IBM

IBM Platform Computing Solutions Reference Architectures and Best Practices

- Helps with the foundation to manage enterprise environments
- Delivers reference architectures and best practices guides
- Provides case scenarios

Dino Quintero
Luis Carlos Cruz
Ricardo Machado Picone
Dusan Smolej
Daniel de Souza Casali
Gheorghe Tudor
Joanna Wong

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM Platform Computing Solutions Reference Architectures and Best Practices

April 2014

SG24-8169-00

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

First Edition (April 2014)

This edition applies to RedHat 6.4, IBM Platform Cluster Manager Standard Edition (PCM-SE) 4.1.1, IBM Platform Symphony Advanced Edition 6.1.1, GPFS FPO 3.5.0.13, Hadoop 1.1.1.

Contents

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

Trademarks

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

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

AIX®
Algorithmics®
BigInsights™
Cognos®
Global Technology Services®
GPFS™
IBM Flex System™
IBM SmartCloud®
IBM®
iDataPlex®
InfoSphere®
Intelligent Cluster™
LoadLeveler®
LSF®
Passport Advantage®
POWER7+™
POWER7®
PowerHA®
PowerLinux™
PowerVM®
POWER®
Redbooks®
Redbooks (logo) ®
Storwize®
Symphony®
System Storage®
System x®
Tivoli®

The following terms are trademarks of other companies:

Adobe, the Adobe logo, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Algorithmics, and Ai logo are trademarks or registered trademarks of Algorithmics, an IBM Company.

Intel Xeon, Intel, 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.

Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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® Redbooks® publication demonstrates and documents that the combination of IBM System x®, IBM GPFS™, IBM GPFS-FPO, IBM Platform Symphony®, IBM Platform HPC, IBM Platform LSF®, IBM Platform Cluster Manager Standard Edition, and IBM Platform Cluster Manager Advanced Edition deliver significant value to clients in need of cost-effective, highly scalable, and robust solutions. IBM depth of solutions can help the clients plan a foundation to face challenges in how to manage, maintain, enhance, and provision computing environments to, for example, analyze the growing volumes of data within their organizations.

This book addresses topics to educate, reiterate, confirm, and strengthen the widely held opinion of IBM Platform Computing as the systems software platform of choice within an IBM System x environment for deploying and managing environments that help clients solve challenging technical and business problems.

This book also addresses topics to that help answer customer's complex challenge requirements to manage, maintain, and analyze the growing volumes of data within their organizations and provide expert-level documentation to transfer the how-to-skills to the worldwide support teams.

This publication is targeted toward technical professionals (consultants, technical support staff, IT Architects, and IT Specialists) who are responsible for delivering cost-effective computing solutions that help 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 Senior Certified IT Specialist with the ITSO in Poughkeepsie, NY. His areas of knowledge 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.

Luis Carlos Cruz is a Solution Architect for Midrange and Storage with IBM CES SO Delivery with over a year at IBM. He has held research in big data analytics, winning the 2013 Regional Technical Exchange with a Big Data architecture for cloud banking. Among the positions Luis comes from is GBM, a strategic IBM Alliance company in Latin America where he holds positions in strategy, Tivoli® Architecture, project management, and management positions primarily working in and around Tivoli Service Management capabilities, management systems, data warehouse infrastructure, information integration, database administration, performance management, and database development technology. He has been a prominent speaker at industry events and customer briefings and a frequent contributor to industry articles, analyst research, and other publications.

Ricardo Machado Picone is an IBM Senior Certified IT Specialist in Brazil. He works with business intelligence architecture and reporting systems. He supports a project that provides revenue and aspiration metrics for the IBM worldwide sales teams. His areas of expertise include SQL language, ETL jobs, databases, data modeling, IBM Cognos® dashboard/scorecards, IBM Cognos administration, IT infrastructure support, and operational systems. Ricardo is an IBM Certified Business Intelligence Solutions Specialist and is leading the Business Intelligence Brazilian Center of Competence (COC-BI) in areas, such as IBM Cognos technical support and education. He holds a bachelor's degree in Business Administration from the Universidade Paulista.

