indicated (`http://st6a04-rh-sym3.pok.stglabs.ibm.com:8380`) to access the notebook, as shown in Figure 8-13.

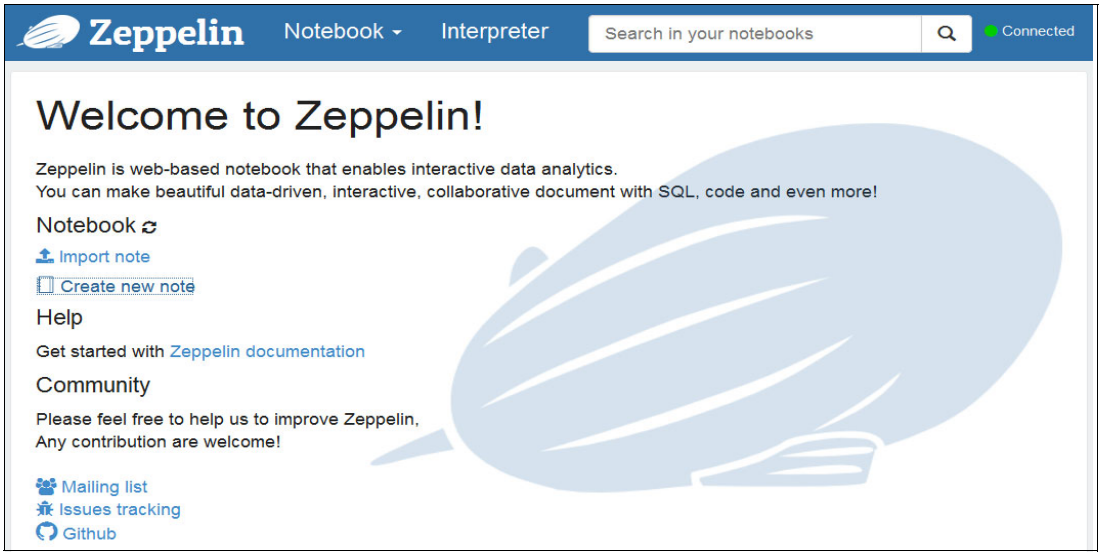


Figure 8-13 Zeppelin Notebook Interface

11. Click **Create new note** and provide your note any name.

8.2 Spark and static data processing use case

Now you have access to a Zeppelin note. You can start analyzing data in your own cluster. This use case scenario uses the data available at the New York City web page.

This example is based on a tutorial found at the IBM Bluemix® site:

The example is changed to work in a Zeppelin notebook.

Complete the following steps:

1. We download the CSV with the data directly to our shared name space created with IBM Spectrum Scale so that it can be accessed by any node of the cluster.
2. We copy the link of the CSV file by right-clicking the **CSV** option on the **Download** menu and selecting **Copy Link Location**, as shown in Figure 8-14 on page 143.

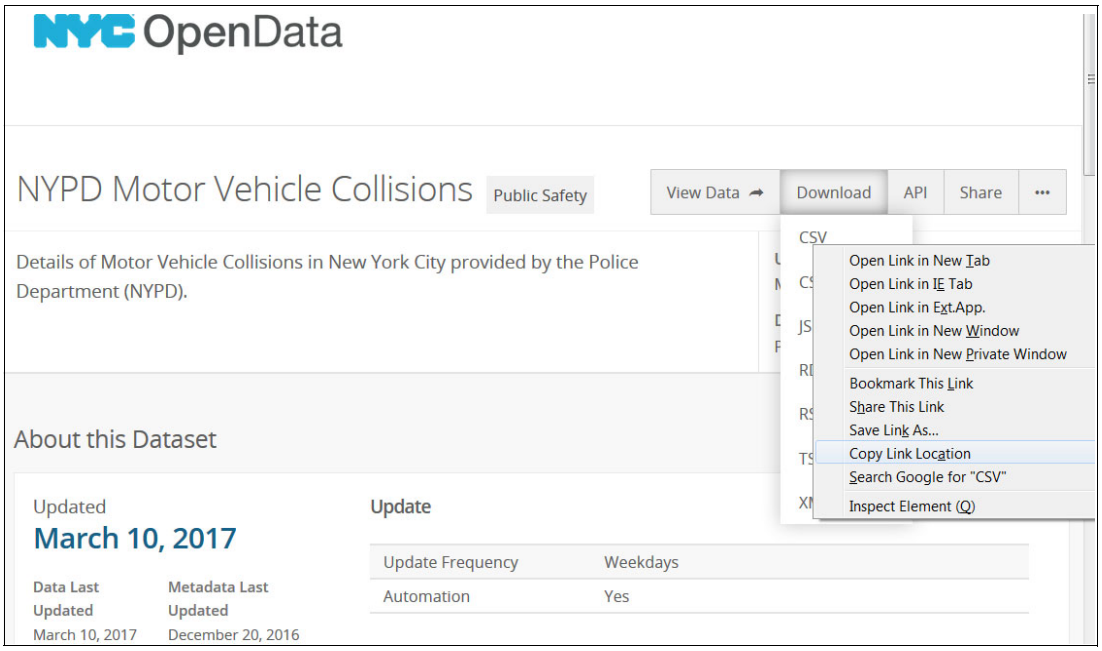


Figure 8-14 Accessing the data available

3. We download it directly to our namespace so we do not need to download to the desktop and then upload it to our cluster as shown in Example 8-18. If needed, you can do this in a two-step approach for your environment.

Example 8-18 Downloading the data

```
[root@st6a04-rh-sym1 gpfs]# wget -O NYPD_Motor_Vehicle_Collisions.csv
https://data.cityofnewyork.us/api/views/h9gi-nx95/rows.csv?accessType=DOWNLOAD
--2017-03-12 12:26:19--
https://data.cityofnewyork.us/api/views/h9gi-nx95/rows.csv?accessType=DOWNLOAD
Resolving data.cityofnewyork.us (data.cityofnewyork.us)... 52.206.140.205
Connecting to data.cityofnewyork.us
(data.cityofnewyork.us)|52.206.140.205|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/csv]
Saving to: 'NYPD_Motor_Vehicle_Collisions.csv'

    [ <=>                                   ] 189,100,287 1.84MB/s   in 98s

Last-modified header invalid -- time-stamp ignored.
2017-03-12 12:27:58 (1.84 MB/s) - 'NYPD_Motor_Vehicle_Collisions.csv' saved
[189100287]
```

4. Taking a quick look at the data to understand it, we see that we have the data values separated by commas and 29 fields for the header. We use bash tools for this first approach, because it is a tool already available after the download. We also can see that we have many years of data. Because we are interested only in 2017, we filter this data.

You can see the first analysis in Example 8-19.

Example 8-19 First approach to understand the data

```
[root@st6a04-rh-sym1 gpfs]# head -1 NYPD_Motor_Vehicle_Collisions.csv|awk -F','
'{print NF} '
29
[root@st6a04-rh-sym1 gpfs]# head -2 NYPD_Motor_Vehicle_Collisions.csv
DATE,TIME,BOROUGH,ZIP CODE,LATITUDE,LONGITUDE,LOCATION,ON STREET NAME,CROSS
STREET NAME,OFF STREET NAME,NUMBER OF PERSONS INJURED,NUMBER OF PERSONS
KILLED,NUMBER OF PEDESTRIANS INJURED,NUMBER OF PEDESTRIANS KILLED,NUMBER OF
CYCLIST INJURED,NUMBER OF CYCLIST KILLED,NUMBER OF MOTORIST INJURED,NUMBER OF
MOTORIST KILLED,CONTRIBUTING FACTOR VEHICLE 1,CONTRIBUTING FACTOR VEHICLE
2,CONTRIBUTING FACTOR VEHICLE 3,CONTRIBUTING FACTOR VEHICLE 4,CONTRIBUTING
FACTOR VEHICLE 5,UNIQUE KEY,VEHICLE TYPE CODE 1,VEHICLE TYPE CODE 2,VEHICLE
TYPE CODE 3,VEHICLE TYPE CODE 4,VEHICLE TYPE CODE 5
03/07/2017,13:37,BROOKLYN,11205,40.697254,-73.96183,"(40.697254,
-73.96183)",,,101       CLASSON AVENUE,0,0,0,0,0,0,0,0,Following Too
Closely,Unspecified,,,,3628188,SPORT UTILITY / STATION WAGON,PASSENGER
VEHICLE,,,
[root@st6a04-rh-sym1 gpfs]#
[root@st6a04-rh-sym1 gpfs]# awk ' $0 ~ /^......2017/ {print $0}'
NYPD_Motor_Vehicle_Collisions.csv >2017_NYPD_Motor_Vehicle_Collisions.csv
[root@st6a04-rh-sym1 gpfs]#
[root@st6a04-rh-sym1 gpfs]# awk -F',' '{print NF} '
2017_NYPD_Motor_Vehicle_Collisions.csv |sort -u
29
30
[root@st6a04-rh-sym1 gpfs]#
```

5. We see on Example 8-19 that in the filtered file we have 29 and 30 fields. If you look at the sixth field, we see that it contains a comma. So on the lines that contain this field, we change the seventh appearance of the comma to a semi-colon and build the final file to be use in our data analysis. This process is shown in Example 8-20.

Example 8-20 Creating the file is used for the analysis

```
[root@st6a04-rh-sym1 gpfs]# #adding the header
[root@st6a04-rh-sym1 gpfs]# head -1 NYPD_Motor_Vehicle_Collisions.csv >
USE_2017_NYPD_Motor_Vehicle_Collisions.csv
[root@st6a04-rh-sym1 gpfs]# #adding the lines with 29 fields
[root@st6a04-rh-sym1 gpfs]# awk -F',' 'NF == 29 {print $0} '
2017_NYPD_Motor_Vehicle_Collisions.csv
>>USE_2017_NYPD_Motor_Vehicle_Collisions.csv
[root@st6a04-rh-sym1 gpfs]# #adding the lines with 30 fields, changing the 7th
comma to a semi-colon.
[root@st6a04-rh-sym1 gpfs]# awk -F',' 'NF == 30 {print $0} '
2017_NYPD_Motor_Vehicle_Collisions.csv |sed 's/,/;/7'
>>USE_2017_NYPD_Motor_Vehicle_Collisions.csv
[root@st6a04-rh-sym1 gpfs]# #checking number of fields
[root@st6a04-rh-sym1 gpfs]#  awk -F',' '{print NF}'
USE_2017_NYPD_Motor_Vehicle_Collisions.csv |sort -u
29
[root@st6a04-rh-sym1 gpfs]# #checking correct comma was changed
[root@st6a04-rh-sym1 gpfs]# tail -1 USE_2017_NYPD_Motor_Vehicle_Collisions.csv
01/01/2017,17:32,BROOKLYN,11215,40.66831,-73.987971,"(40.66831;
-73.987971)",,,311       11 STREET,1,0,1,0,0,0,0,0,,,,,,3590473,,,,,
```

```
[root@st6a04-rh-sym1 gpfs]#
```

For this example, we use the *PySpark Interpreter* and some Python packages that need to be installed on our environment to make sure that we have all needed libraries.

Note: Using yum, install python2-pip, freetype-devel, libpng-devel, python-devel, numpy, and scipy packages.

6. Now, use **pip** to install the seaborn packages that we use to plot graphs to analyze the car crash data, as shown in Example 8-21.

Example 8-21 Installing required packages

```
[root@st6a04-rh-sym6 ~]# pip install seaborn
Collecting seaborn
  Downloading seaborn-0.7.1.tar.gz (158kB)
    100% |                                | 163kB 2.1MB/s
Collecting matplotlib (from seaborn)
  Downloading matplotlib-2.0.0.tar.gz (53.2MB)
    99% |                                | 53.2MB 2.9MB/s
.
.
.
  Running setup.py install for functools32 ... done
  Running setup.py install for subprocess32 ... done
  Running setup.py install for matplotlib ... done
  Running setup.py install for pandas ... done
  Running setup.py install for seaborn ... done
Successfully installed cycler-0.10.0 functools32-3.2.3.post2 matplotlib-2.0.0
pandas-0.19.2 pyparsing-2.2.0 seaborn-0.7.1 six-1.10.0

subprocess32-3.2.7
You are using pip version 8.1.2, however version 9.0.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

Note: Remember to install the packages in all the nodes of the cluster.

7. Now we are ready to analyze the data. We created a new note, as shown in Figure 8-15.

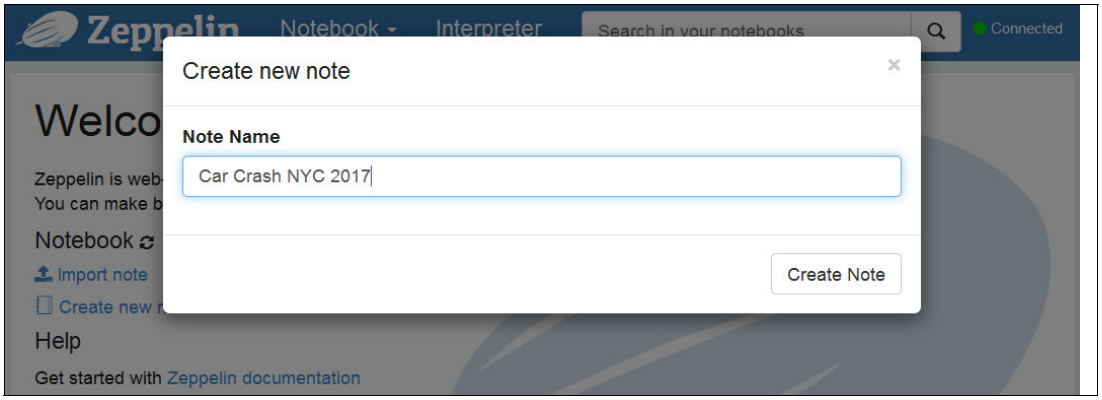


Figure 8-15 Creating note

Now see Figure 8-16, which shows that the note and the first paragraph are ready.

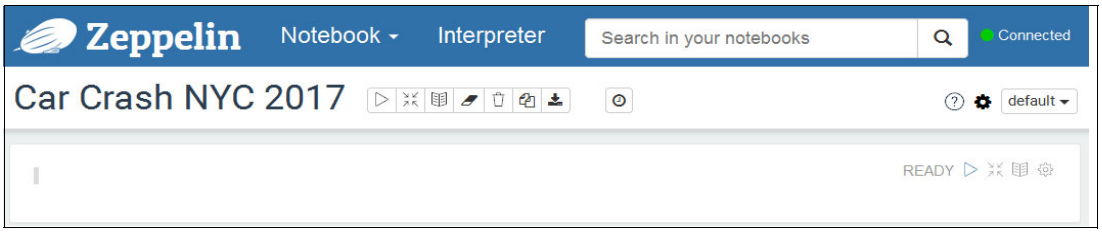


Figure 8-16 Notebook with first paragraph ready

8. In the first paragraph of the note, import the CSV data into a spark data frame. Since the default interpreter is Scala, add the `%pyspark` directive in the beginning of each paragraph to note that we are using PySpark. The import of the data is shown in Example 8-22.

Example 8-22 Importing CSV data into Spark data frame

```
%pyspark
#We are using Python 2.7, importing the function to make the "/" a true
division
from __future__ import division
#importing numpy
import numpy as np

#importing SQLContext
from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)

# adding the PySpark_csv module to SparkContext
sc.addPyFile("https://raw.githubusercontent.com/seahboonsiew/pyspark-csv/master
/pyspark_csv.py")
import pyspark_csv as pycsv
# Importing file we treated as a text file
collisions = sc.textFile("/gpfs/USE_2017_NYPD_Motor_Vehicle_Collisions.csv")
# defining skip_header helper function
def skip_header(idx, iterator):
    if (idx == 0):
        next(iterator)
    return iterator

#First line is the header
collisions_header = collisions.first()
#Header list for column creation
collisions_header_list = collisions_header.split(",")
#Data body without the first line
collisions_body = collisions.mapPartitionsWithIndex(skip_header)

# create Spark DataFrame using pyspark-csv
collisions_df = pycsv.csvToDataFrame(sqlContext, collisions_body, sep=",",
columns=collisions_header_list)
collisions_df.cache()
```

Note: At the end of your paragraph, after you ensure that everything is correct, press **Shift+Enter** or click the **Play** Icon on the upper right corner of your paragraph.

After you submit the job, a progress bar appears and after the computation is done, you see a FINISHED status on the upper right corner of your paragraph, as shown in Figure 8-17.

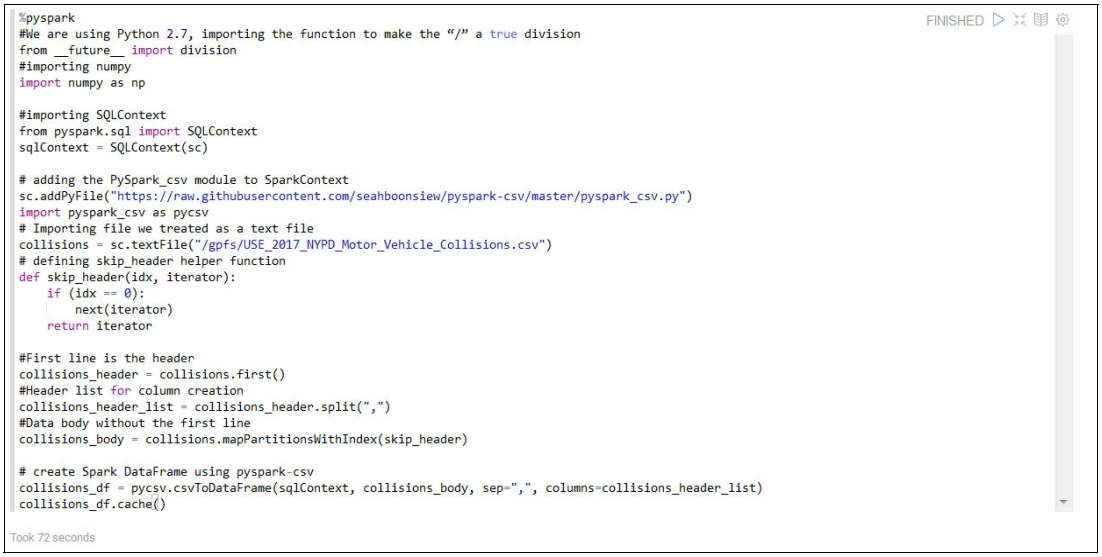


Figure 8-17 Paragraph correctly run

9. Now print the schema of the data frame with the command, as shown in Example 8-23.

Example 8-23 Printing data frame schema

```
%pyspark
collisions_df.printSchema()
```

Figure 8-18 shows the header successfully imported in the data frame.

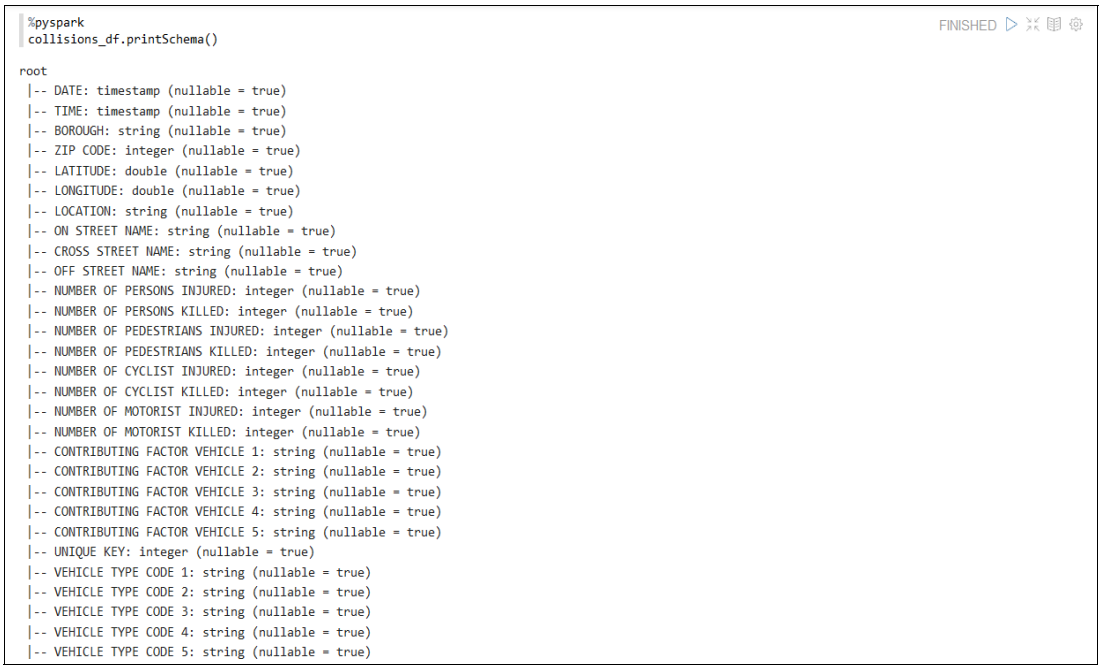


Figure 8-18 Data frame schema

10. We analyze the data using graphs. To perform this task, import the math and graph packages installed at the beginning of this section, as shown in Example 8-24.

Example 8-24 Importing math and graph packages

```
%pyspark
import matplotlib.pyplot as plt; plt.rcdefaults()
import numpy as np
import StringIO
# matplotlib.patches allows us create colored patches, we can use for legends
in plots
import matplotlib.patches as mpatches
# seaborn also builds on matplotlib and adds graphical features and new plot
types
import seaborn as sns
import pandas as pd
```

Example 8-25 shows how to get the data into a smaller pandas data frame using all lines that contain a defined latitude.

Example 8-25 Creating a pandas table to plot graphs

```
%pyspark
#loading all lines that contains a defined Latitude to a Pandas dataframe and
selecting only needed columns
collisions_pd = collisions_df[collisions_df['LATITUDE'] != 0][['LATITUDE',
'LONGITUDE', 'DATE', 'TIME',
                                                              'BOROUGH', 'ON
STREET NAME', 'CROSS STREET NAME',
                                                              'NUMBER OF
PERSONS INJURED', 'NUMBER OF PERSONS KILLED',
                                                              'CONTRIBUTING
FACTOR VEHICLE 1']].toPandas()

#adding headers of the columns on the pandas dataframe
collisions_pd.columns = ['Latitude', 'Longitude', 'Date', 'Time', 'Borough',
'On Street',
                         'Cross Street', 'Persons Injured', 'Persons Killed',
'Contributing Factor']

#divide data set in accidents which are: fatal, non-lethal but with person
damage or none of the two
killed_pd = collisions_pd[collisions_pd['Persons Killed']!=0]
injured_pd = collisions_pd[np.logical_and(collisions_pd['Persons Injured']!=0,
collisions_pd['Persons Killed']==0)]
nothing_pd = collisions_pd[np.logical_and(collisions_pd['Persons Killed']==0,
collisions_pd['Persons Injured']==0)]
```

11. Now plot all collisions using the code shown in Example 8-26.

Example 8-26 Plotting collision graph

```
%pyspark
def show(p):
    img = StringIO.StringIO()
    p.savefig(img, format='svg')
    img.seek(0)
    print "%html <div style='width:900px'>" + img.buf + "</div>"

plt.figure(figsize=(10,7))

#create scatterplots in Black with big dots
plt.scatter(collisions_pd.Longitude, collisions_pd.Latitude, alpha=1, s=3,
color='Black')

#adjust more settings
plt.title('Motor Vehicle Collisions in New York City 2017', size=20)
plt.xlim((-74.26,-73.7))
plt.ylim((40.5,40.92))
plt.xlabel('Longitude',size=10)
plt.ylabel('Latitude',size=10)

show(plt)
```

Figure 8-19 shows that in less than three months, in 2017, almost all areas of New York City already had at least one collision.