Dusan Smolej is a Senior IT Architect at Global Technology Services®, IBM Czech Republic. He has over 18 years of experience in the IT industry, and 12 years with IBM. He works as an Integration Architect and was a Lead Architect for some of the successful complex engagements for local and international clients. His areas of expertise include Service Oriented Architecture, Enterprise Application Integration, Digital Archiving, Cloud Computing, Grid Computing, Performance Analysis, and Optimization. He holds a M.Sc. in Electrical Engineering from the Technical University of Kosice (TUKE), Slovakia.

Daniel de Souza Casali has 10 years working with IBM and is an IBM CrossSystems Senior Certified. He works for the Systems and Technology Group as Latin America Platform Computing 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, and clustering solutions.

Gheorghe Tudor is an IT Specialist and works for IBM Global Technologies Services in Romania. He has experience in designing and implementing solutions that are based on AIX®, PowerHA®, PowerVM®, General Parallel File System (GPFS), IBM System Storage®, VMware, Linux, Cisco and Brocade SAN, and Cloud Computing. Gheorghe holds a degree in Computer Science and earned various IBM Product Certifications.

Joanna Wong is an Executive IT Specialist for the IBM STG Worldwide Client Centers, focusing on IBM Platform Computing solutions. She has experience in HPC application optimization and large systems scalability performance on x86-64 and IBM Power architecture. Joanna also has industry experience in engagements with Oracle database server and Enterprise Application Integration. Joanna has an Artium Baccalaureatus degree in Physics from Princeton University and a Master of Science degree and a doctorate degree in Theoretical Physics from Cornell University. She also has a Master of Business Administration degree from the Walter Haas School of Business with the University of California at Berkeley.

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

- Ella Bushlovic

  International Technical Support Organization, Poughkeepsie Center

- Kailash Marthi

  IBM US

- Chong Chan
- Lei Guo
- Gord Sissons
- William Lu
- Mehdi Bozzo-Rey

  IBM Canada

- Tarsio Zambrana
- Marcelo Braustein

  IBM Brazil

- Dominic Lancaster

  IBM Australia

Now you can become a published author, too!

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

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

http://www.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 found at this website:

  http://www.ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:

  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm
- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Introduction

IBM Platform Computing provides infrastructure software solutions that help clients succeed and gain competitive advantages. This publication addresses not only the architectural concepts of the platform software family, but focuses on two implementation reference architectures that address the enterprise needs: High performance computing for technical workloads and Big Data analytics.

Throughout this book we use term *Big Data*, which is also referred as BigData, Big data, and big data.

This chapter includes the following topics:

- Why IBM Platform Computing?
- High performance clusters
- IBM Platform HPC implementation scenario
- Big Data implementation on an IBM high performance computing cluster
- IBM Platform Computing solutions and products

1.1 Why IBM Platform Computing?

The data that is generated globally in two days is the same amount of data that was generated in human history since the beginning of writing until 2003. For more information, see the article by Eric Schmidt, *Every 2 Days We Create As Much Information As We Did Up To 2003*, which is available at this website:

http://techcrunch.com/2010/08/04/schmidt-data/

This phenomenon causes data to increase exponentially day after day. Most of this data is user-generated, not organized, which creates a demand on businesses and organizations to deliver an infrastructure that can accelerate time-to-results for compute and data-intensive applications to generate useful and competitive information from this data as quickly as possible. The challenge is to deliver timely and effectively a distributed computing environment that can manage dynamic clusters, grids, and high performance computing (HPC) clouds to handle this data.

Enterprise core business teams are constrained by long processing times and the explosion of data that new initiatives, such as Big Data is creating to transform the huge amount of scattered information that is generated every day in knowledge that helps businesses growth (competitors are using the same approach to increase sales and revenue). At the same time, the infrastructure IT team is attempting to manage costs while maintaining service levels to the line of business. Due to an insatiable need for increased computing power, and applications running in an over-provisioned infrastructure silo with low usage, there is interest in lower-cost x86-based resources, virtualization, high performance computing (HPC) cloud, and the evolving trends towards heterogeneous, multi-core programming models, such as GPUs, to help consolidating these silos and optimizing these applications.