```
%pyspark
def show(p):
    img = StringIO.StringIO()
    p.savefig(img, format='svg')
    img.seek(0)
    print "%html <div style='width:900px'>" + img.buf + "</div>"

plt.figure(figsize=(10,7))

#create scatterplots in Black with big dots
plt.scatter(collisions_pd.Longitude, collisions_pd.Latitude, alpha=1, s=3, color='Black')

#adjust more settings
plt.title('Motor Vehicle Collisions in New York City 2017', size=20)
plt.xlim((-74.26,-73.7))
plt.ylim((40.5,40.92))
plt.xlabel('Longitude',size=10)
plt.ylabel('Latitude',size=10)

show(plt)
```

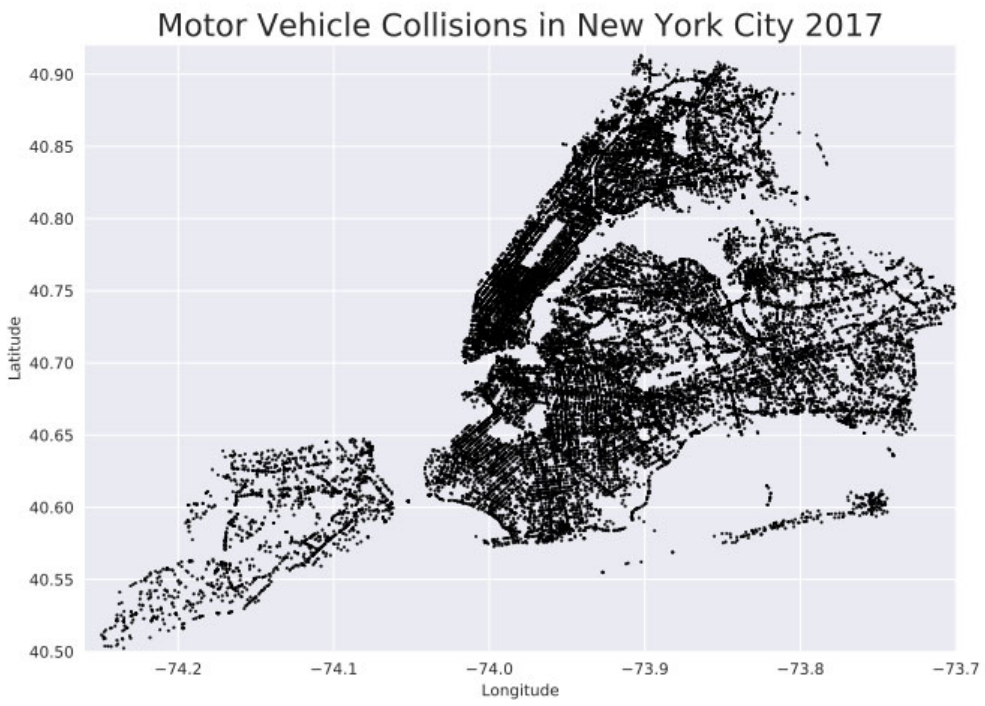


Figure 8-19 All collisions in New York City until 10 March 2017

12. If you want to a deeper analysis of the data, use the separation made in Example 8-25 on page 148 to get an idea of the severity of the accidents. We use the code shown in Example 8-27 to perform this task.

Example 8-27 Plotting the accidents by severity

```
%pyspark
def show(p):
    img = StringIO.StringIO()
    p.savefig(img, format='svg')
    img.seek(0)
    print "%html <div style='width:900px'>" + img.buf + "</div>"

plt.figure(figsize=(10,7))

#create scatterplots
plt.scatter(killed_pd.Longitude, killed_pd.Latitude, alpha=1, s=6,
color='Black')
plt.scatter(injured_pd.Longitude, injured_pd.Latitude, alpha=0.20, s=3,
color='Red')
plt.scatter(nothing_pd.Longitude, nothing_pd.Latitude, alpha=0.05, s=2,
color='Green')

#adjust more settings
plt.title('Motor Vehicle Collisions in New York City 2017', size=20)
plt.xlim((-74.26,-73.7))
plt.ylim((40.5,40.92))
plt.xlabel('Longitude',size=10)
plt.ylabel('Latitude',size=10)

#create legend
red_patch = mpatches.Patch(color='red', label='Personal Injury')
green_patch = mpatches.Patch(color='green', label='Car Damage Only')
black_patch = mpatches.Patch(color='black', label='Fatal')
plt.legend([green_patch, red_patch, black_patch],
           ('Car Damage Only', 'Personal Injury', 'Fatal'),
           loc='upper left', prop={'size':20})

show(plt)
```

Figure 8-20 shows in green all non-injury and non-lethal accidents, and in red all accidents that caused non-lethal personal injury. The black dots are the lethal accidents in 2017.

```
%pyspark
def show(p):
    img = StringIO.StringIO()
    p.savefig(img, format='svg')
    img.seek(0)
    print "%html <div style='width:900px'>" + img.buf + "</div>"

plt.figure(figsize=(10,7))

#create scatterplots
plt.scatter(killed_pd.Longitude, killed_pd.Latitude, alpha=1, s=6, color='Black')
plt.scatter(injured_pd.Longitude, injured_pd.Latitude, alpha=0.20, s=3, color='Red')
plt.scatter(nothing_pd.Longitude, nothing_pd.Latitude, alpha=0.05, s=2, color='Green')

#adjust more settings
plt.title('Motor Vehicle Collisions in New York City 2017', size=20)
plt.xlim((-74.26,-73.7))
plt.ylim((40.5,40.92))
plt.xlabel('Longitude',size=10)
plt.ylabel('Latitude',size=10)

#create legend
red_patch = mpatches.Patch(color='red', label='Personal Injury')
green_patch = mpatches.Patch(color='green', label='Car Damage Only')
black_patch = mpatches.Patch(color='black', label='Fatal')
plt.legend([green_patch, red_patch, black_patch],
           ('Car Damage Only', 'Personal Injury', 'Fatal'),
           loc='upper left', prop={'size':20})

show(plt)
```

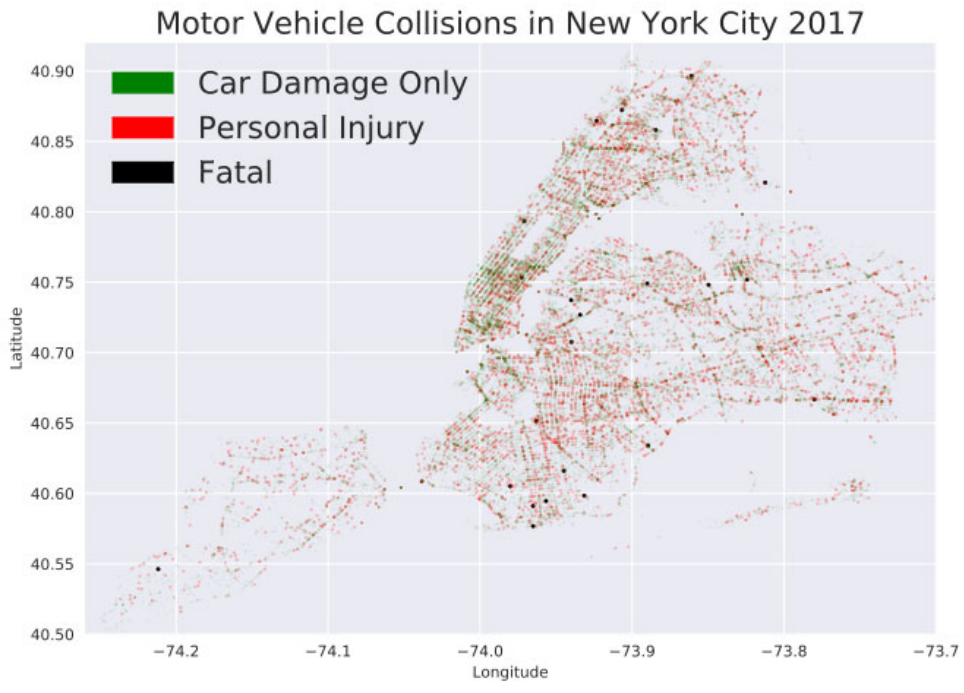


Figure 8-20 Accidents by Severity in New York City until March 10th, 2017

8.3 Spark and streaming data processing use case

To show the use of Spark with streaming data processing, we use a tutorial based on the Zeppelin page.

Complete the following steps:

1. We use Twitter as the source of the data stream so first we create a Twitter application.
2. We do not have any registered application at this moment, so click **Create New App**, as shown in Figure 8-21.

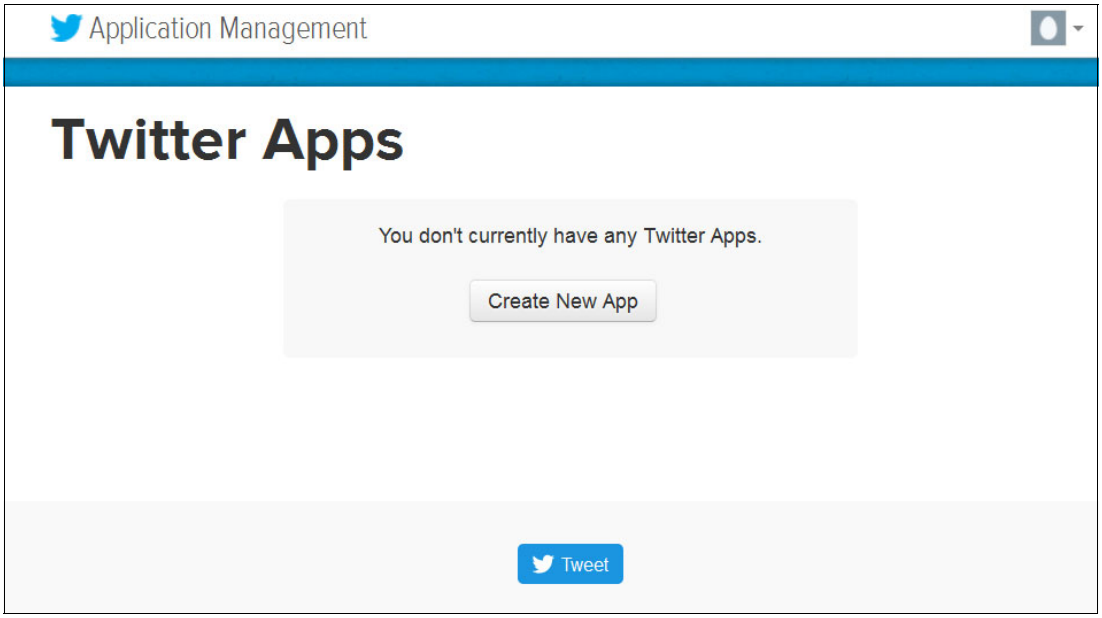


Figure 8-21 Twitter Create New App

3. Complete the fields to create your new application. Because we are only doing a test, put a placeholder for the website field, as shown in Figure 8-22.

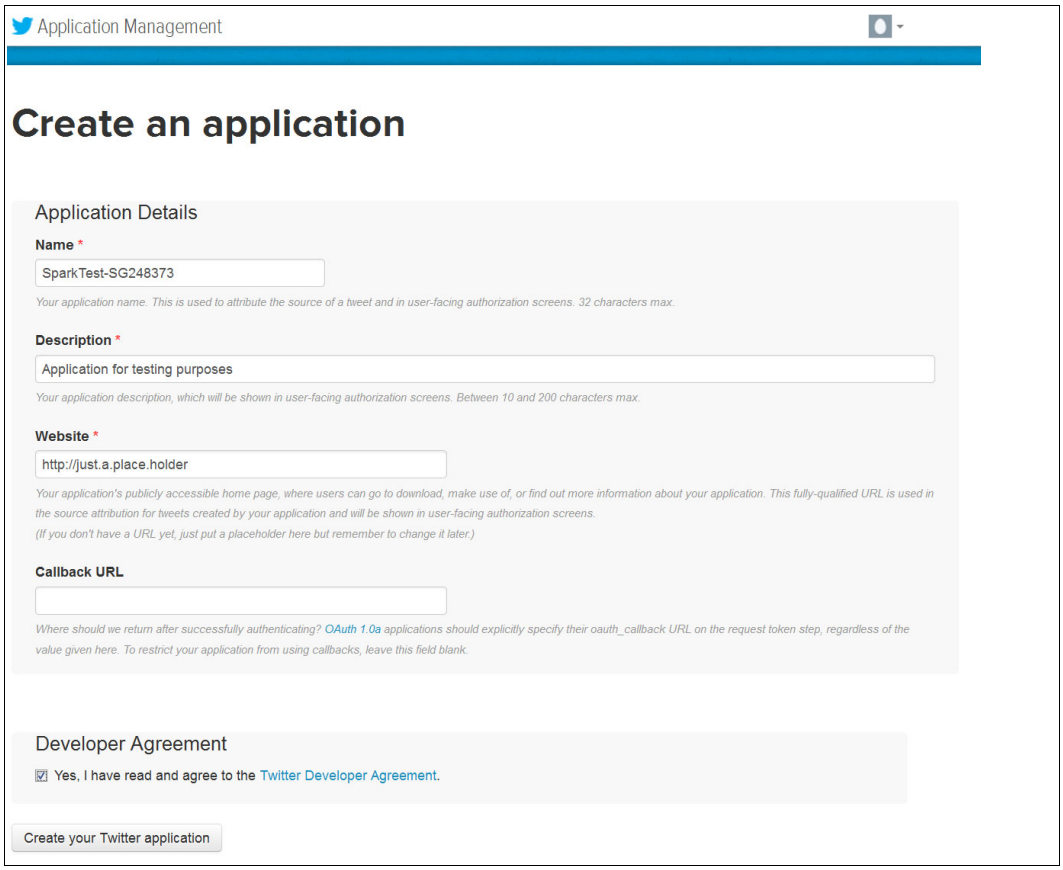


Figure 8-22 Creating Twitter application

Note: You need to have a Twitter account with a mobile phone registered or you get an error. If you receive the error, you must add your phone number to your Twitter account before creating an application.

4. After you create the application, you have access the keys that enable you to read from your Twitter account. In the application screen, click the **Keys and Access Tokens** tab, as shown in Figure 8-23.

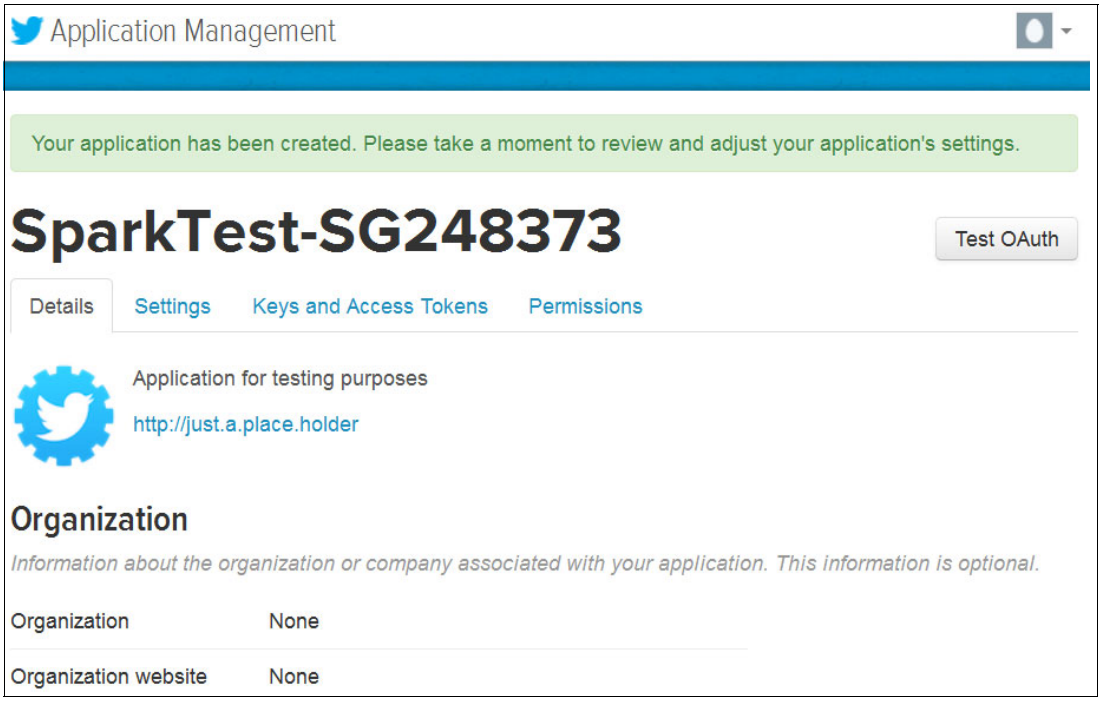


Figure 8-23 Application created successfully

You find the information you need for this task in the tab, as shown in Figure 8-24.

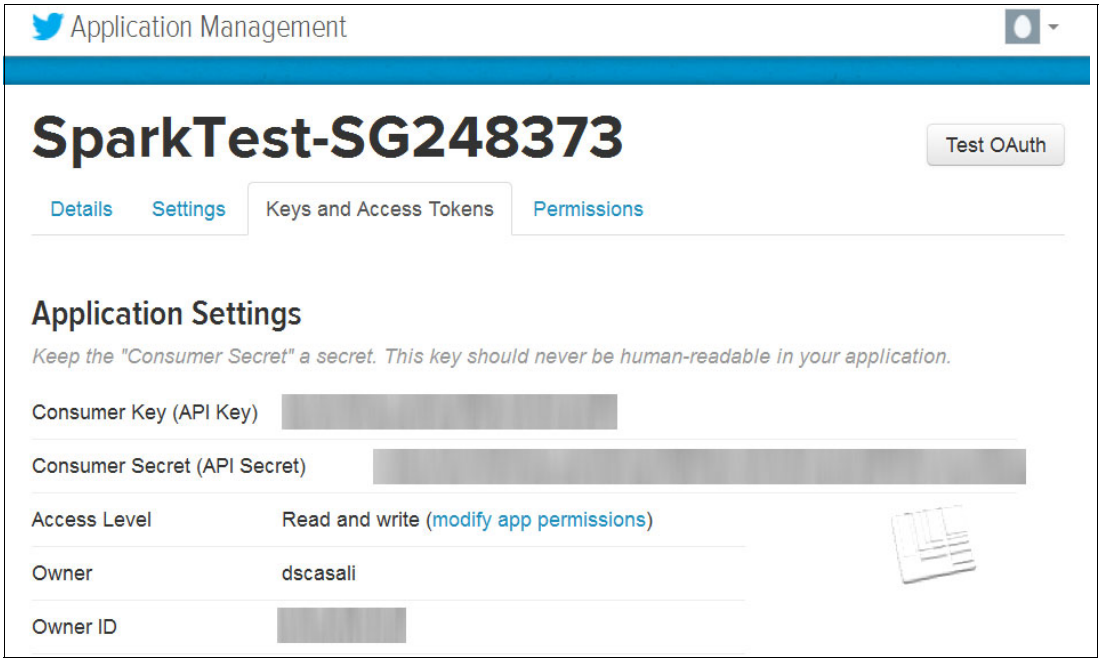


Figure 8-24 Keys and Access Tokens

5. On this page, scroll down and find the place to find the **Create my access token** button, as shown in Figure 8-25.

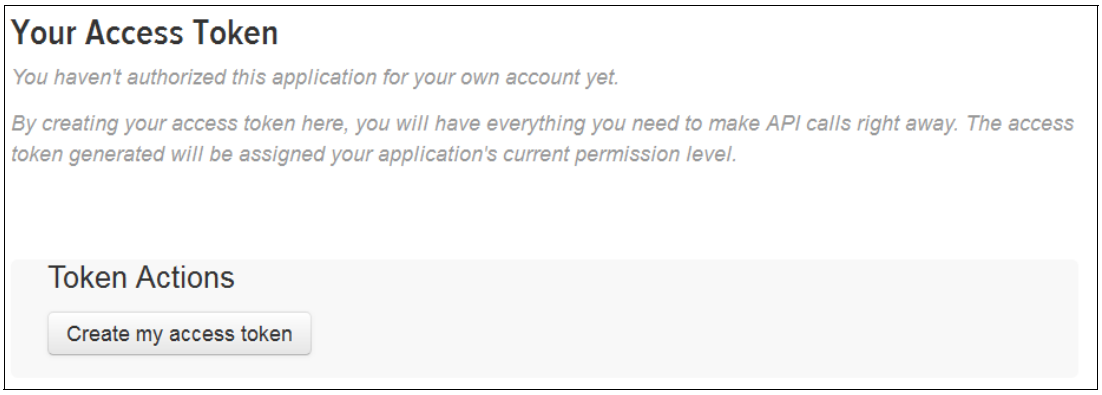


Figure 8-25 create my access token Button

6. Click this button and it generates tokens so that you can access information about your account by way of the API. The page now looks like Figure 8-26.

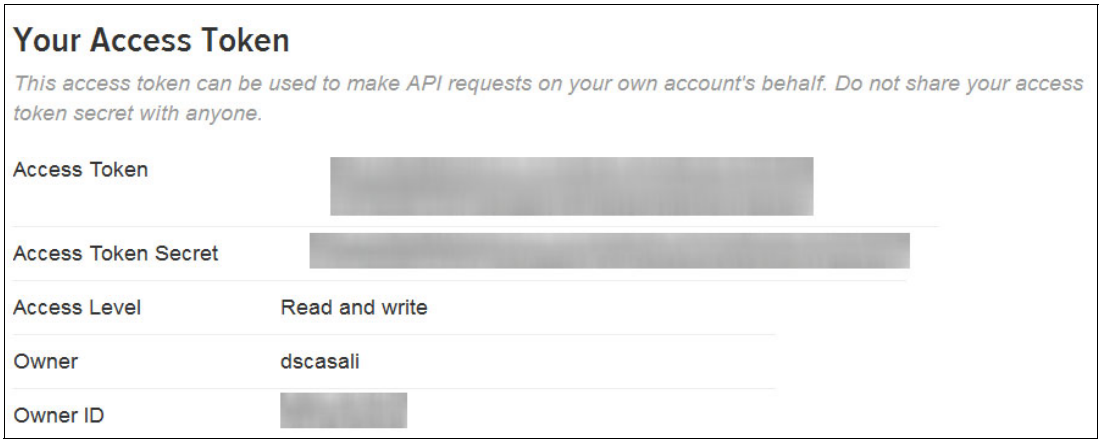


Figure 8-26 Access token page

7. We have a multi-tenant environment, so we created another Spark Instance so that the first user can still use the car crash notebook without any interference from the streaming use case. We again follow the steps performed in 8.1.2, “Creating Spark instances and defining a notebook for a user” on page 137.

We now have two instances running, as you can see in Figure 8-27.

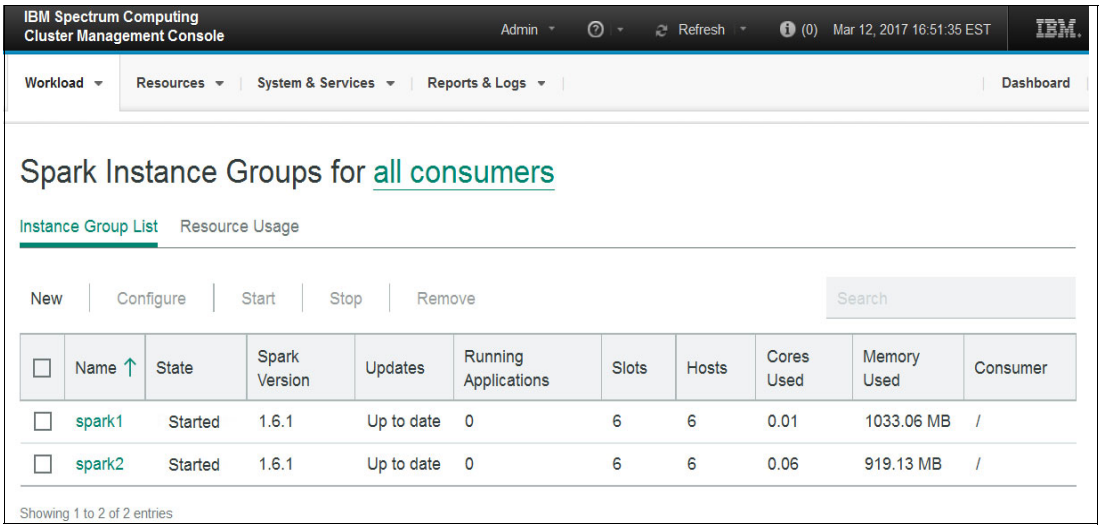


Figure 8-27 Second Spark instance created

> **Tip:** To improve service level agreement controls, you can create and manage different consumers and resource groups by using an advance resource plan. You can find more information about Resource scheduling and Resource sharing and distribution in IBM Knowledge Center.

8. The first paragraph of our new notebook must be the twitter dependency that we need for this exercise. You can see how to load it, as shown in Example 8-28.

Example 8-28 Loading twitter dependencies to the note