The IBM Platform Computing products and services include middleware and infrastructure management software for mission-critical technical computing, analytics, Big Data, and HPC cloud in distributed computing environments. The Platform Computing portfolio can help IT infrastructure managers to perform the following tasks:

- Increase competitive advantage with faster results and increased throughput.
- Reduce costs by consolidating IT silos and driving usage.
- Embrace complexity of heterogeneous applications, users, and locations.
- Reduce risk with choice of ready-to-use integrated systems or best-of-breed offerings.
- Maximize agility-delivering dynamic clusters, grids, and HPC cloud environments.
- Process relevant data in faster processing times.
- Run data correlation in faster times to obtain better data analytics.

1.2 High performance clusters

Clustered systems are the answer for the infrastructure challenges for businesses and organizations that require to accelerate time-to-results for compute and data-intensive applications. Chapter 2, “High performance clusters” on page 9 highlights the architectural aspects for HPC clusters design and management aspects.

Chapter 2, “High performance clusters” on page 9 includes the following sections:

- Cluster management
- Workload management
- Reference architectures
- Workload optimized systems

1.3 IBM Platform HPC implementation scenario

Chapter 3, “IBM Platform High Performance Computing implementation scenario” on page 67 shows an Implementation of the IBM Platform HPC Suite, which delivers ease of installation to use the features of IBM Platform Cluster Manager, IBM Platform LFS, and IBM Platform MPI, which provide solutions that help the needs of most HPC users.

The cluster implementation scenario takes into account that the application needs intensive, high throughput I/O, low latency communication between compute nodes, high availability, and monitoring.

The steps that are described in Chapter 3, “IBM Platform High Performance Computing implementation scenario” on page 67 help understand the IBM Platform Computing cluster products and the interactions with the different parts of the solution including the following tasks:

- Installing Platform HPC (pHPC) and other software solutions, such as IBM General Parallel File System (GPFS), OFED, and independent software vendors (ISV) applications.
- Provisioning GPFS server nodes and compute nodes.
- Integrating ISV applications and GPFS with pHPC.
- Submitting jobs by using pHPC GUI.
- Monitoring pHPC cluster and GPFS.

1.4 Big Data implementation on an IBM high performance computing cluster

Big Data is a new generation of technologies and architectures that are designed to economically extract value from large volumes of various data by enabling high velocity capture, discovery, or analysis. In today’s environment, Big Data environments are complex, adding more challenges to the already cumbersome company technology infrastructures. IBM high performance computing solutions with Platform Symphony provides class reference architecture for companies that are seeking to cross-reference data for specific analytics.

The output of the Big Data analysis can generate key insights to improve client experience and focal data for marketing, operations, and finance, among others.

IBM Platform Computing software provides a graphical interface for submitting and monitoring Big Data jobs and integrates easily with IBM GPFS in a reference architecture, as shown in Figure 1-1.

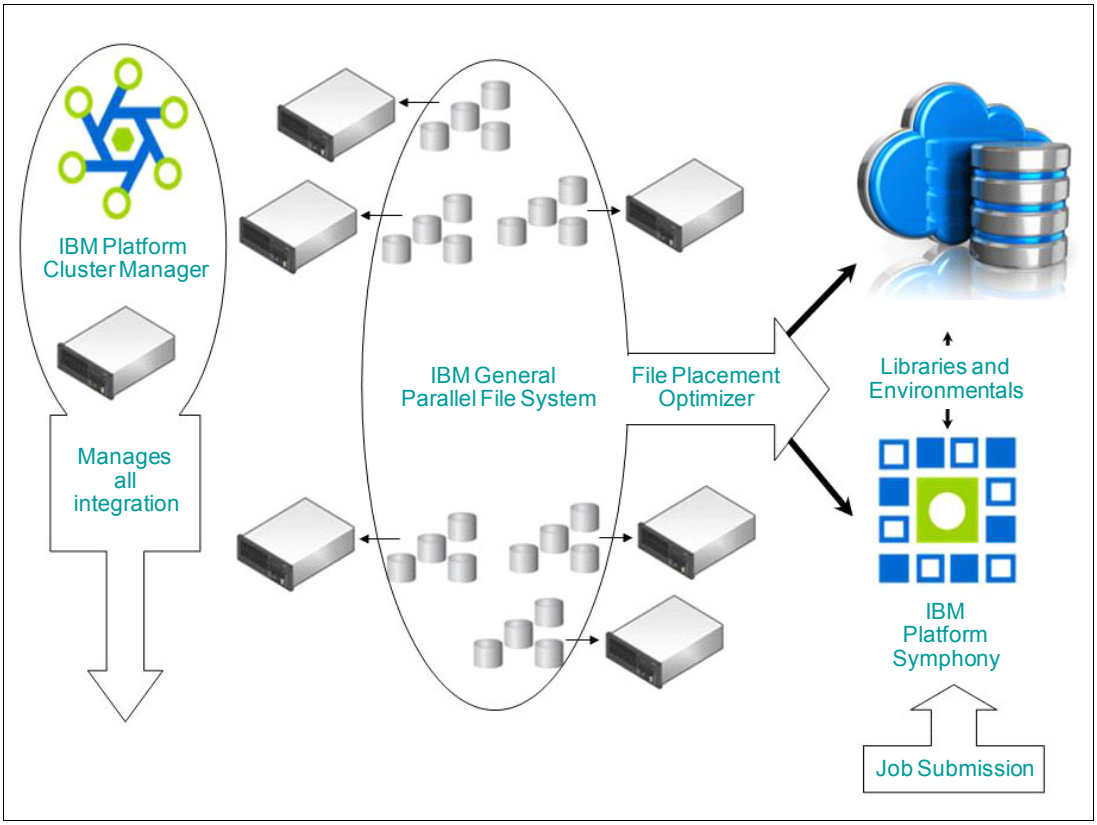

Figure 1-1 Platform Symphony and Hadoop Integration with GPFS FPO

Chapter 4, “IBM Big Data implementation on an IBM High Performance Computing cluster” on page 133 includes the following sections:

- High Performance Computing for Big Data reference architecture.
- High Performance low latency Big Data solution stack.
- Big Data Integration with Platform Symphony GPFS FPO and managing Hadoop.
- Big SQL Server Symphony MapReduce Workload Management.
- Use Platform Cluster Manager SE for the Platform Symphony cluster.
- Deployment considerations for the use case scenario.

1.5 IBM Platform Computing solutions and products

The IBM Platform Computing solutions portfolio includes the following solutions and products:

- IBM Platform LSF

  A powerful and comprehensive workload management family for demanding, distributed, and mission-critical heterogeneous technical computing environments.

- IBM Platform HPC

  Simplified, integrated, complete (cluster and workload management, reporting and MPI) HPC management software that is bundled with systems.

- IBM Platform Symphony

  A powerful high-throughput and low-latency management software family for compute and data-intensive grid applications.

- IBM Platform Cluster Manager

  For provisioning and management of HPC clusters, including self-service creation and optimization of heterogeneous clusters by multiple user groups.

- HPC Cloud Suite

  A software suite to simply and efficiently manage workload-optimized clusters, grid, or HPC cloud environments.

- Global Parallel File System (IBM GPFS)

  A fast, simple, scalable, and complete storage solution for today's data intensive enterprise.

Note: For more information about IBM Platform Computing products, see this website:

http://www-03.ibm.com/systems/technicalcomputing/platformcomputing/products/index.html

THIS PAGE INTENTIONALLY LEFT BLANK

2

High performance clusters

Today's computing infrastructure is now being used increasingly as a cost-effective way to provide scalable, high-performance, high-available solutions for various workloads. For many, clustered systems are the answer. This chapter highlights the architectural aspects for high-performance computing (HPC) clusters design and management aspects. It also includes some of the major features of IBM Platform Computing products and the way these features can help to address the challenges of HPC clusters.