```
%dep

z.reset()
z.load("org.apache.spark:spark-streaming-twitter_2.10:1.6.1")
```

> **Note:** If you already ran any other paragraph go to the **Interpreter tab** and restart the Spark Interpreter.

Now we are ready to create the streaming data into a data frame to be analyzed. you need to import all needed libraries, configure the API connection to twitter using the API key, secret, and Token generated in the beginning of this section.

9. To do this, use the code in Example 8-29 and substitute the values for the following entries: `apiKey`, `apiSecret`, `accessToken`, and `accessTokenSecret`.

Example 8-29 Authenticating with Twitter and setting up data frame

```
import org.apache.spark.streaming._
import org.apache.spark.streaming.twitter._
import org.apache.spark.storage.StorageLevel
import scala.io.Source
import scala.collection.mutable.HashMap
import java.io.File
import org.apache.log4j.Logger
import org.apache.log4j.Level
import sys.process.stringSeqToProcess

/** Configures the Oauth Credentials for accessing Twitter */
def configureTwitterCredentials(apiKey: String, apiSecret: String, accessToken:
String, accessTokenSecret: String) {
  val configs = new HashMap[String, String] ++= Seq(
    "apiKey" -> apiKey, "apiSecret" -> apiSecret, "accessToken" -> accessToken,
"accessTokenSecret" -> accessTokenSecret)
  println("Configuring Twitter OAuth")
  configs.foreach{ case(key, value) =>
    if (value.trim.isEmpty) {
      throw new Exception("Error setting authentication - value for " + key + "
not set")
    }
    val fullKey = "twitter4j.oauth." + key.replace("api", "consumer")
    System.setProperty(fullKey, value.trim)
    println("\tProperty " + fullKey + " set as [" + value.trim + "]")
  }
  println()
}

// Configure Twitter credentials
val apiKey = "xxxxxxxxxxxxxxxxxxxx"
val apiSecret = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
val accessToken = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
val accessTokenSecret = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
configureTwitterCredentials(apiKey, apiSecret, accessToken, accessTokenSecret)

import org.apache.spark.streaming.twitter._
val ssc = new StreamingContext(sc, Seconds(2))
val tweets = TwitterUtils.createStream(ssc, None)
val twt = tweets.window(Seconds(60))

case class Tweet(createdAt:Long, text:String)
twt.map(status=>
  Tweet(status.getCreatedAt().getTime()/1000, status.getText())
).foreachRDD(rdd=>
  rdd.toDF().registerTempTable("tweets")
)

twt.print
ssc.start()
```

10. Now we are ready to analyze the data. Following the tutorial from the Apache Zeppelin documentation, now look for the data. Example 8-30 shows a line to look for the word *dog* in the stream, and limit it to 10 matches.

Example 8-30 Look for 10 matches with the word dog

```
%sql select * from tweets where text like '%dog%' limit 10
```

Figure 8-28 shows the output. Whenever you run it, you get a different result because it is streaming data.

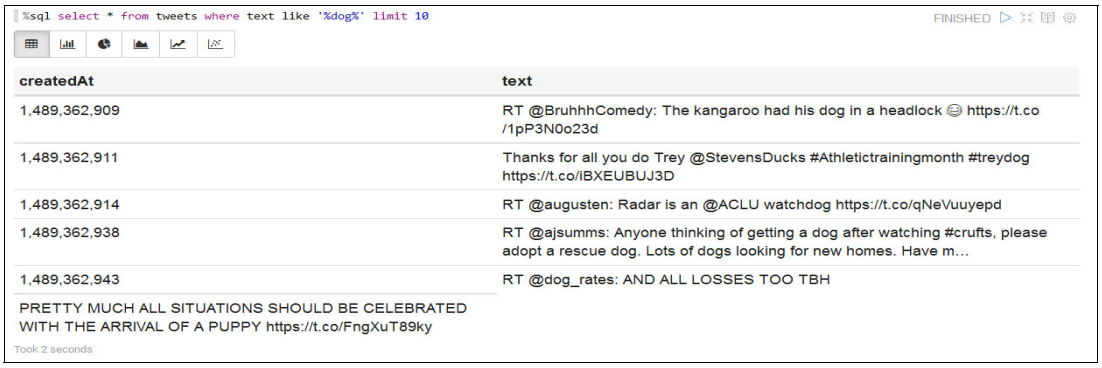


Figure 8-28 Select tweets with the word dog

11. Now define a function to measure sentiment for the word *girl* as an approach shown in Example 8-31.

Example 8-31 Defining sentiment user function

```
def sentiment(s:String) : String = {
    val positive = Array("like", "love", "good", "great", "happy", "cool")
    val negative = Array("hate", "bad", "stupid", "terrible")

    var st = 0;
    val words = s.split(" ")
    positive.foreach(p =>
        words.foreach(w =>
            if(p==w) st = st+1
        )
    )
    negative.foreach(p=>
        words.foreach(w=>
            if(p==w) st = st-1
        )
    )
    if(st>0)
        "positive"
    else if(st<0)
        "negative"
    else
        "neutral"
}

sqlc.udf.register("sentiment", sentiment _)
```

The execution results are shown in Figure 8-29.

```
def sentiment(s:String) : String = {
    val positive = Array("like", "love", "good", "great", "happy", "cool")
    val negative = Array("hate", "bad", "stupid", "terrible")

    var st = 0;

    val words = s.split(" ")
    positive.foreach(p =>
        words.foreach(w =>
            if(p==w) st = st+1
        )
    )

    negative.foreach(p=>
        words.foreach(w=>
            if(p==w) st = st-1
        )
    )
    if(st>0)
        "positive"
    else if(st<0)
        "negative"
    else
        "neutral"
}


sqlc.udf.register("sentiment", sentiment _)
```

FINISHED

```
sentiment: (s: String)String
res66: org.apache.spark.sql.UserDefinedFunction = UserDefinedFunction(<function1>,StringType,List(StringType))
```

Took 0 seconds

Figure 8-29 Defining sentiment function

12. Now you see the results of the sentiment with the command, shown in Example 8-32.

Example 8-32 Checking sentiment

```
%sql select sentiment(text), count(1) from tweets where text like '%girl%'
group by sentiment(text)
```

13. The output is shown in Figure 8-30. We select the bar graphs to better display the data.

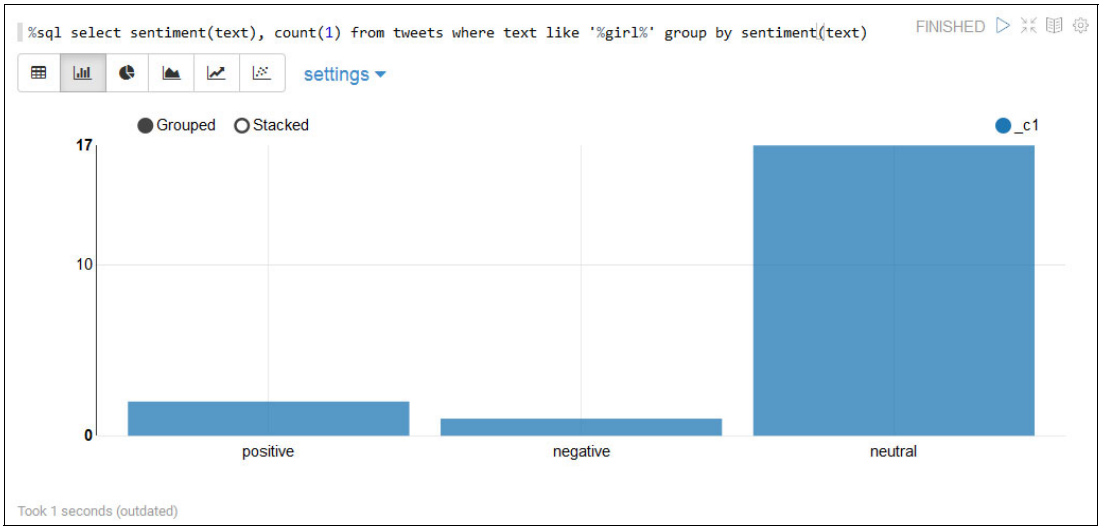


Figure 8-30 Sentiment analysis for the word girl

8.4 Multi-head cluster for improved multi-tenancy

Sharing a large pool of resources started with high-performance computing (HPC) more than 20 years ago. This was done to improve time to results and at the same time maintain needed SLAs across all business (or research) lines using the cluster. Following this trend, IBM Spectrum Computing products can be installed on a single cluster to share resources for different use cases.

8.5 Multi-head possibilities

A multi-head environment can be created in an existing cluster where other products in the IBM Spectrum Computing family are installed. In an environment with multiple products, after you install your first product, you add the other products to your cluster. You can install IBM Spectrum Conductor with Spark either before or after installing IBM Spectrum Symphony, IBM Spectrum LSF Standard Edition for Symphony, or both. However, IBM Spectrum Symphony must always be installed before IBM Spectrum LSF Standard Edition for Symphony is added to the multi-head environment.

Note: This section describes only the scenario where the IBM Conductor with Spark is the first product installed and Symphony is the second, because we already have Conductor installed. See the following website to learn more about Installing or upgrading IBM Spectrum Conductor with Spark with other IBM products with other scenarios.

8.6 Adding IBM Spectrum Symphony to the existing IBM Spectrum Conductor cluster

The first step to add IBM Spectrum Symphony to the cluster is to shut down IBM Spectrum Conductor:

1. To perform this step, follow Example 8-33.

Example 8-33 Shut down IBM Spectrum Conductor

```
[root@st6a04-rh-sym1 ~]# su - egoadmin
Last login: Fri Mar 10 10:36:47 EST 2017 on pts/1
Last failed login: Fri Mar 10 10:42:03 EST 2017 from
st6a04-rh-sym2.pok.stglabs.ibm.com on ssh:notty
There were 2 failed login attempts since the last successful login.
[egoadmin@st6a04-rh-sym1 ~]$ . /opt/ibm/spectrumcomputing/profile.platform
[egoadmin@st6a04-rh-sym1 ~]$ egosh user logon
user account: Admin
password:
Logged on successfully
[egoadmin@st6a04-rh-sym1 ~]$ egoshutdown.sh
Service <GPFSmonitor> has been stopped successfully
Service <elk-shipper> has been stopped successfully
Service <elk-indexer> has been stopped successfully
Service <elk-indexcleanup> has been stopped successfully
Service <elk-elasticsearch> has been stopped successfully
Service <purger> has been stopped successfully
Service <plc> has been stopped successfully
```

```
Service <derbydb> has been stopped successfully
Service <WEBGUI> has been stopped successfully
Service <spark1-sparkss> has been stopped successfully
Service <spark1-sparkms-notebook> has been stopped successfully
Service <spark1-sparkms-batch> has been stopped successfully
Service <spark2-sparkss> has been stopped successfully
Service <spark2-sparkms-notebook> has been stopped successfully
Service <spark2-sparkms-batch> has been stopped successfully
Service <SparkCleanup> has been stopped successfully
Service <ascd> has been stopped successfully
Service <EGOYARN> has been stopped successfully
Service <REST> has been stopped successfully
Service <ExecProxy> has been stopped successfully
Service <ServiceDirector> has been stopped successfully
Service <RS> has been stopped successfully
Service <WebServiceGateway> has been stopped successfully
Service <spark2-Zeppelin-0-5-6-9> has been stopped successfully
Shut down LIM on <st6a04-rh-sym2.pok.stglabs.ibm.com> ...... done
Shut down LIM on <st6a04-rh-sym3.pok.stglabs.ibm.com> ...... done
Shut down LIM on <st6a04-rh-sym4.pok.stglabs.ibm.com> ...... done
Shut down LIM on <st6a04-rh-sym5.pok.stglabs.ibm.com> ...... done
Shut down LIM on <st6a04-rh-sym6.pok.stglabs.ibm.com> ...... done
Shut down LIM on <st6a04-rh-sym1.pok.stglabs.ibm.com> ...... done
[egoadmin@st6a04-rh-sym1 ~]
```

2. Then create the *$EGO_TOP*/perf/3.4/etc directory. *$EGO_TOP* is the installation directory, in our case /opt/ibm/spectrumcomputing, as shown in Example 8-34.

Example 8-34 Creating perf/3.4/etc directory under $EGO_TOP

```
[egoadmin@st6a04-rh-sym1 ~]$ mkdir -p /opt/ibm/spectrumcomputing/perf/3.4/etc
[egoadmin@st6a04-rh-sym1 ~]$
```

3. Now install IBM Spectrum Symphony with the exact same configurations used to install IBM Spectrum Conductor as shown in Example 8-5 on page 129 and Example 8-6 on page 130, and add another configuration, **"***SIMPLIFIEDWEM=N***"**, as you can see in Example 8-35.

Example 8-35 Setting configurations

```
[root@st6a04-rh-sym1 sym]# export DISABLESSL=Y
[root@st6a04-rh-sym1 sym]# export JAVA_HOME=/usr/bin/
[root@st6a04-rh-sym1 sym]# export DERBY_DB_HOST=st6a04-rh-sym1
[root@st6a04-rh-sym1 sym]# export HADOOP_YARN_HOME=/opt/hadoop-2.7.3
[root@st6a04-rh-sym1 sym]# export CLUSTERNAME=itsocluster
[root@st6a04-rh-sym1 sym]# export CLUSTERADMIN=egoadmin
[root@st6a04-rh-sym1 sym]# export SIMPLIFIEDWEM=N
[root@st6a04-rh-sym1 sym]#
```

4. Now initialize IBM Spectrum Symphony installation as shown in Example 8-36. If you accept the License Agreement, respond 1.

Example 8-36 Initializing IBM Spectrum Symphony installation

```
[root@st6a04-rh-sym1 sym]# ./sym-7.1.2.0_ppc64le.bin
Extracting files... done.