This chapter includes the following sections:

- Cluster management
- Workload management
- Reference architectures
- Workload optimized systems

2.1 Cluster management

IBM Platform Computing products provide a focused technical computing management software portfolio for clients who are looking for simplified, high-performance, and agile systems workload and resource management. For cluster management, we can choose between two products that are based on business requirements and functionalities: IBM Platform HPC or IBM Platform Cluster Manager. There are several differences between these cluster management products and their editions.

The differences between Platform Cluster Manager Standard Edition (PCM-SE) and Platform HPC including xCAT (Extreme Cloud Administration Toolkit) are shown in Table 2-1.

Table 2-1 Comparison of xCAT, Platform Cluster Manager Standard Edition, and Platform HPC

Features	xCAT	PCM-SE	Platform HPC
Linux x64 support	X	X	X
AIX support	X		
Windows support	X		
Linux support	X	X	
Scalability	10,000+	2,500	200-300
Hardware management	X	X	X
Hardware and system monitoring	Third-party agents	X	X
Management portal		X	X
One-step installation		X	X
Easy of use (provision template and so on)		X	X
Third-party software kit (ICR, OFED and so on)		X	X
Workload management			X
Platform MPI			X
Job management portal			X
Open source	X		
Commercial support	Optional	X	X

xCAT is an open source software package for cluster management and offers support for different operating systems with good scalability. xCAT can support clusters of more than 10,000 nodes. However, one draw back is that xCAT requires a third-party monitoring agent for the hardware and system monitoring.

PCM-SE supports only Linux x86 and Linux. In terms of scalability, PCM-SE in its current version can scale up to 2,500 nodes via the GUI. The added functionalities in PCM-SE are centralized web-interface and the Web Portal, which makes the product easy to manage and use to manage a complex cluster as a single system. PCM-SE can provision the operating system with the software components and allows administrators to define provisioning templates for ease of software package management for cluster nodes.

The kits framework allows third-party users to package multiple software components (such as InfiniBand drivers and GPU runtime software) with the configuration, then deploy them into the cluster nodes. The PCM-SE installation is quick and easy. It is a one-step installation that installs all of the required components (including embedded xCAT as a provisioning engine), all dependant packages, and completes the postinstallation that includes configuring services.

IBM Platform HPC (pHPC) is targeted to small and medium clusters. The scalability is around 200 - 300 compute nodes. pHPC has most of the PCM-SE functionality (pHPC supported only Linux x86). PCM-SE is a part of Platform HPC (providing cluster management functionality) and works seamlessly with the intelligent workload scheduler that is based on Platform LSF and Platform MPI and Platform Application Center (PAC) that are bundled in Platform HPC. PCM-SE also has the job management portal, which allows the users to submit and manage their jobs. Together, they deliver a complete set of cluster management functions for technical computing users.

IBM Platform Cluster Manager has two editions: Standard Edition and Advanced Edition. From a functionality perspective, there are some differences between Platform Cluster Manager Standard Edition (PCM-SE), and the Platform Cluster Manager Advanced Edition (PCM-AE), which are listed in Table 2-2.

Table 2-2 IBM Platform Cluster Manager Standard Edition versus Advanced Edition

Features	**PCM-SE**	**PCM-AE**
Physical provisioning	X	X
Server monitoring	X	X
Hardware monitoring	X	-
IBM Platform HPC integration	X	-
Kit framework for software deployment	X	-
VM provisioning	-	X
Multiple cluster support	-	X
User self-service portal	-	X
Storage management	-	X
Network management	-	X
Cluster definitions for whole cluster deployment	-	X
Multiple tenants support	-	X
Supported environments	**PCM-SE**	**PCM-AE**
IBM Platform LSF family	X	X
IBM Platform Symphony family	X	X
Other workload managers	X	X

As you can see in Table 2-2 on page 11, PCM-SE manages a static computing cluster for a group of users. This is a single cluster; it is static and has a single user group compared with PCM-AE, which manages a dynamic cluster with multi-tenant user groups. This feature is a significant difference because PCM-AE manages a dynamic cluster. The second difference is that PCM-AE manages a multi-tenant environment. The third differentiator is that PCM-AE also manages virtualized environments.