IBM Spectrum Symphony License Agreement
International Program License Agreement

Part 1 - General Terms
.
.
.
Press Enter to continue viewing the license agreement, or
enter "1" to accept the agreement, "2" to decline it, "3"
to print it, "4" to read non-IBM terms, or "99" to go back
to the previous screen.
1
The following packages are not required because the installer detected an equal
or higher version already installed:
egocore-3.4.0.0.ppc64le.rpm
egojre-1.8.0.3.ppc64le.rpm
egowlp-8.5.5.9.noarch.rpm
egorest-3.4.0.0.noarch.rpm
egomgmt-3.4.0.0.noarch.rpm
egoyarn-3.4.0.0.ppc64le.rpm
egogpfsmonitor-3.4.0.0.noarch.rpm
The following commands will install the .rpm packages on the host.
rpm --prefix /opt/ibm/spectrumcomputing -ivh  egospark-3.4.0.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  soamcore-7.1.2.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  soammgmt-7.1.2.0.noarch.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  soammrcore-7.1.2.0.ppc64le.rpm
rpm --prefix /opt/ibm/spectrumcomputing -ivh  soammrmgmt-7.1.2.0.noarch.rpm
Do you want to continue?(Y/N)Y
Start install...
Preparing...                          ################################# [100%]

Updating / installing...
   1:egospark-3.4.0.0-408454          ################################# [100%]

Successfully installed the Spark on EGO integration package.
.
.
.
Updating / installing...
   1:soammrmgmt-7.1.2.0-408454        ################################# [100%]

Successfully installed the MapReduce management package for IBM Spectrum
Symphony.
[root@st6a04-rh-sym1 sym]#
```

5. Now set the entitlement for the use of IBM Spectrum Symphony Advanced Edition, as shown in Example 8-37.

Example 8-37 Setting entitlement for IBM Spectrum Symphony Advanced

```
[root@st6a04-rh-sym1 ~]# su - egoadmin
Last login: Mon Mar 13 16:38:42 EDT 2017 on pts/2
[egoadmin@st6a04-rh-sym1 ~]$ . /opt/ibm/spectrumcomputing/profile.platform
[egoadmin@st6a04-rh-sym1 ~]$ egoconfig setentitlement
/gpfs/pkgs/sym/sym_adv_entitlement.dat
To complete the installation, start the cluster by using the egosh ego start
command. If the cluster is already started, stop all services (egosh service
stop all), shut down the cluster (egosh ego shutdown all), and restart the
cluster (egosh ego start all).
Then, log on to the management console as cluster administrator and run the
cluster configuration wizard when prompted. You must complete this step to
ensure correct cluster configurations for IBM Spectrum Symphony and IBM
Spectrum Conductor with Spark.
The system successfully set the specified entitlement.
[egoadmin@st6a04-rh-sym1 ~]$
```

6. The last step before starting the cluster is to copy *$EGO_TOP*/gui/3.4/lib/* directory to the *$EGO_TOP*/gui/3.5/lib, as shown in Example 8-38.

Example 8-38 Copying libraries to the new library path

```
[egoadmin@st6a04-rh-sym1 ~]$ cp /opt/ibm/spectrumcomputing/gui/3.4/lib/*
/opt/ibm/spectrumcomputing/gui/3.5/lib/
[egoadmin@st6a04-rh-sym1 ~]$
```

7. Now restart the cluster. We use egostartup.sh, as shown in Example 8-39.

Example 8-39 Starting up the cluster

```
[egoadmin@st6a04-rh-sym1 ~]$ egostartup.sh
Do you really want to start up LIM on all hosts ? [y/n]y
Start up LIM on <st6a04-rh-sym1.pok.stglabs.ibm.com> ...... done
Start up LIM on <st6a04-rh-sym2.pok.stglabs.ibm.com> ...... done
Start up LIM on <st6a04-rh-sym3.pok.stglabs.ibm.com> ...... done
Start up LIM on <st6a04-rh-sym4.pok.stglabs.ibm.com> ...... done
Start up LIM on <st6a04-rh-sym5.pok.stglabs.ibm.com> ...... done
Start up LIM on <st6a04-rh-sym6.pok.stglabs.ibm.com> ...... done
```

8. You can now access the graphical user interface (GUI). After login, a pop-up window prompts you to reconfigure the cluster, as shown in Figure 8-31. Click **Continue**.

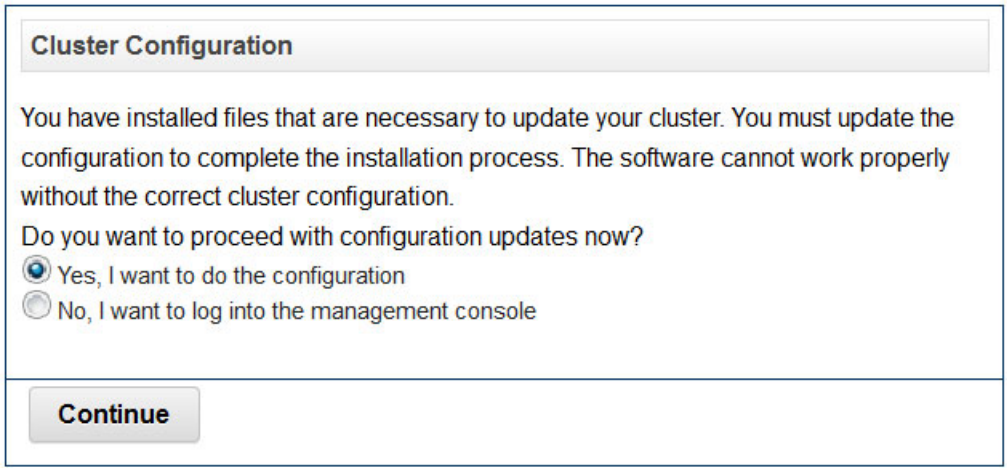


Figure 8-31 Cluster Configuration pane

9. The Cluster Configuration wizard starts, as shown in Figure 8-32. These steps do not remove your existing configuration or affect your workload. Hence the cluster continues running while you work through these steps. Click **Continue** to start the wizard.

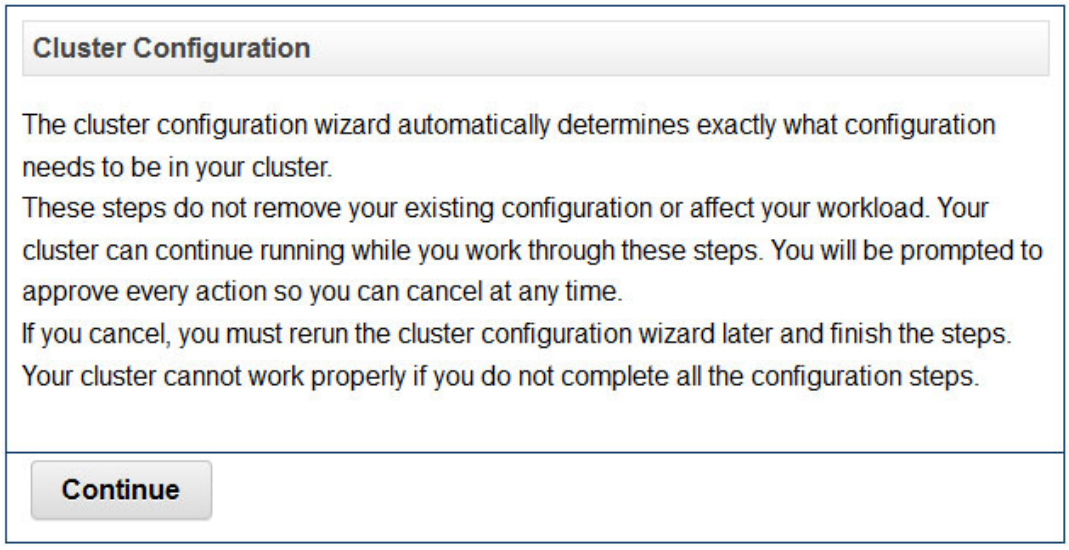


Figure 8-32 First Wizard Panel

10. The first step adds a new Resource Groups for IBM Spectrum Symphony, as shown in Figure 8-33. Click **Create**.

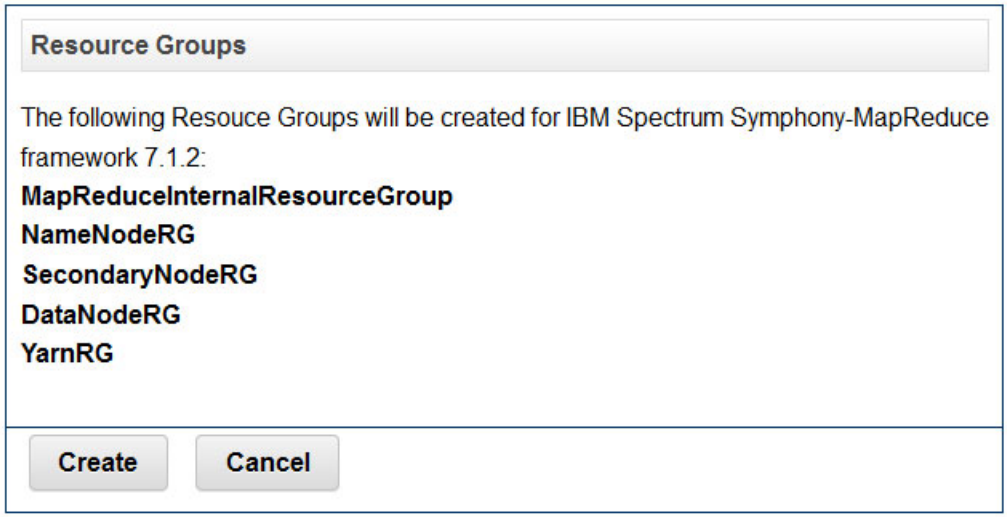


Figure 8-33 Creating Resource Groups for IBM Spectrum Symphony

11. The wizard creates all new Resource Groups and issues a message stating that the YarnRG already exists, as shown in Figure 8-34. Click **Continue**.

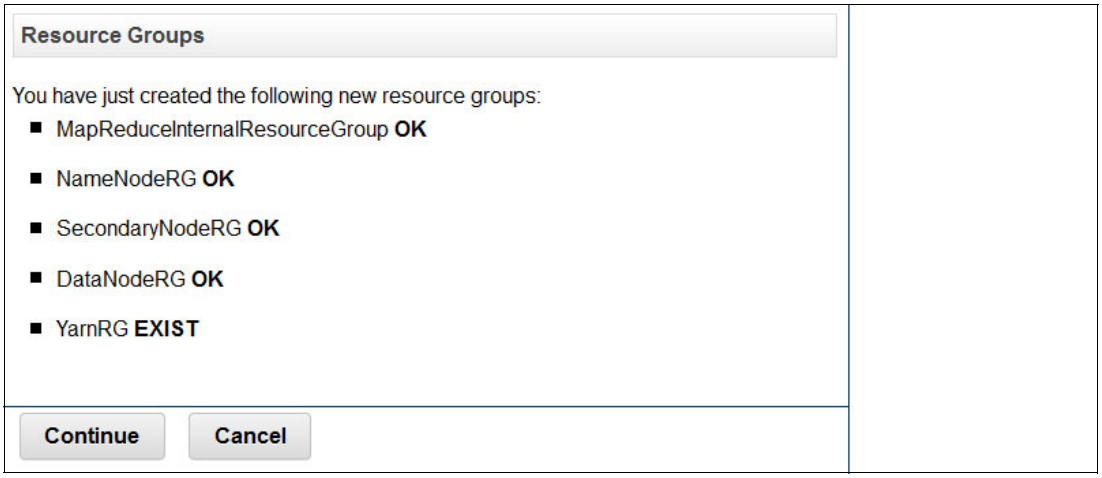


Figure 8-34 New Resource Groups created

12. A list of needed consumers appear, as shown in Figure 8-35. Click **Create**.

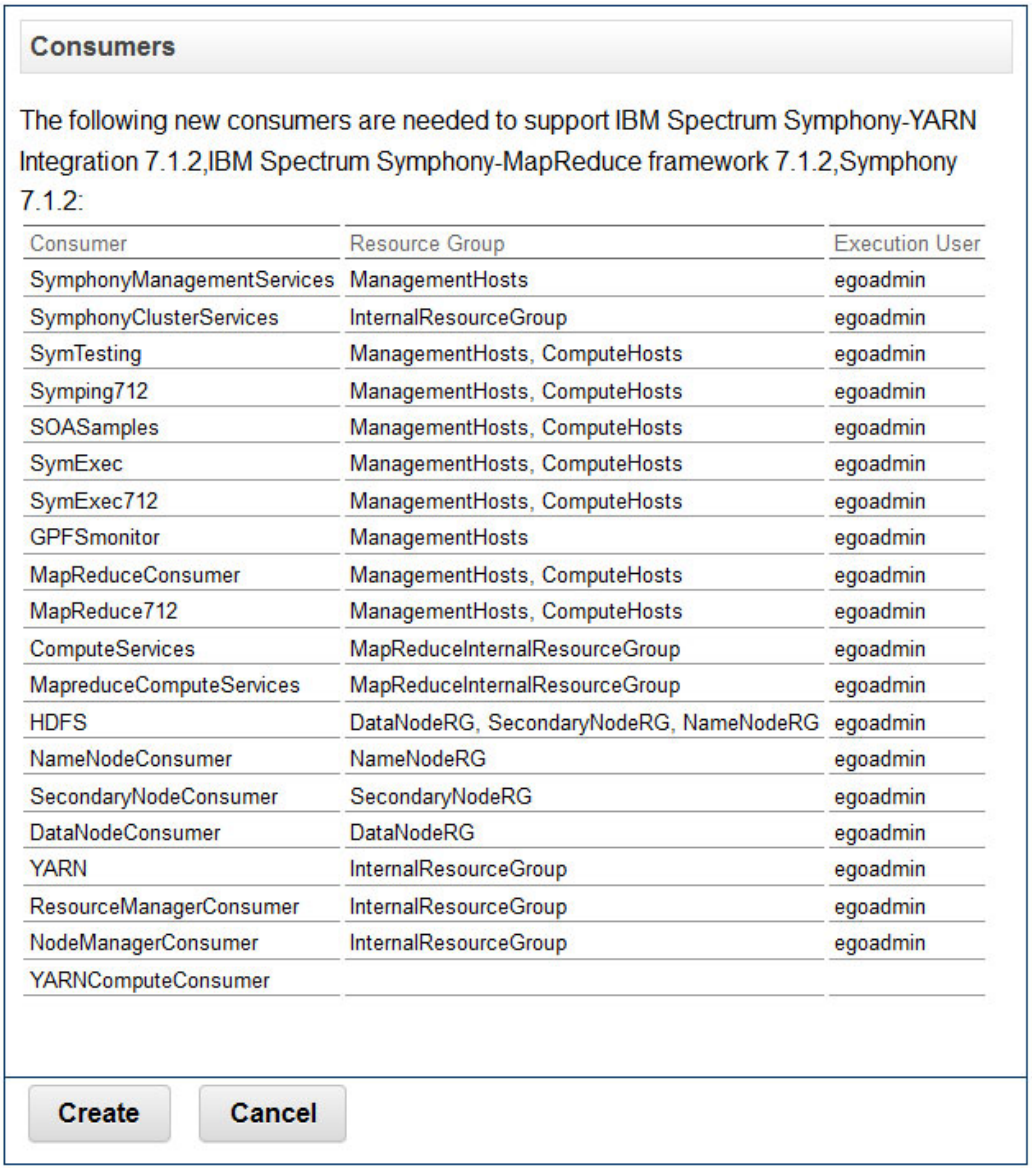


Figure 8-35 New consumers to be created

13. The list of newly created consumers is shown (Figure 8-36). Consumers that existed because of the previous installation of IBM Spectrum Conductor appear as `EXIST` and no actions are taken on that consumer. Click **Continue**.

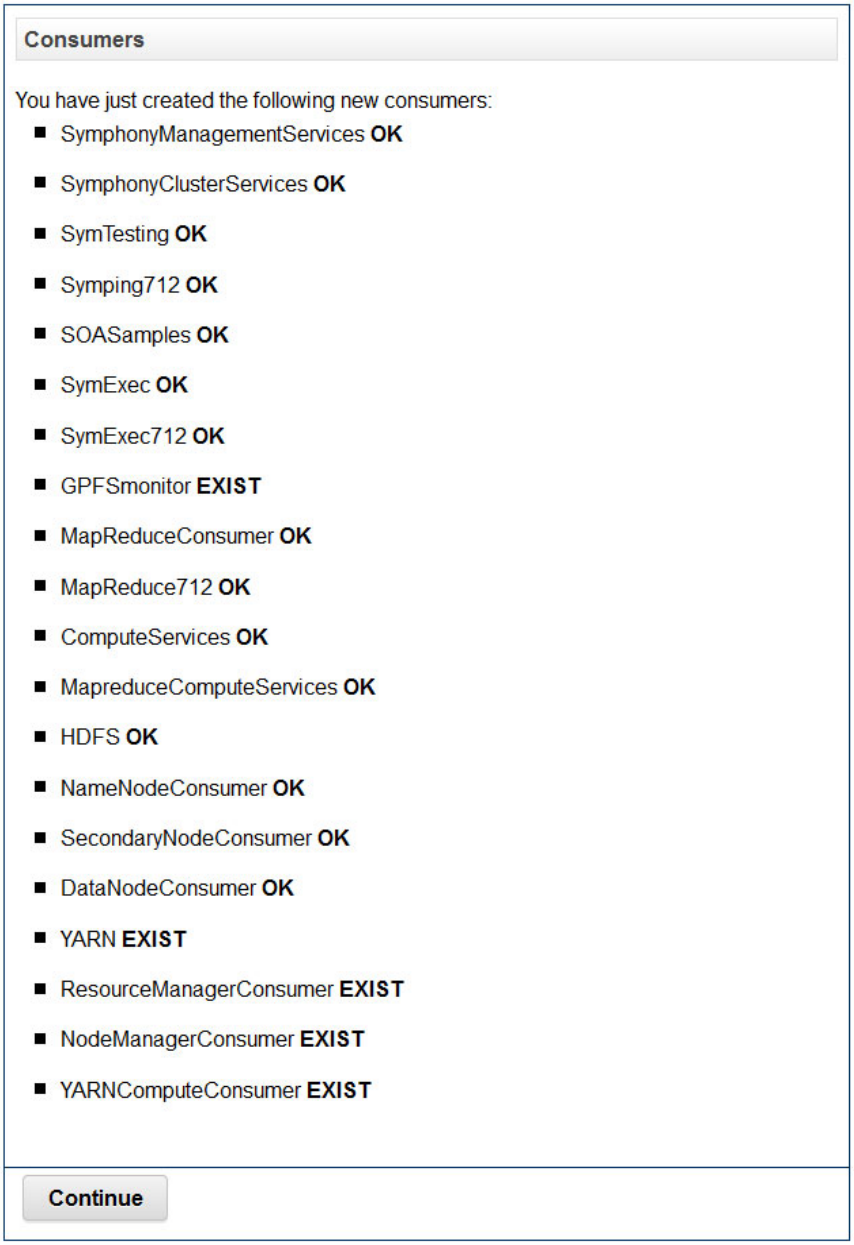


Figure 8-36 Consumers created.

14. The Wizard shows two new Clusters Services that IBM Spectrum Symphony needs, as shown in Figure 8-37. Click **Create**.

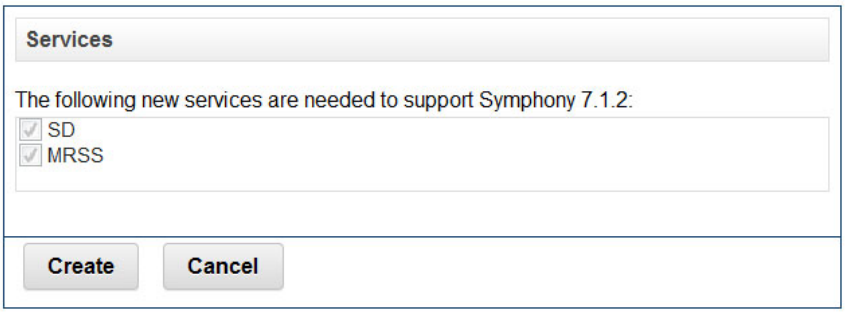


Figure 8-37 New Services needed.

15. The Cluster shows the newly defined Services, as shown in Figure 8-38. Click **Continue**.

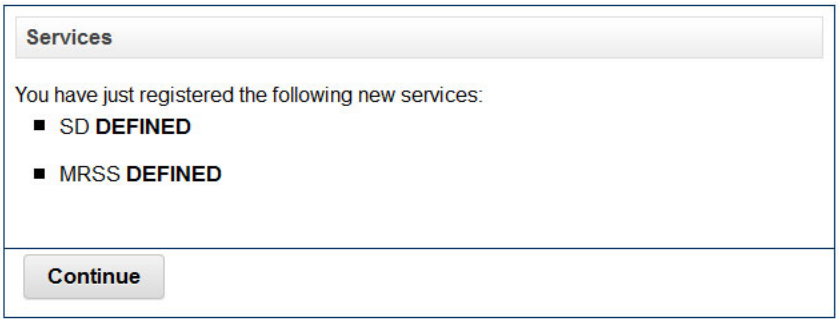


Figure 8-38 Defined Services

16. You can see that some Cluster Services need to be restarted to support the Symphony MapReduce framework, as shown in Figure 8-39. Click **Restart**.

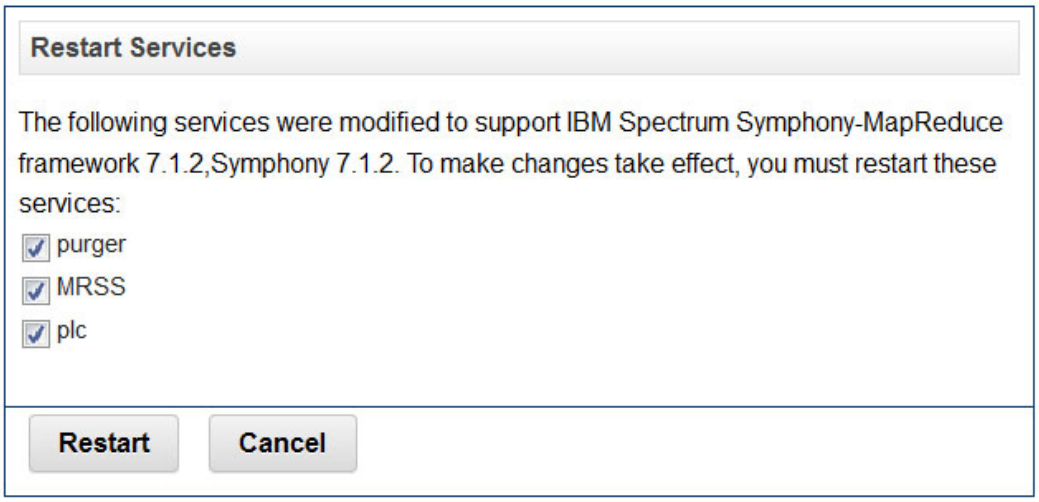


Figure 8-39 Services to be restarted

17. The signal to restart the services is sent, as shown in Figure 8-40. Click **Continue**.

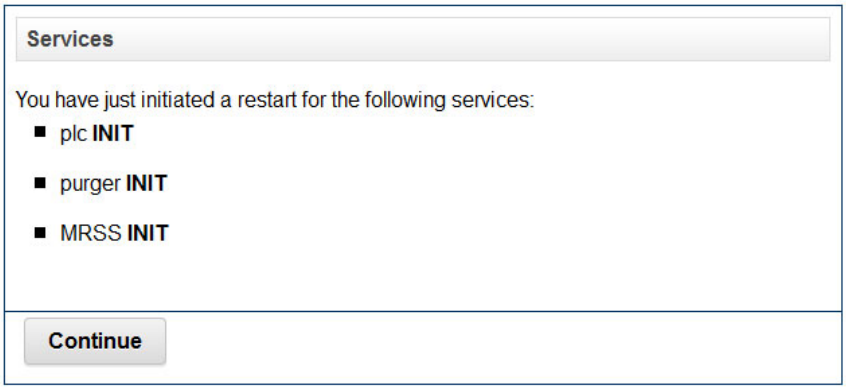


Figure 8-40 Services being restarted

18. You go to the last step of the Wizard, which is the console Restart. The processing can take some time. If you want, you can access the logs of the update by clicking the **View upgrade log** button, as shown in Figure 8-41, to end the Wizard and restart the console.

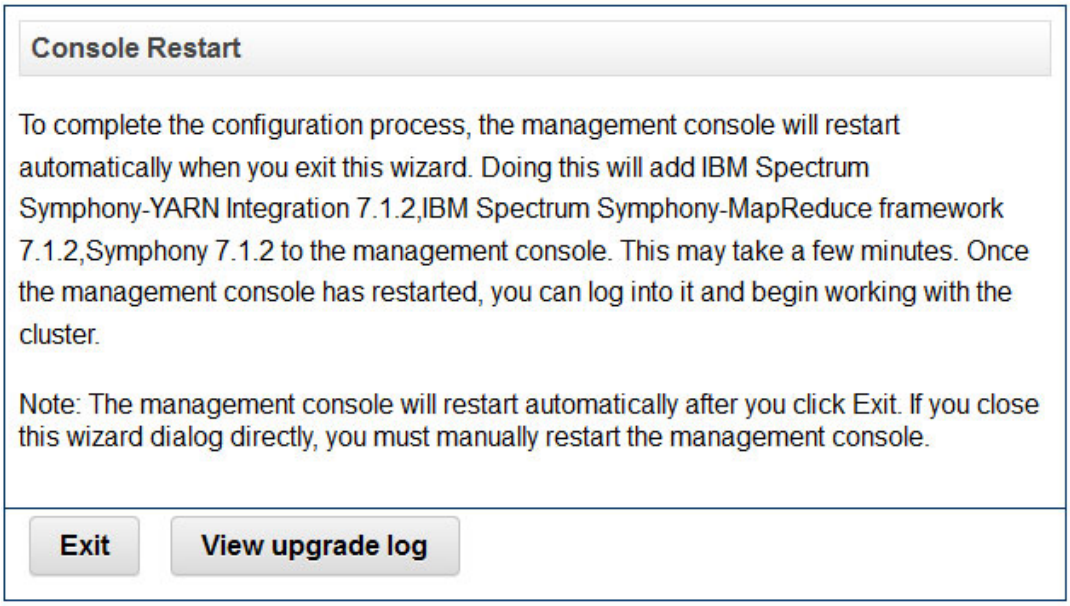


Figure 8-41 Console Restart

19. After the Restart, you can log on again in the console and shown in Figure 8-42. Now the Workload tab has more workload options when compared to Figure 8-4 on page 137 when we had only Conductor for Spark.

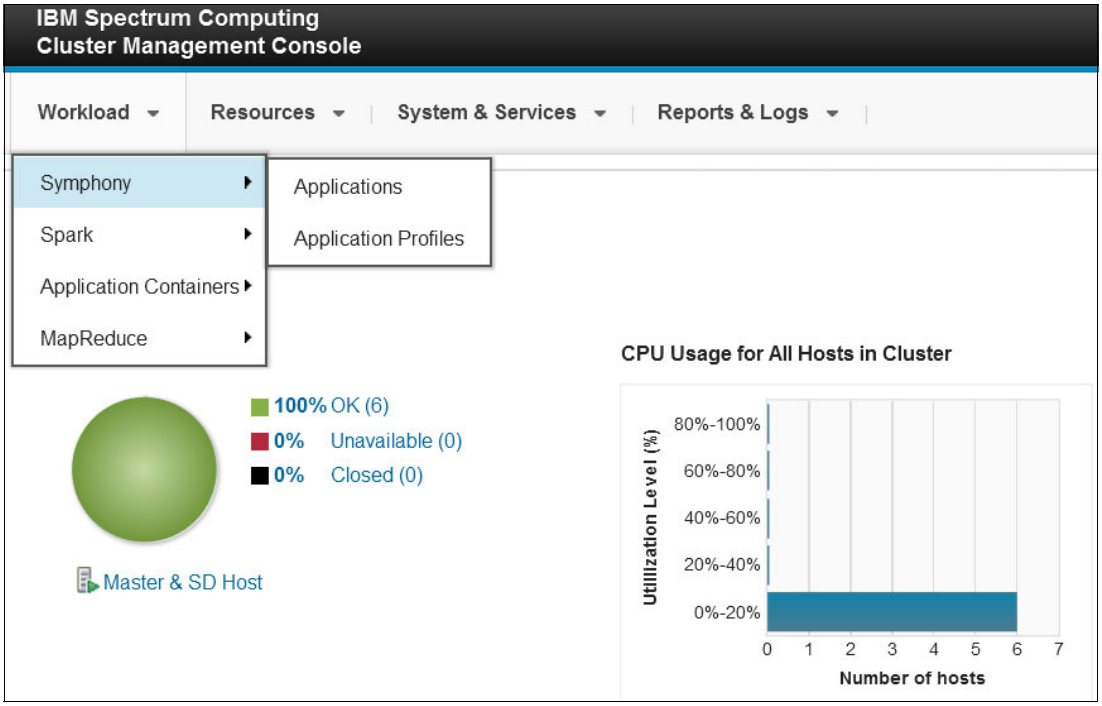


Figure 8-42 Dashboard with new workloads available including MapReduce

Understanding how to configure MapReduce and bind it to IBM Spectrum Scale is outside the scope of this book. You can see how to perform this task in Chapter 4, *IBM Big Data Implementation on an IBM High Performance Computing Cluster* of the IBM Redbooks publication, *IBM Platform Computing Solutions Reference Architectures and Best Practices*, SG24-8169.

8.7 Adding MongoDB to the IBM Spectrum Conductor

IBM Spectrum Conductor is able to manage different applications, and you can use this feature to install and control a MongoDB with scalable number of Shards on POWER Linux machines. Many different templates can be found at the jazz.net hub.

You can use these templates as a base for installing and controlling applications using containers or not. The templates use a YAML file to define the services needed and scripts to use given a defined task that needs to be performed (such as deploy, start, stop, scale, and so on).

Complete the following steps:

1. In order to use MongoDB 3.5, download and alter the template.

   We followed the previous step to comply with new MongoDB configuration replica initialization and to deploy correctly on ppc64le servers.

   After the sample has been changed, it is published in the Bluemix DevOps Services so you can download it.

As shown in the template sample published, you also need a MongoDB compiled for POWER. You are using an Open Development Version. However, Enterprise versions should be used for production.

Note: We are using a development package as we are using it for demonstration purposes. At the time this publication was written, MongoDB 3.5 is not intended for production use. Enterprise versions should be used for production with the application template sample used here and changed to fit your environment needs.

2. Now follow the step-by-step readme file of the Project present to generate the service package that is used to deploy MongoDB in the environment. The steps are shown in Example 8-40.

Example 8-40 Creating the service package

```
[root@st6a04-rh-sym1 template]# wget -O template.zip
https://hub.jazz.net/git/dancasali/sample-MongoDBppc64le/archive?revstr=master
--2017-03-18 15:58:22--
https://hub.jazz.net/git/dancasali/sample-MongoDBppc64le/archive?revstr=master
Resolving hub.jazz.net (hub.jazz.net)... 184.173.83.37
Connecting to hub.jazz.net (hub.jazz.net)|184.173.83.37|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13994 (14K) [application/force-download]
Saving to: ‘template.zip’

100%[=======================================>] 13,994      --.-K/s   in
0.07s

2017-03-18 15:58:22 (190 KB/s) - ‘template.zip’ saved [13994/13994]

[root@st6a04-rh-sym1 template]# unzip template.zip
Archive:  template.zip
  inflating: .gitignore
  inflating: License.txt
  inflating: MongoDB_3.5_Sharded.yaml
  inflating: MongoDB_3.5_Sharded_scripts/checkdecomm.sh
  inflating: MongoDB_3.5_Sharded_scripts/common.inc
  inflating: MongoDB_3.5_Sharded_scripts/default/check.js
  inflating: MongoDB_3.5_Sharded_scripts/default/config_status.js
  inflating: MongoDB_3.5_Sharded_scripts/default/configdb.cfg.template
  inflating: MongoDB_3.5_Sharded_scripts/default/replica_add.js.template
  inflating: MongoDB_3.5_Sharded_scripts/default/replica_config_status.js
  inflating: MongoDB_3.5_Sharded_scripts/default/replica_init.js
  inflating: MongoDB_3.5_Sharded_scripts/default/replica_remove.js.template
  inflating: MongoDB_3.5_Sharded_scripts/default/replica_stepdownprimary.js
  inflating: MongoDB_3.5_Sharded_scripts/default/shard.cfg.template
  inflating: MongoDB_3.5_Sharded_scripts/default/shard_add.js.template
  inflating: MongoDB_3.5_Sharded_scripts/default/shard_remove.js.template
  inflating: MongoDB_3.5_Sharded_scripts/default/shard_status.js
  inflating: MongoDB_3.5_Sharded_scripts/deploy.sh
  inflating: MongoDB_3.5_Sharded_scripts/monitorshard.sh
  inflating: MongoDB_3.5_Sharded_scripts/startconfig.sh
  inflating: MongoDB_3.5_Sharded_scripts/startdecomm.sh
  inflating: MongoDB_3.5_Sharded_scripts/startrouter.sh
  inflating: MongoDB_3.5_Sharded_scripts/startshard.sh
```

```
  inflating: MongoDB_3.5_Sharded_scripts/stopconfig.sh
  inflating: MongoDB_3.5_Sharded_scripts/stoprouter.sh
  inflating: MongoDB_3.5_Sharded_scripts/stopshard.sh
  inflating: MongoDB_3.5_Sharded_scripts/undeploy.sh
  inflating: README.md
  inflating: manifest.txt
[root@st6a04-rh-sym1 template]# mkdir -p mongodbasc/ascscripts
[root@st6a04-rh-sym1 template]# cp -r MongoDB_3.5_Sharded_ppc64le_scripts/*
mongodbasc/ascscripts/
[root@st6a04-rh-sym1 template]# mkdir mongodbasc/package
[root@st6a04-rh-sym1 template]# cp
../mongodb-linux-ppc64le-enterprise-rhel71-3.5.4-25.tgz mongodbasc/package/
[root@st6a04-rh-sym1 template]# cp
/opt/ibm/spectrumcomputing/ascd/conf/samples/deployment.xml  mongodbasc/
[root@st6a04-rh-sym1 template]# cd mongodbasc
[root@st6a04-rh-sym1 mongodbasc]# ls
ascscripts  deployment.xml  package
[root@st6a04-rh-sym1 mongodbasc]# tar -czvf mongodbasc.tar.gz *
ascscripts/
ascscripts/stoprouter.sh
ascscripts/stopconfig.sh
ascscripts/checkdecomm.sh
ascscripts/startconfig.sh
ascscripts/undeploy.sh
ascscripts/startrouter.sh
ascscripts/stopshard.sh
ascscripts/startdecomm.sh
ascscripts/deploy.sh
ascscripts/startshard.sh
ascscripts/monitorshard.sh
ascscripts/default/
ascscripts/default/replica_remove.js.template
ascscripts/default/replica_init.js
ascscripts/default/check.js
ascscripts/default/shard_add.js.template
ascscripts/default/shard_remove.js.template
ascscripts/default/replica_stepdownprimary.js
ascscripts/default/shard.cfg.template
ascscripts/default/shard_status.js
ascscripts/default/configdb.cfg.template
ascscripts/default/config_status.js
ascscripts/default/replica_add.js.template
ascscripts/common.inc
deployment.xml
package/
package/mongodb-linux-ppc64le-enterprise-rhel71-3.5.4-25.tgz
[root@st6a04-rh-sym1 mongodbasc]#
```

MongoDB has a dependency for the `net-snmp-agent-libs` package. Example 8-41 shows the installation of this dependency.

Example 8-41 Installing net-snmp-agent-libs

```
[root@st6a04-rh-sym1 media]# yum install -y net-snmp-agent-libs
Loaded plugins: product-id, search-disabled-repos, subscription-manager
```

```
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
No packages marked for update
.
.
.
Installed:
  net-snmp-agent-libs.ppc64le 1:5.7.2-24.el7

Dependency Installed:
  lm_sensors-libs.ppc64le 0:3.3.4-11.el7  net-snmp-libs.ppc64le 1:5.7.2-24.el7
```

3. Now retrieve the `mongodbasc.tar.gz` service package created in Example 8-40 on page 172 to your desktop where you already have `MongoDB_3.5_Sharded_ppc64le.yaml` file downloaded. Now you are ready to register your application. Click **Workload** → **Application Containers** → **Application Instances**, as shown in Figure 8-43.

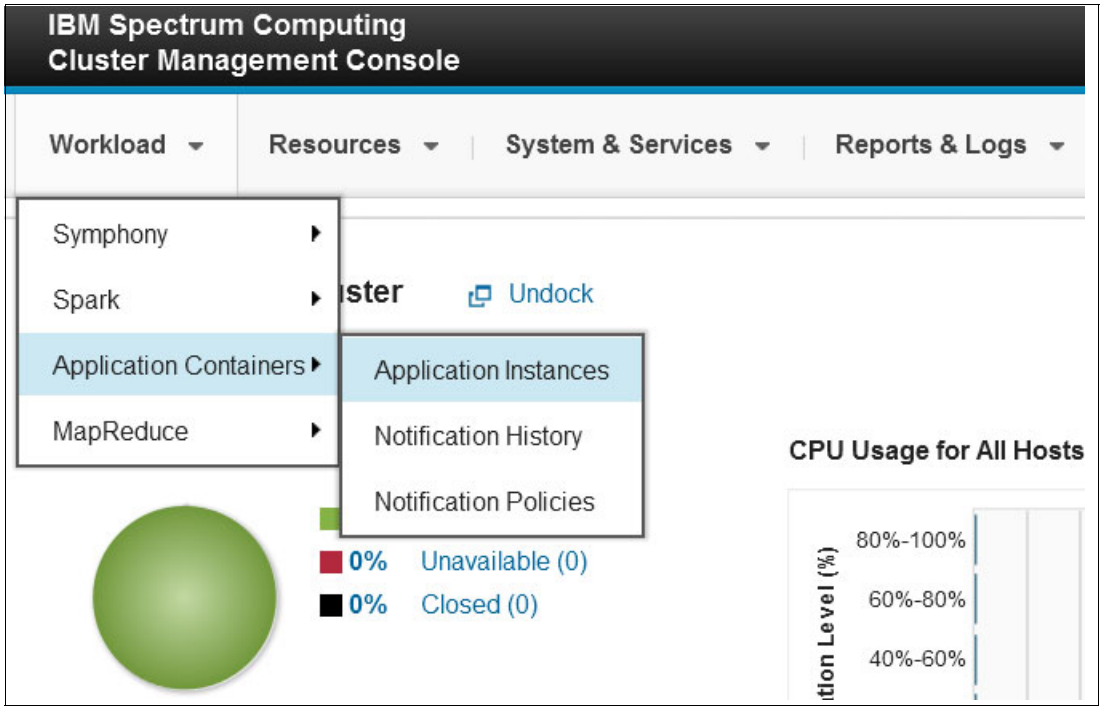


Figure 8-43 Select Application Instances

4. In the Application Instance panel, click **New** to Open a new Application Instance Registration dialog box, as shown in Figure 8-44.

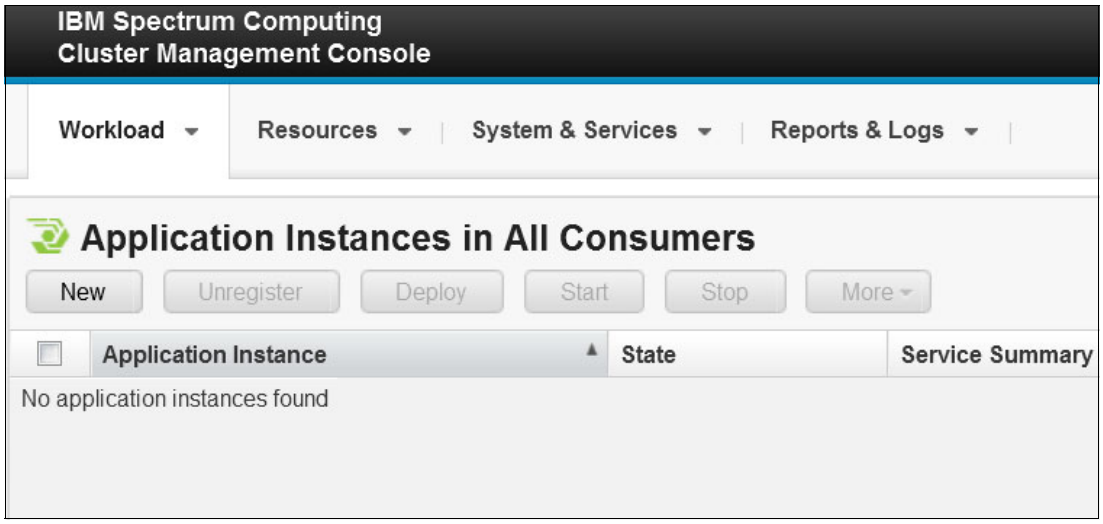


Figure 8-44 Application Instance panel

5. In the New Application Instance dialog box, Figure 8-45, click **Browse** and Select the yaml file downloaded from the sample template.

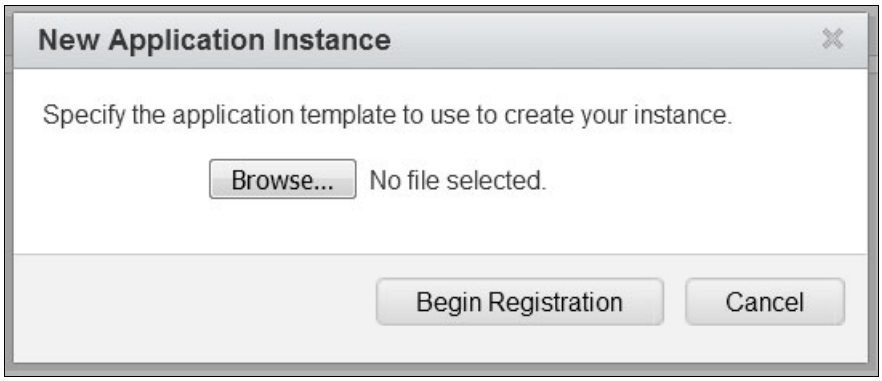


Figure 8-45 New Application Instance

6. With the correct file selected, click **Begin Registration** on the dialog box, as shown in Figure 8-46.

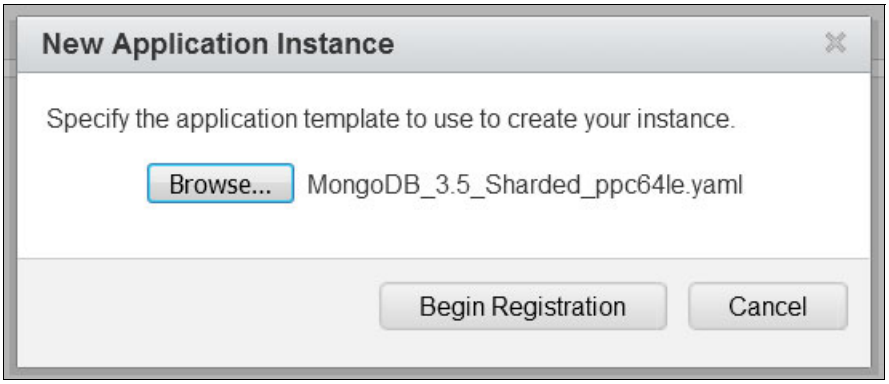


Figure 8-46 Template selection at New Application Instance dialog

7. First choose the application instance name. As you can see in Figure 8-47, it is called MongoDB1. Click the **Next** button to get to the Consumer step.

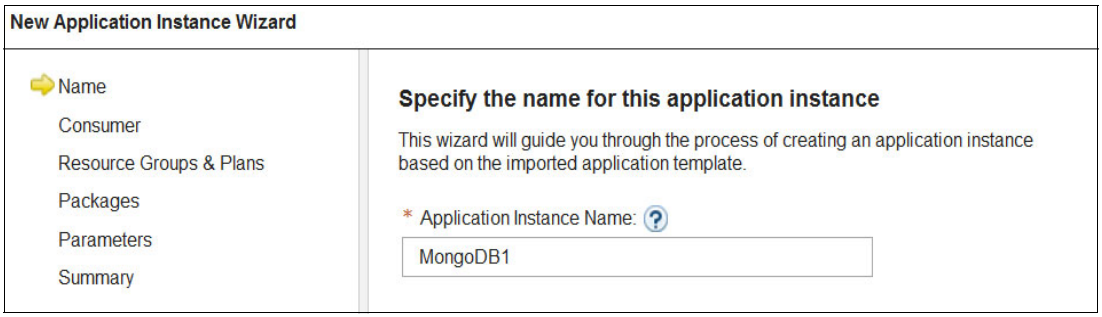


Figure 8-47 New Application Instance Wizard Name Selection

8. As shown in Figure 8-48, create a new Consumer for this application and call it MongoDB. Click the **Next** button to get to the Resource Group and Plans Dialog, as shown in Figure 8-49 on page 177.

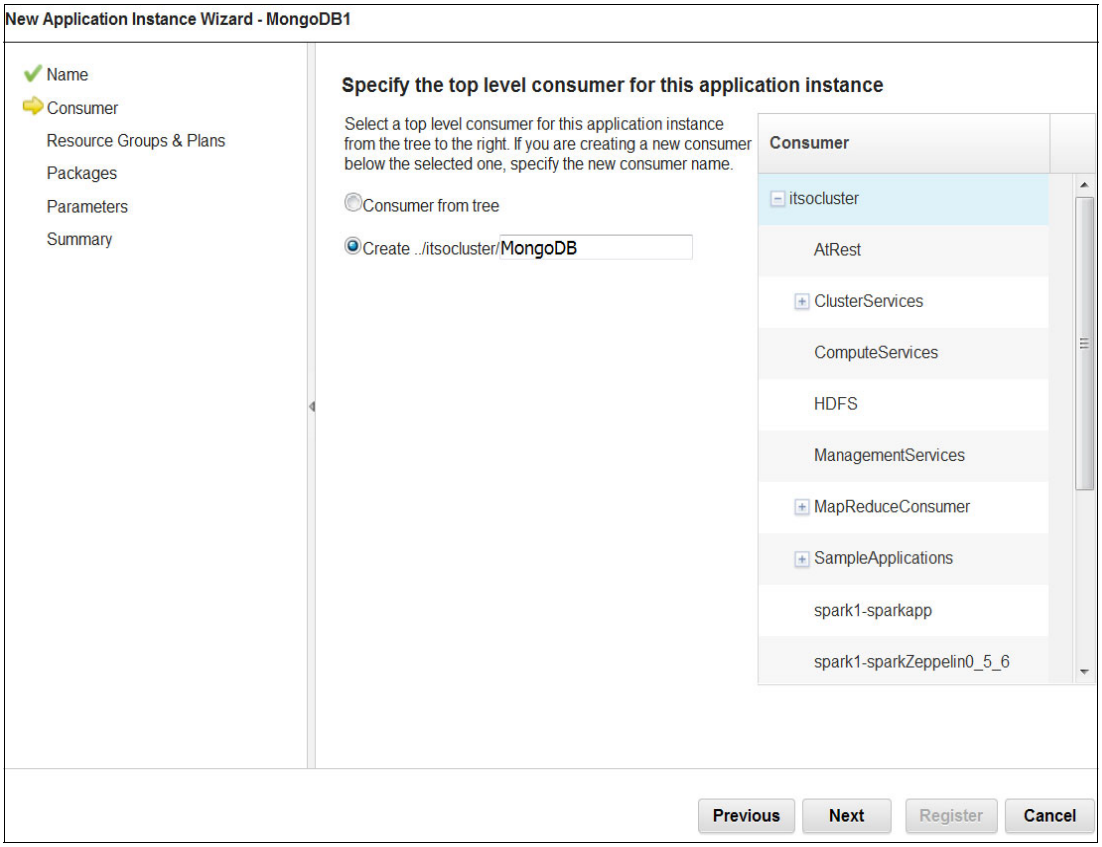


Figure 8-48 New Application Instance Wizard Consumer Selection

9. This is a small cluster test, so we use all of our compute hosts to provide resources for the MongoDB consumer. To do so, in Figure 8-49, select the ComputeHosts Resource Group for Routers, Shards, and Config Servers. Then click **Next** to skip to the Packages selection.

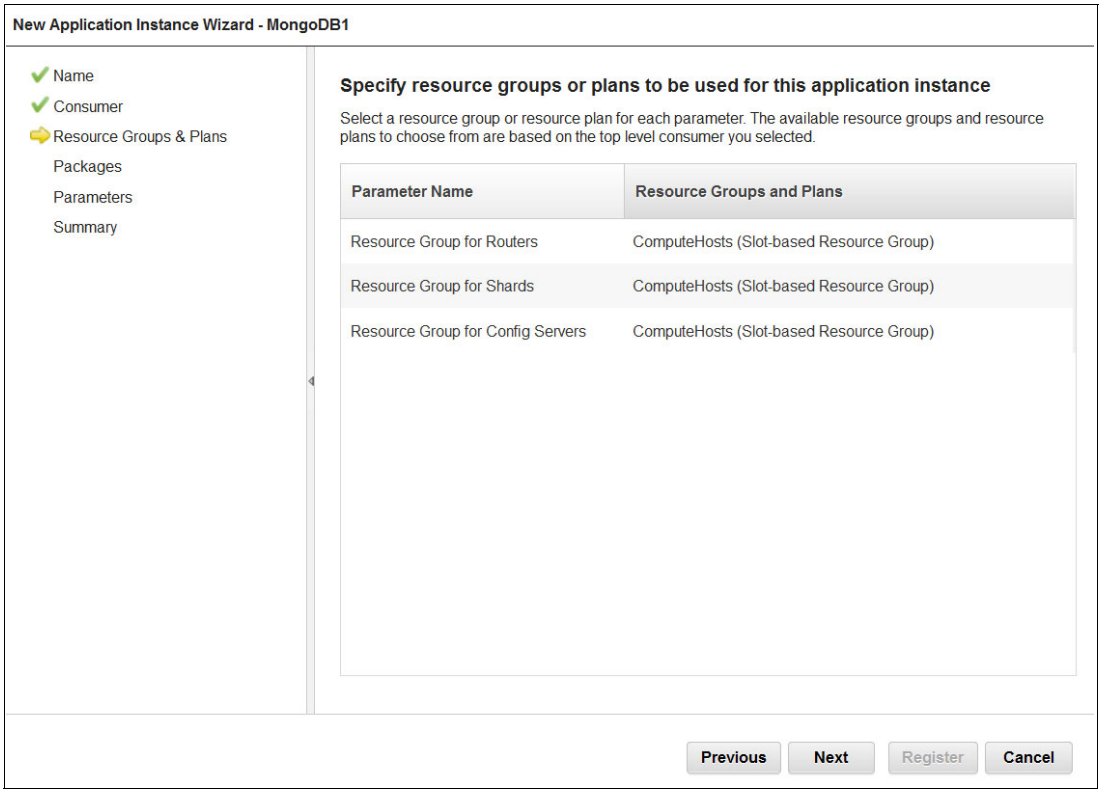


Figure 8-49 New Application Instance Wizard Resource Group Selection

10. At the step shown in Figure 8-50, select the MongoDB package created in Example 8-40 on page 172 and downloaded to our Desktop. Then click **Next** to select the parameters of the application (Figure 8-51).

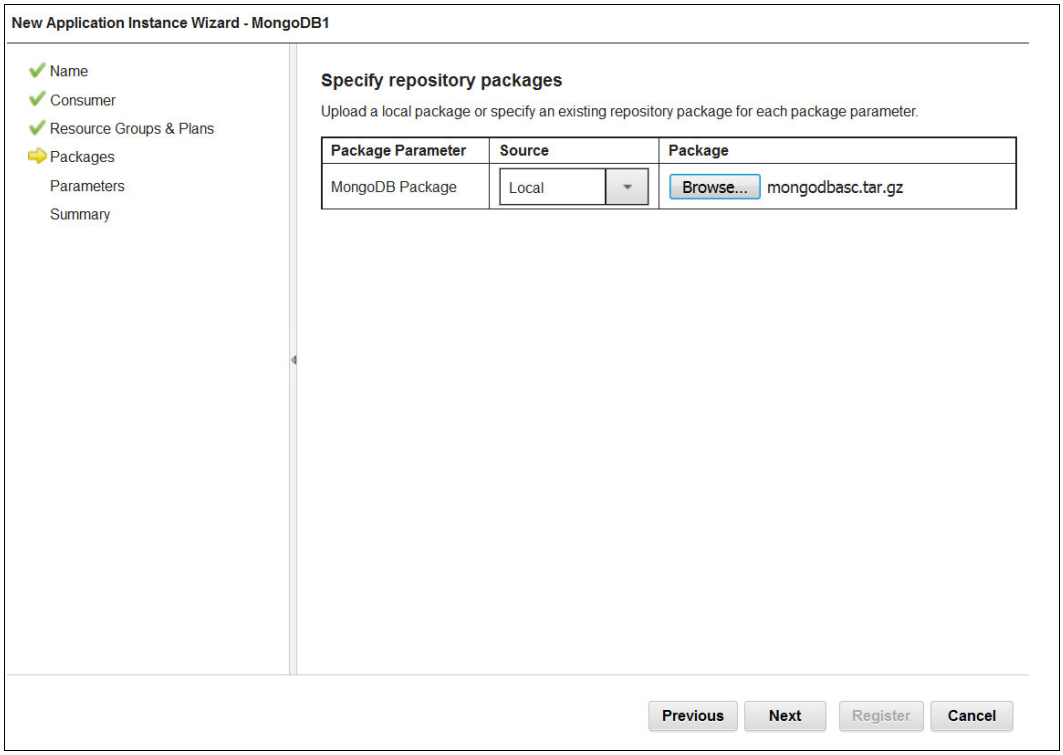


Figure 8-50 New Application Instance Wizard Package Selection

As you can see in Figure 8-51, this test uses `egoadmin` as the execution user and its home directory as the working directory.

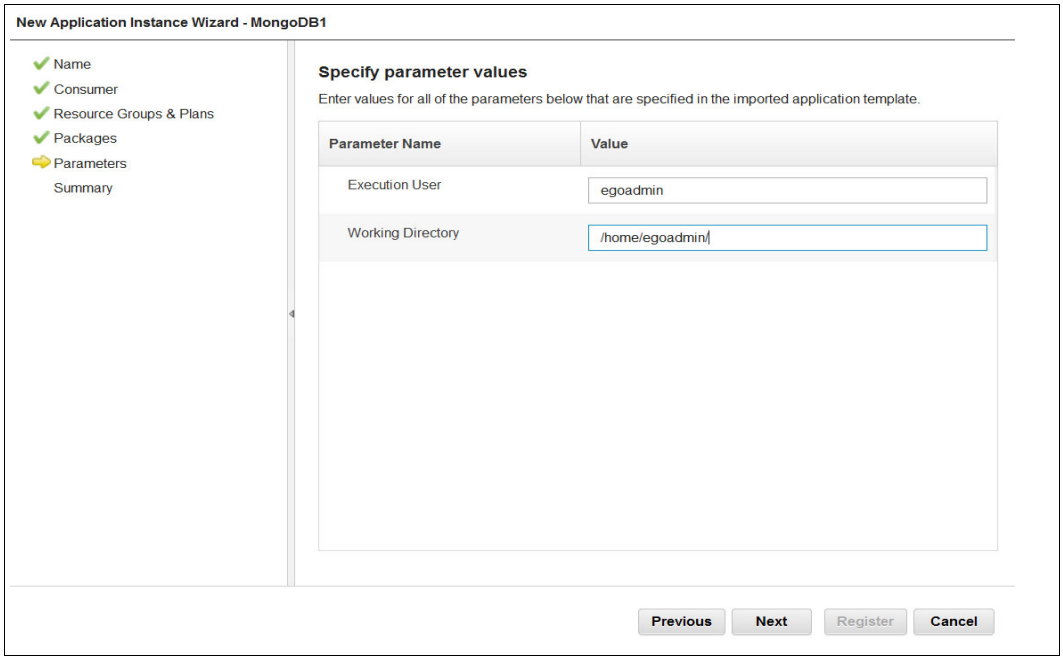


Figure 8-51 New Application Instance Wizard Parameters Selection

11. The last step on the Wizard is shown in Figure 8-52. The summary shows all the choices made for the new application instance. Then click **Register** to process the application registration.

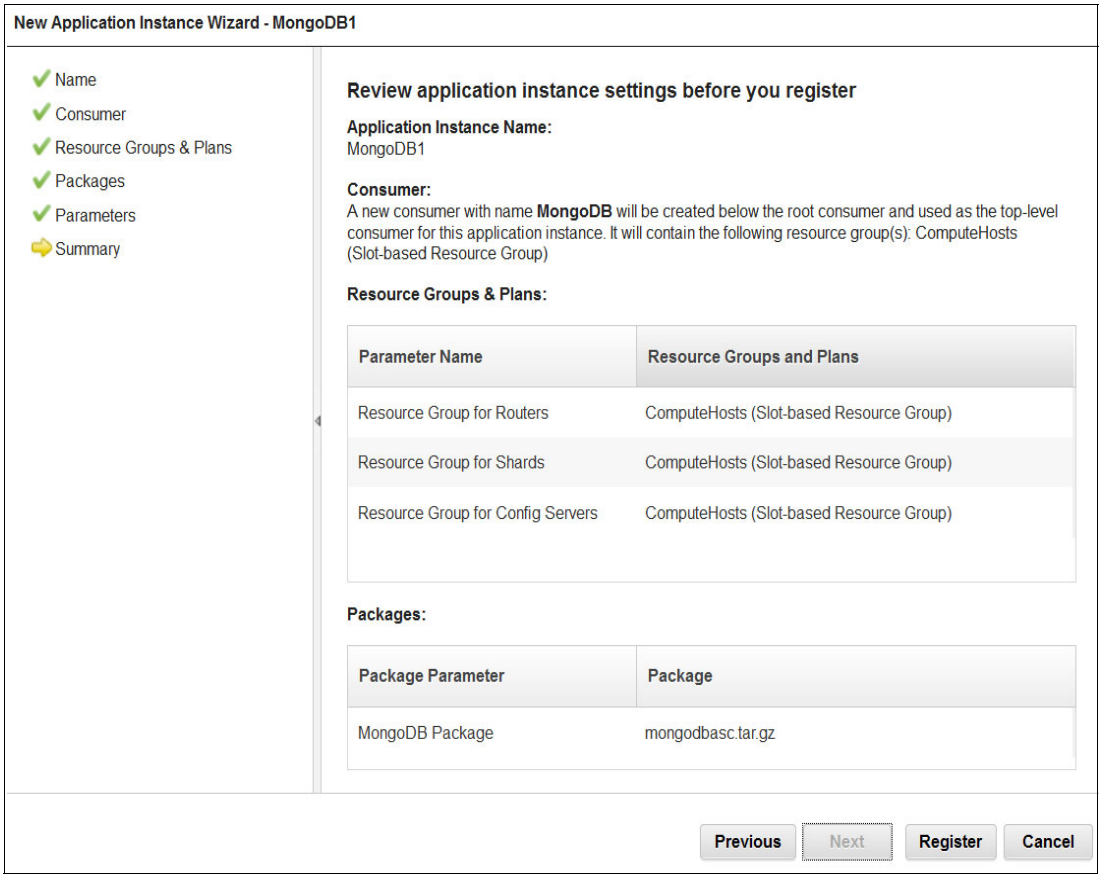


Figure 8-52 New Application Instance Wizard Summary

The application registration can take some time depending on the package size as shown in Figure 8-53.

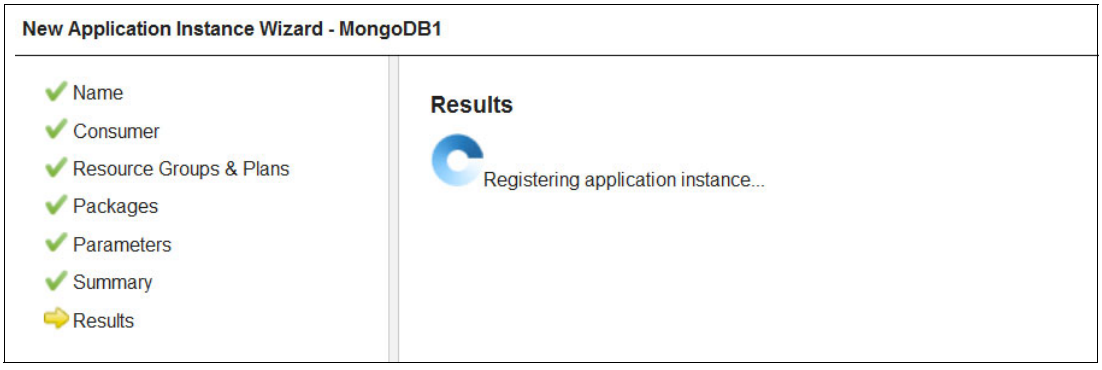


Figure 8-53 Registering new application instance in the cluster

12. After the Application is successfully registered, you see the message displayed in Figure 8-54. Click the **Done** button.

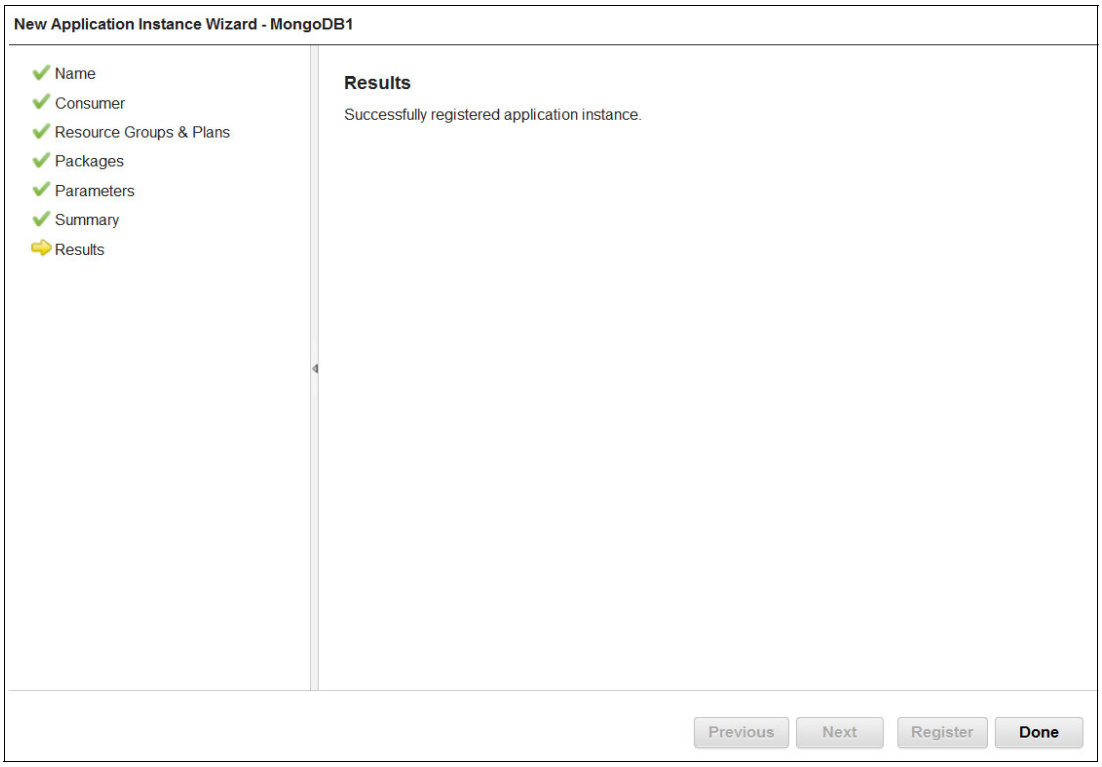


Figure 8-54 Application successfully registered

13. Back in the Application Instances, you can see now, as shown in Figure 8-55, the MongoDB1 Application just registered. Select it and click the **Deploy** button.

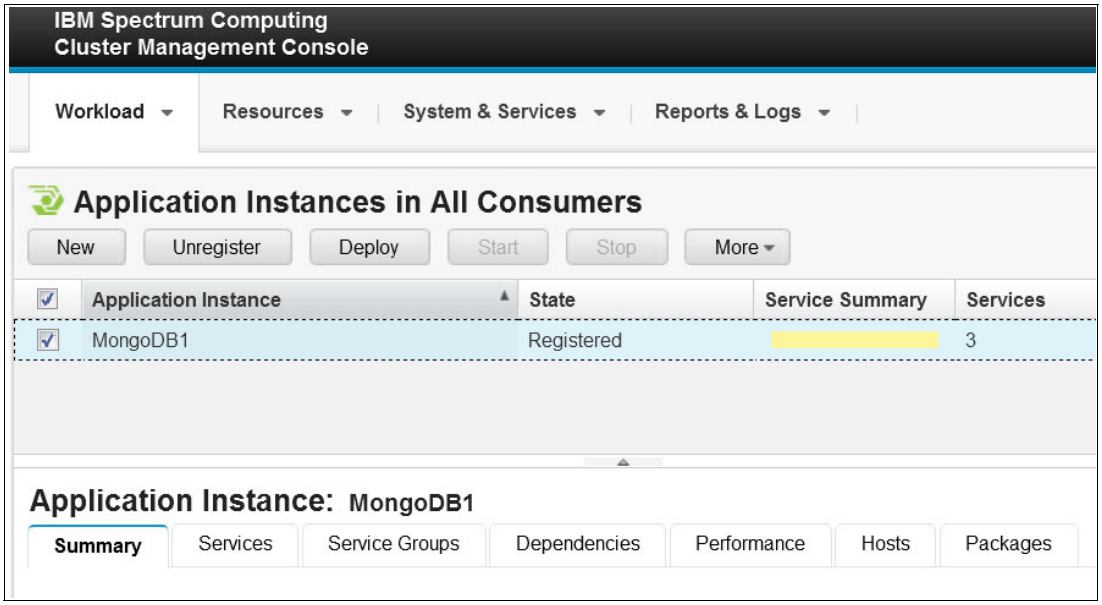


Figure 8-55 MongoDB1 application registered in the Application Instances panel

14. After you click **Deploy**, it is possible to define a timeout value for the deploy. You just left this field blank and click the Deploy button on the dialog box as shown in Figure 8-56.

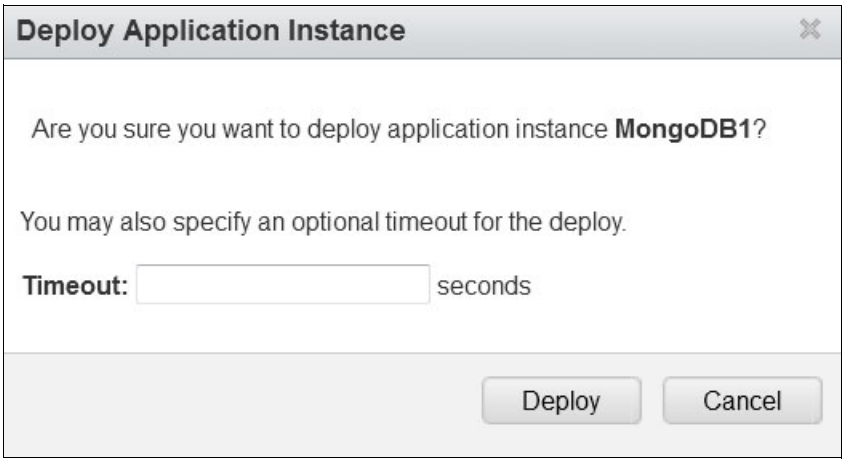


Figure 8-56 Deploy Application Instance

15. Deploy Application Instance dialog box.After you click **Deploy**, you are back to the Application Instances panel and can track the progress of the deployment as shown in Figure 8-57.

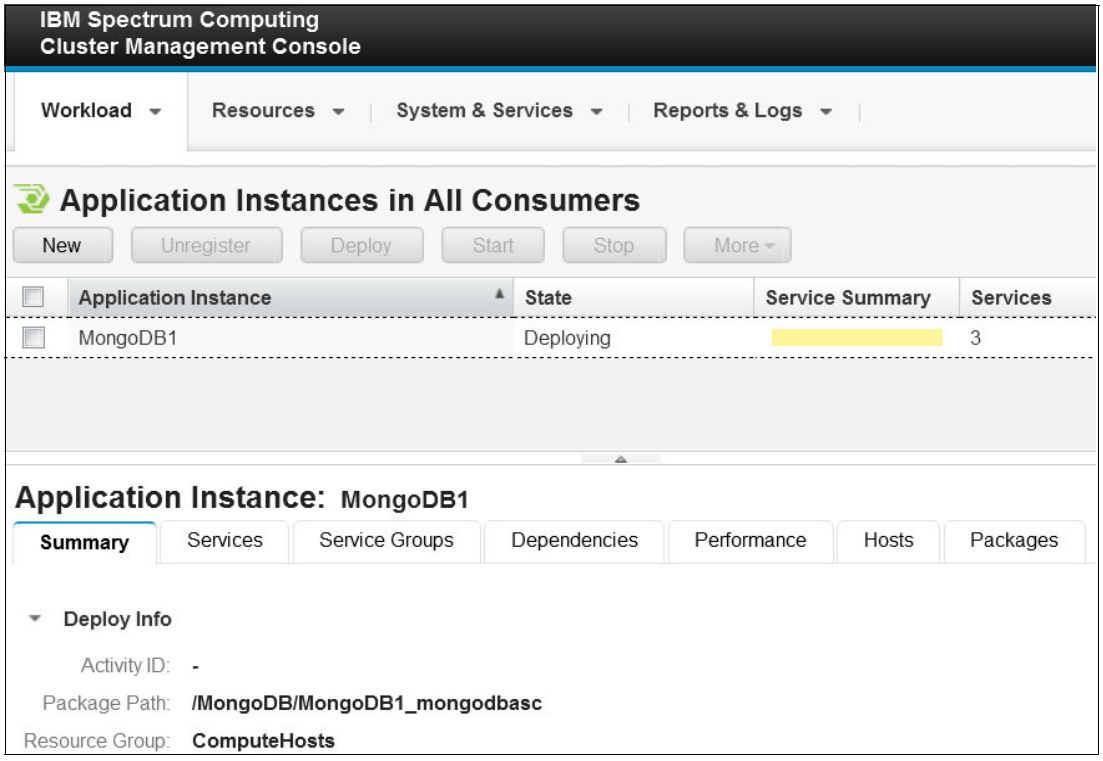


Figure 8-57 Application Instance MongoDB1 deploying

16. After the Deploy is done successfully, the application shows Ready in the State column as shown in Figure 8-58.

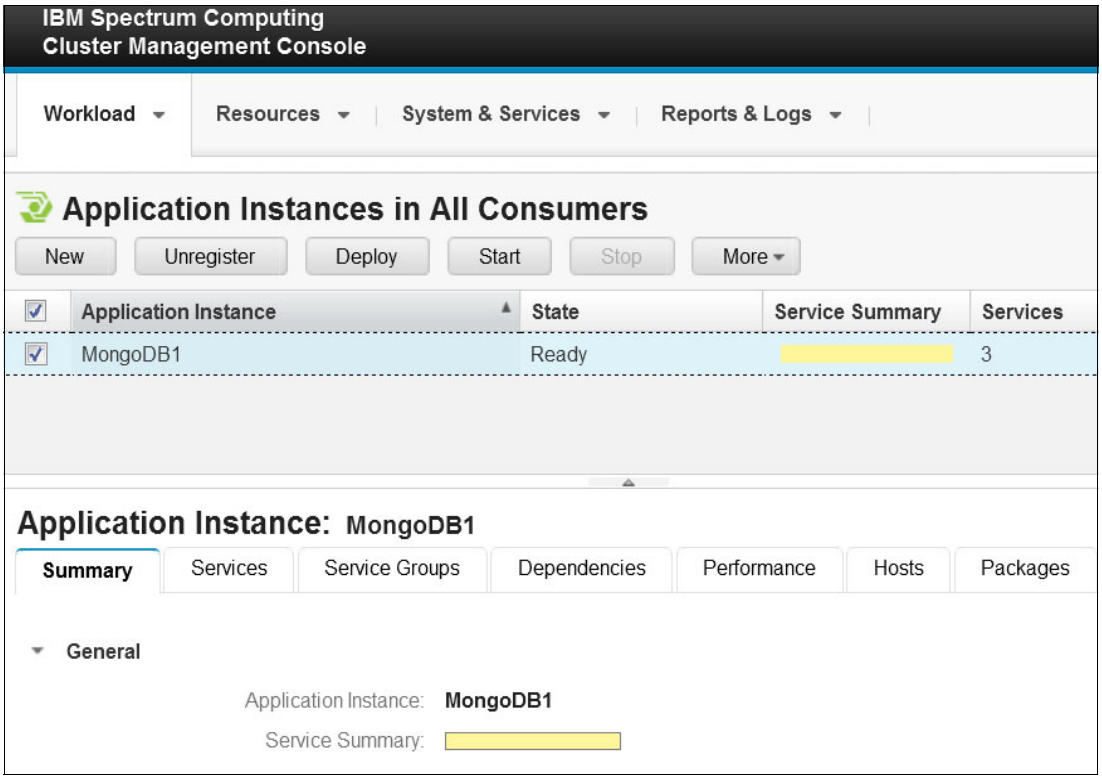


Figure 8-58 MongoDB1 Ready

17. Select the MongoDB1 application and click **Start**. A dialog box as shown in Figure 8-59 appears. Confirm by Clicking the **Start** Button.

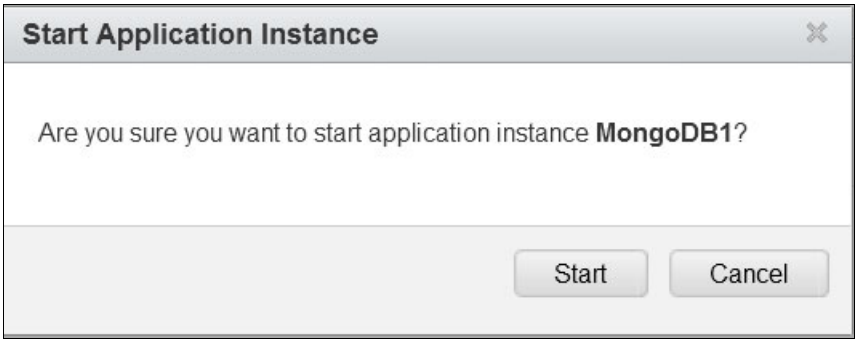


Figure 8-59 Start MongoDB1 Application Instance

18. Now follow the progress of the start of the application in the Application Instances Panel, as shown in Figure 8-60.

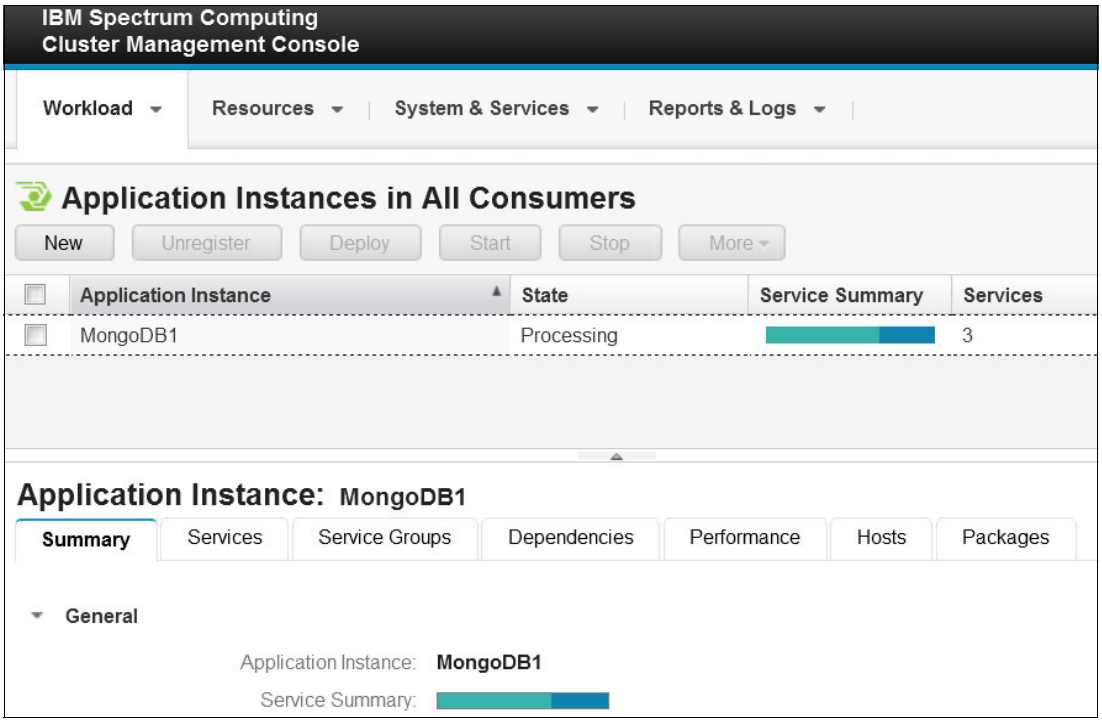


Figure 8-60 Start of MongoDB1 processing

After the correct start of all MongoDB services, Figure 8-61, the state changed to `Started`.

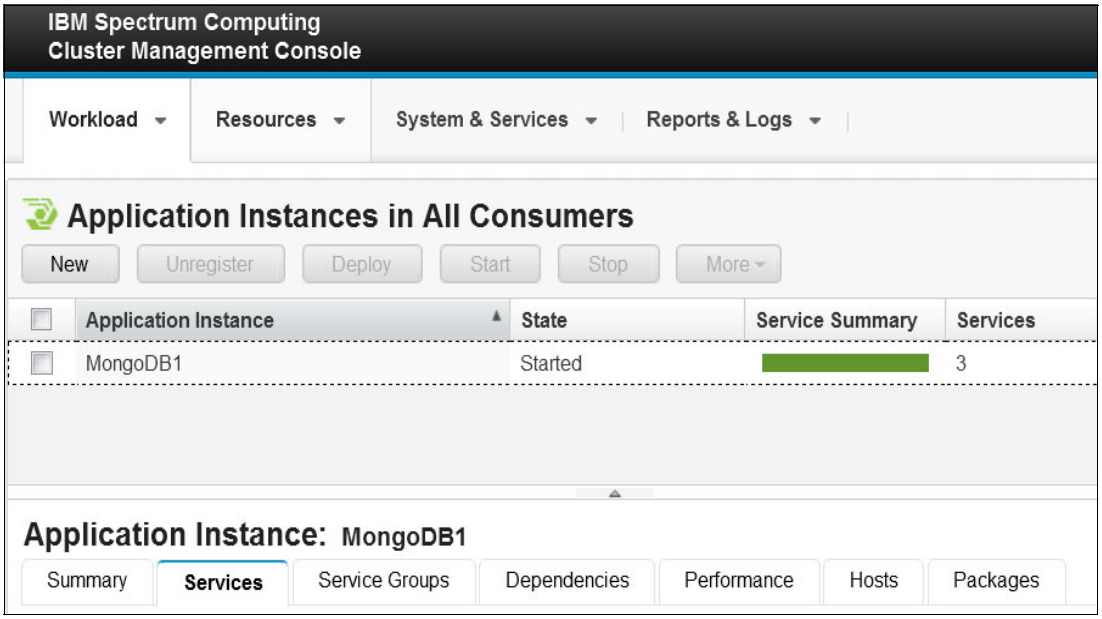


Figure 8-61 MongoDB1 Started

19. If you click the **Services** tab, you can see all Services that are running as shown in Figure 8-62.

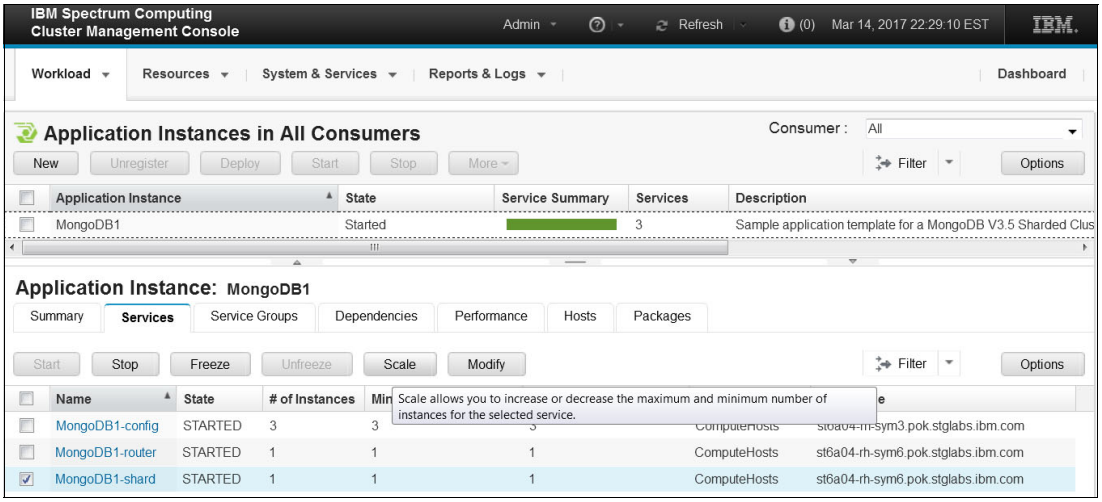


Figure 8-62 MongoDB1 Services

20. In Figure 8-62, you selected the MongoDB1-shard Services and clicked Scale. Change the number of instances to 3. You can now see in Figure 8-63 that there are 3 Instances of this Service with guarantee 3 replicas of the MongoDB database.

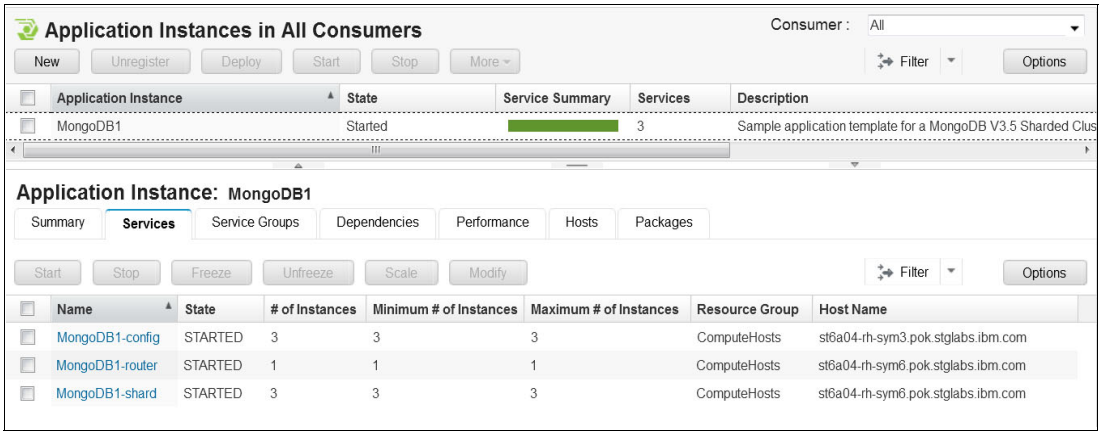


Figure 8-63 MongoDB Shard Service Scale-Out

21. At this point, click the Dashboard Link in the upper Right corner of the GUI, and check the usage distribution of the applications, as shown in Figure 8-64.

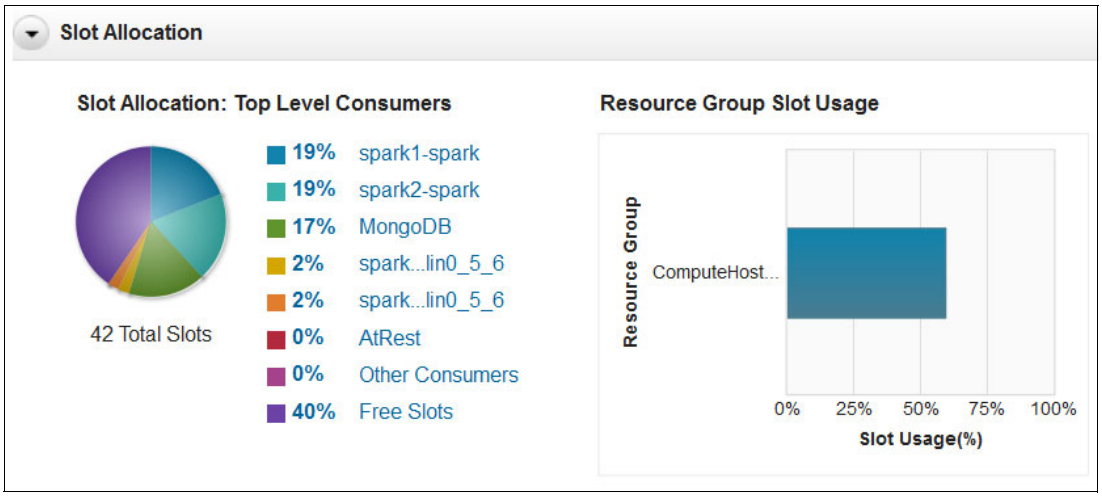


Figure 8-64 Applications in a Multi Tenant Cluster

MongoDB has started and it is up and running. Follow the MongoDB getting started tutorial.

We skipped the installation part of the tutorial because the process we just did on IBM Spectrum Conductor already installed MongoDB.

22. We download the sample data set to work and verify that MongoDB is correctly working. In Example 8-42, we use **wget** to get the data set to our IBM Spectrum Scale namespace.

Example 8-42 Importing data set to IBM Spectrum Scale namespace

```
[root@st6a04-rh-sym1 gpfs]# wget
https://raw.githubusercontent.com/mongodb/docs-assets/primer-dataset/primer-dat
aset.json
--2017-03-18 09:11:48--
https://raw.githubusercontent.com/mongodb/docs-assets/primer-dataset/primer-dat
aset.json
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
151.101.192.133, 151.101.64.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|151.101.192.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11874761 (11M) [text/plain]
Saving to: ‘primer-dataset.json’

100%[=======================================>] 11,874,761  15.4MB/s   in
0.7s
2017-03-18 09:11:49 (15.4 MB/s) - ‘primer-dataset.json’ saved
[11874761/11874761]

[root@st6a04-rh-sym1 gpfs]#
```

23. To see where to connect to the MongoDB Database, go to **Workload** → **Application Containers** → **Application Instances** then select MongoDB1 Application and click the **Services** Tab. Click the **MongoDB1-router** Service link and the **Service Instance** Tab.

    The Host to point to the client is shown in the Host column as you can see in Figure 8-65. In this case, the client is on the `st6a04-rh-sym5` host.

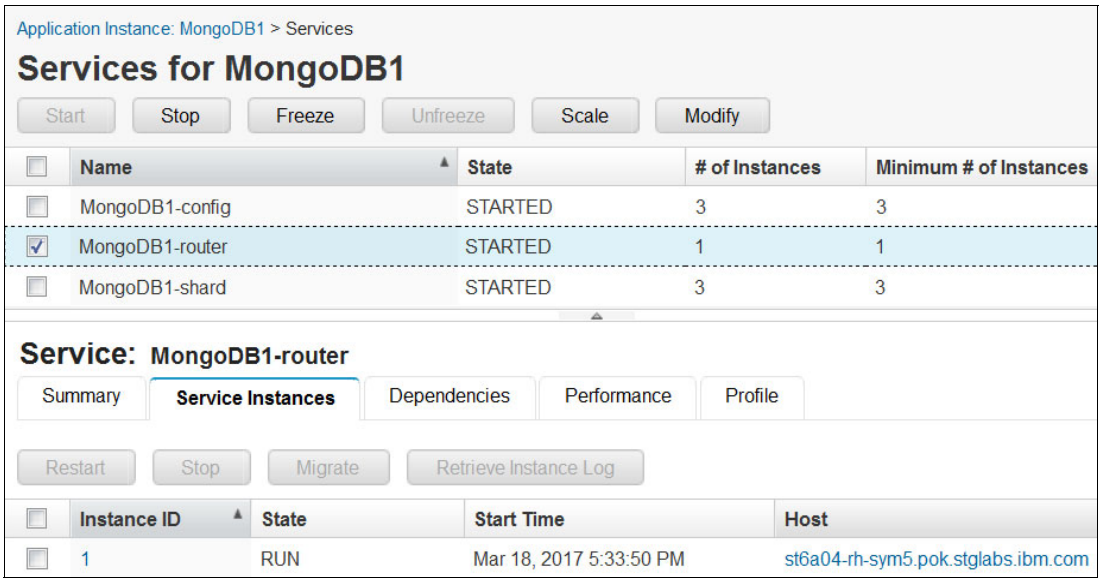


Figure 8-65 MongoDB1 router server

24. We use this information now in Example 8-43 to import the data set downloaded in Figure 8-65 to a new database called `test`.

Example 8-43 Creating a new database with restaurants collection

```
[root@st6a04-rh-sym1 template]#
/home/egoadmin/var/asc/auto/mongodb/mongodb-linux-ppc64le/bin/mongoimport
--host st6a04-rh-sym5 --db test --collection restaurants --drop --file
/gpfs/primer-dataset.json
2017-03-18T17:59:42.089-0400    connected to: st6a04-rh-sym5
2017-03-18T17:59:42.089-0400    dropping: test.restaurants
2017-03-18T17:59:45.080-0400    [################........] test.restaurants
7.84MB/11.3MB (69.2%)
2017-03-18T17:59:46.519-0400    [########################] test.restaurants
11.3MB/11.3MB (100.0%)
2017-03-18T17:59:46.519-0400    imported 25359 documents
```

25. In Example 8-44 we connect to the mongoDB router and switched to the new database (`--db test`) that was just created.

Example 8-44 Connecting to MongoDB test database

```
[root@st6a04-rh-sym1 template]#
/home/egoadmin/var/asc/auto/mongodb/mongodb-linux-ppc64le/bin/mongo --host
st6a04-rh-sym5
MongoDB shell version v3.5.4-25-g54f3fff
connecting to: mongodb://st6a04-rh-sym5:27017/
MongoDB server version: 3.5.4-25-g54f3fff
Server has startup warnings:
2017-03-18T17:33:51.246-0400 I CONTROL  [main]
```

```
2017-03-18T17:33:51.246-0400 I CONTROL  [main] ** NOTE: This is a development
version (3.5.4-25-g54f3fff) of MongoDB.
2017-03-18T17:33:51.246-0400 I CONTROL  [main] **       Not recommended for
production.
2017-03-18T17:33:51.246-0400 I CONTROL  [main]
2017-03-18T17:33:51.246-0400 I CONTROL  [main] ** WARNING: Access control is
not enabled for the database.
2017-03-18T17:33:51.246-0400 I CONTROL  [main] **          Read and write
access to data and configuration is unrestricted.
2017-03-18T17:33:51.246-0400 I CONTROL  [main]
MongoDB Enterprise mongos> use test
switched to db test
MongoDB Enterprise mongos>
```

26. In Example 8-45, we manually insert a new document in the restaurants collection imported.

Example 8-45 Inserting a new document

```
MongoDB Enterprise mongos> db.restaurants.insert(
...     {
...       "address" : {
...          "street" : "2 Avenue",
...          "zipcode" : "10075",
...          "building" : "1480",
...          "coord" : [ -73.9557413, 40.7720266 ]
...       },
...       "borough" : "Manhattan",
...       "cuisine" : "Italian",
...       "grades" : [
...          {
...             "date" : ISODate("2014-10-01T00:00:00Z"),
...             "grade" : "A",
...             "score" : 11
...          },
...          {
...             "date" : ISODate("2014-01-16T00:00:00Z"),
...             "grade" : "B",
...             "score" : 17
...          }
...       ],
...       "name" : "Vella",
...       "restaurant_id" : "41704620"
...     }
... )
WriteResult({ "nInserted" : 1 })
MongoDB Enterprise mongos>
```

27. Now use the find method as shown in Example 8-46 to call for all documents that has Manhattan as the borough. We show just the first line of the results.

Example 8-46 Querying the collection

```
MongoDB Enterprise mongos> db.restaurants.find( { "borough": "Manhattan" } )
{ "_id" : ObjectId("58cdadce90e4c11ad6db36c8"), "address" : { "building" :
"351", .
.
.
MongoDB Enterprise mongos>
```

28. To get a little bit more of insight on the restaurants, use the aggregate method to find the number of Brazilian restaurants in Queens per zip code as shown in Example 8-47.

Example 8-47 Using aggregate method

```
MongoDB Enterprise mongos> db.restaurants.aggregate(
...    [
...      { $match: { "borough": "Queens", "cuisine": "Brazilian" } },
...      { $group: { "_id": "$address.zipcode" , "count": { $sum: 1 } } }
...    ]
... );
{ "_id" : "11377", "count" : 1 }
{ "_id" : "11103", "count" : 1 }
{ "_id" : "11106", "count" : 3 }
{ "_id" : "11368", "count" : 1 }
{ "_id" : "11101", "count" : 2 }
```

A

IBM Cluster Foundation high availability configuration example

This appendix contains the output of the IBM Cluster Foundation high availability configuration example.

Example A-1 shows the output of the **pcmhatool** command used to configure high availability for the IBM Cluster Foundation.

The syntax of the command is as follows:

```
#pcmhatool config -i /etc/ha.info -s st6a04-rh-cf02
```

where the file /etc/ha.info contains the following lines:

```
# cat /etc/ha.info
st6a04-rh-cf:
   nicips.eth1:0=10.0.0.4
   nicips.eth0:0=9.47.76.106
   sharefs_type= gpfs
   sharefs_mntp.work =/gpfs
```

Example A-1 Output for the pcmhatool command used to set up high availability

```
Setting up high availability between the management node 'st6a04-rh-cf01' and the
standby management node 'st6a04-rh-cf02'. Do not perform any operations during
this time.

 Parsing high availability data...
 EGO installation path: /opt/ibm/spectrumcomputing
 ================================================================

 Virtual node name: st6a04-rh-cf
 Virtual IP address: eth1:0; 10.0.0.4
 Virtual IP address: eth0:0; 9.47.76.106
```

```
 Shared work directory on: /gpfs
 ================================================================


 During the high availability setup, data is copied to shared directories
 and IBM Spectrum Cluster Foundation services are restarted several times.
 Ensure that the external shared storage server and network connections are
available.


 Continue? (Y/N) Y
 Checking if passwordless SSH is available...
 Getting standby management node data...
 Copying the root id_rsa keys to the standby management node...
 Generating a node definition file on the standby management node...
 Copying the node definition file from the standby management node to
/tmp/pcmhatool/smn.info
 Checking if the secondary management node is consistent with the management
node...
 Checking if the secondary management node is
 consistent with the management node...
 Check the product of management node and secondary management node name.
 Run command '/opt/pcm/bin/pcm-ha-support check --product' on secondary management
node 'st6a04-rh-cf02'
 Command '/opt/pcm/bin/pcm-ha-support check --product' run on management node.
 Check the secondary management node name: st6a04-rh-cf02
 Checking the network interfaces on the secondary management node.
 Checking if the IP address 10.0.0.11 is in the IP address range
10.0.0.3-10.0.0.90.
 Checking if the IP address 10.0.0.12 is in the IP address range
10.0.0.3-10.0.0.90.
 Checking if the IP address 10.0.0.11 is in the IP address range
9.47.76.96-9.47.76.254.
 Checking if the IP address 9.47.76.96 is in the IP address range
10.0.0.3-10.0.0.90.
 Checking if the IP address 9.47.76.96 is in the IP address range
9.47.76.96-9.47.76.254.
 Checking if the IP address 9.47.76.97 is in the IP address range
9.47.76.96-9.47.76.254.
 Checking image profiles on secondary management node.
 Checking timezone on the standy management node.
 Command '/opt/pcm/bin/pcm-ha-support check --tz' run on the manage node
'st6a04-rh-cf01'
 Command '/opt/pcm/bin/pcm-ha-support check --tz' run on the secondary management
node 'st6a04-rh-cf02'
 Checking the virtual network interface...
 Checking if the virtual network interface <eth1:0>
 is available...
 Checking the network interfaces on the virtual node...
 Command '/opt/pcm/bin/pcm-ha-support check --nic eth1:0' run on secondary
management node 'st6a04-rh-cf02'
 Run command '/opt/pcm/bin/pcm-ha-support check --nic eth1:0'
 Checking the network interface: eth1:0
 Checking if the virtual IP address <10.0.0.4>
 is available...
```

```
 Checking the IP address of the virtual node...
 Command '/opt/pcm/bin/pcm-ha-support check --ip 10.0.0.4' run on secondary
management node 'st6a04-rh-cf02'
 Run command '/opt/pcm/bin/pcm-ha-support check --ip 10.0.0.4'
 Checking the IP address 10.0.0.4
 Checking if the virtual network interface <eth0:0>
 is available...
 Checking the network interfaces on the virtual node...
 Command '/opt/pcm/bin/pcm-ha-support check --nic eth0:0' run on secondary
management node 'st6a04-rh-cf02'
 Run command '/opt/pcm/bin/pcm-ha-support check --nic eth0:0'
 Checking the network interface: eth0:0
 Checking if the virtual IP address <9.47.76.106>
 is available...
 Checking the IP address of the virtual node...
 Command '/opt/pcm/bin/pcm-ha-support check --ip 9.47.76.106' run on secondary
management node 'st6a04-rh-cf02'
 Run command '/opt/pcm/bin/pcm-ha-support check --ip 9.47.76.106'
 Checking the IP address 9.47.76.106
 Checking share storage...
 Checking if the shared directory work is available...
 Checking /gpfs on the secondary management node...
 Command '/opt/pcm/bin/pcm-ha-support check --path /gpfs --sharetype work --smn
--gpfstype' run on secondary management node 'st6a04-rh-cf02'
 Checking /gpfs on the management node...
 Run command '/opt/pcm/bin/pcm-ha-support check --path /gpfs --sharetype work
--gpfstype'
 Checking if /shared_pcm mounted on the path /gpfs
 Checking if the path /gpfs is mountable.
 Checking the mount point /tmp/pcm-nfsutilv7HnDO permissions
 Checking the mount point /gpfs size
 Lock file created. High availability setup starting on st6a04-rh-cf01.
 Stopping services...
 stopping elk-shipper...
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 stopping elk-indexer...
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 stopping elk-kibana...
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 stopping elk-elasticsearch...
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 Stop EGOSC services
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 EGO installation path: /opt/ibm/spectrumcomputing
 stop 'ego'
 EGO installation path: /opt/ibm/spectrumcomputing
 Skip, service 'autofs' is already stopped
 Service 'dhcpd' is stopped
 Service 'named' is stopped
```

```
Service 'nfs-server' is stopped
Skip, service 'rpcidmapd' is already stopped
Service 'xcatd' is stopped
Synchronizing system work data. This can take
several minutes...
Create directory: /shared_pcm/root
Create directory: /shared_pcm/etc
Create directory: /shared_pcm/var
Create directory: /shared_pcm/var/lib
Create directory: /shared_pcm/var/lib/pgsql
Create directory: /shared_pcm/opt
Create directory: /shared_pcm/opt/pcm
Create directory: /shared_pcm/opt/pcm/activemq
Creating local link for directory: /install.
Renaming "/install" to "/install.PCMHA".
Creating local link for directory: /root/.xcat.
Renaming "/root/.xcat" to "/root/.xcat.PCMHA".
Creating local link for directory: /tftpboot.
Renaming "/tftpboot" to "/tftpboot.PCMHA".
Creating local link for directory: /etc/xcat.
Renaming "/etc/xcat" to "/etc/xcat.PCMHA".
Creating local link for directory: /etc/conserver.cf.
Renaming "/etc/conserver.cf" to "/etc/conserver.cf.PCMHA".
Creating local link for directory: /etc/ntp.conf.
Renaming "/etc/ntp.conf" to "/etc/ntp.conf.PCMHA".
Creating local link for directory: /etc/group.merge.
Renaming "/etc/group.merge" to "/etc/group.merge.PCMHA".
Creating local link for directory: /etc/passwd.merge.
Renaming "/etc/passwd.merge" to "/etc/passwd.merge.PCMHA".
Creating local link for directory: /etc/shadow.merge.
Renaming "/etc/shadow.merge" to "/etc/shadow.merge.PCMHA".
Creating local link for directory: /var/lib/pgsql/data.
Renaming "/var/lib/pgsql/data" to "/var/lib/pgsql/data.PCMHA".
Creating local link for directory: /opt/pcm/activemq/data.
Renaming "/opt/pcm/activemq/data" to "/opt/pcm/activemq/data.PCMHA".
Creating local link for directory: /opt/pcm/entitlement.
Renaming "/opt/pcm/entitlement" to "/opt/pcm/entitlement.PCMHA".
Creating local link for directory: /opt/pcm/etc.
Renaming "/opt/pcm/etc" to "/opt/pcm/etc.PCMHA".
Creating local link for directory: /opt/pcm/web-portal.
Renaming "/opt/pcm/web-portal" to "/opt/pcm/web-portal.PCMHA".
Creating local link for directory: /opt/pcm/pcmd.
Renaming "/opt/pcm/pcmd" to "/opt/pcm/pcmd.PCMHA".
Creating local link for directory: /opt/pcm/rule-engine.
Renaming "/opt/pcm/rule-engine" to "/opt/pcm/rule-engine.PCMHA".
Synchronizing EGO work data. This can take
several minutes...
EGO installation path: /opt/ibm/spectrumcomputing
Creating '/shared_pcm/opt/ibm/spectrumcomputing' on the management node...
Create directory: /shared_pcm/opt/ibm
Create directory: /shared_pcm/opt/ibm/spectrumcomputing
Running 'su -c ". /opt/ibm/spectrumcomputing/profile.platform; egoconfig mghost
/shared_pcm/opt/ibm/spectrumcomputing -f" pcmadmin' on the management node...
Starting the xCAT daemon...
Starting service 'postgresql'
```

```
 Starting service 'xcatd'
 Saving high availability data...

 Starting high availability configuration on the management node...
 Configured high availability on the management node.
 Generating the high availability data file...

 Starting high availability configuration on the standby management node...
 Copying the high availability data file to the standby management node...
 Running '. /etc/profile.d/pcmenv.sh; /opt/pcm/bin/pcm-ha-support config -i
/tmp/ha.dat' on the standby management node...
 High availability configured on the standby management node.
 Starting IBM Spectrum Cluster Foundation services...
 start 'ego'
 EGO installation path: /opt/ibm/spectrumcomputing
High availability controller starting...
High availability setup is complete. The setup log file pcmhatool.log is located
in the /opt/pcm/log directory.
 If an error occurred while starting services, manually fix the error and restart
the services.

 To source the environment run the 'source /etc/profile.d/pcmenv.sh' command. This
is not required for new login sessions.

 It can take several minutes for all high availability services to start. To get
the status of high availability services, run the 'pcmhatool check' command.
 After IBM Spectrum Cluster Foundation high availability is enabled, you can
access the Web Portal at http://mgtnode-virtual-IP:8080, where mgtnode-virtual-IP
 is the virtual management node IP address. If you are connected to a public
network, you can also navigate to http://mgtnode-virtual-hostname:8080, where
mgtnode-virtual-hostname is the virtual management node hostname.
 If the HTTPS is enabled, you can access the Web Portal at
https://mgtnode-virtual-IP:8443 or https://mgtnode-virtual-hostname:8443.

Checking if the 'st6a04-rh-cf02' is available           [  OK  ]
Starting diagnostic on the management node 'st6a04-rh-cf02':
   Checking the high availability data file...             [  OK  ]
   Checking the virtual network...                         [  OK  ]
   Checking the IBM Spectrum Scale servers...              [  OK  ]
   Checking if the activeMQ is running...                  [  OK  ]
   Checking shared directories...                          [  OK  ]
   Checking if the database is running...                  [  OK  ]
   Checking if the EGO is running...                       [  OK  ]
   Checking if the xcat is running...                      [  OK  ]
   Checking if the dhcp server is running...               [  OK  ]
   Checking the virtual provision IP address and
   external storage location in database...                [  OK  ]
   Checking if the PERF is running...                      [  OK  ]
   Checking if the Rule-Engine is running...               [  OK  ]
   Checking EGO services...                                [  OK  ]
   Checking if the Web Portal is running...                [  OK  ]
   Checking if the PCMD is running...                      [  OK  ]
Starting diagnostic on the standby management node 'st6a04-rh-cf01':
   Checking the high availability data file...             [  OK  ]
   Checking the IBM Spectrum Scale servers...              [  OK  ]
```

```
   Checking shared directories...                          [  OK  ]
   Checking the virtual network...                         [  OK  ]
   Checking if the EGO is running...                       [  OK  ]
   Checking EGO services...                                [  OK  ]
High availability diagnostic check completed. Management node high availability is
ready.
```

Related publications

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

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 Platform Computing Solutions Reference Architectures and Best Practices*, SG24-8169

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

Other publications

These publications are also relevant as further information sources:

- IBM Cluster Foundation installation
  http://www.ibm.com/support/knowledgecenter/SSZUCA_4.2.2/docs_shared/installing.html

Online resources

These websites are also relevant as further information sources:

- Apache Hadoop Core project
  http://hadoop.apache.org/
- IBM Spectrum Computing
  http://www-03.ibm.com/systems/spectrum-computing/
- IBM Spectrum Conductor with Spark
  http://ibm.co/2hDCTcS

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

IBM Spectrum Computing Solutions

(0.2”spine)
0.17”<->0.473”
90<->249 pages

SG24-8373-00

ISBN 0738442522

Printed in U.S.A.