PCM-SE supports the non-server hardware monitoring, which is not supported in PCM-AE. In the PCM-SE, there is a kit framework that is designed for software deployment. On the other side, PCM-AE has the cluster definitions for cluster deployments. These two mechanisms are different, although they are trying to reach the same goal.

PCM-SE is also integrated with pHPC while PCM-AE has no direct integration. You must treat pHPC as a software layer in the cluster deployments. Both PCM-SE and PCM-AE can work with the Platform LSF family, the Platform Symphony family, and non-IBM platform workload management environments.

From the architecture point of view, we should design and add by default PCM-SE into all opportunities that require cluster management functions. This is because we discovered that PCM-SE can match most of the use cases. We consider only PCM-AE if the opportunities require the following two cases:

- The first case is that the cluster is dynamic, which means the size of the cluster constantly changes over time or the servers are constantly shared between clusters (this is in a multiple cluster situation) or the cluster are provisioned and not provisioned over time.
- The second case to consider PCM-AE is that there are some virtual machines in the clustered environment, which means the hypervisor must be deployed in the cluster.

2.1.1 IBM Platform HPC

Clusters that are based on open source software and the Linux operating system dominate HPC. This is due in part to their cost-effectiveness and flexibility and the rich set of open source applications available. Platform HPC (pHPC) provides a complete set of technical and high performance computing management capabilities of Linux clusters in a single product. System administrators can use Platform HPC to manage complex cluster as a single system by automating deployment of the operating system and software components. Platform HPC provides provisioning and maintenance capabilities. It also includes centralized monitoring with alerts and customizable alert actions.

Platform HPC includes the following features:

- Cluster management (embedded xCAT as the provisioning engine)
- Workload management (based on IBM Platform LSF Express)
- Workload monitoring and reporting
- System monitoring and reporting
- Robust commercial MPI Library (based on IBM Platform MPI Standard Edition)
- Application support (integrated application scripts/templates)
- Accelerator support, including GPU and Intel XeonTM Phi coprocessor scheduling, management, and monitoring
- High availability of the pHPC cluster environment
- Unified Web Portal

Use cases for Platform HPC

IBM Platform HPC allows technical computing users in industries such as manufacturing, oil and gas, life sciences, and higher education to deploy, manage, and use their HPC cluster through an easy to use web-based interface. This minimizes the time that is required for setting up and managing the cluster for users and allows them to focus on running their applications rather than managing the infrastructure.

IBM Platform HPC comes complete with job submission templates for ANSYS Mechanical, ANSYS Fluent, ANSYS CFX, LS-DYNA, MSC Nastran, Schlumberger ECLIPSE, Simulia Abaqus, NCBI Blast, NWChem, ClustalW, and HMMER. By configuring these templates that are based on the application settings in your environment, users can start using the cluster without writing scripts. Cluster users who deploy home-grown or open source applications can use the Platform HPC scripting guidelines. These interfaces help minimize job submission errors and are self-documenting, which enables users to create their own job submission templates.

Platform Application Center (PAC) Integration: Platform LSF add-ons are not included in Platform HPC and not installed with it. The add-on must be downloaded and installed separately. Platform HPC contains some functions of PAC (job submission, job management, and application templates). If a customer purchases PAC Standard, they receive the entitlement. By applying the entitlement to existing Platform HPC, some other functions (remote 2D and 3D visualization) are enabled. However, the rest of the PAC Standard functions exist in PAC binary only. Therefore, if the customer requires these functions (specifically, Role Based Access Control), they must install PAC separately.

Component model

Platform HPC software components support various computationally intensive applications that are running over a cluster. To support such applications, Platform HPC software components that are shown in Figure 2-1 must provide several services.

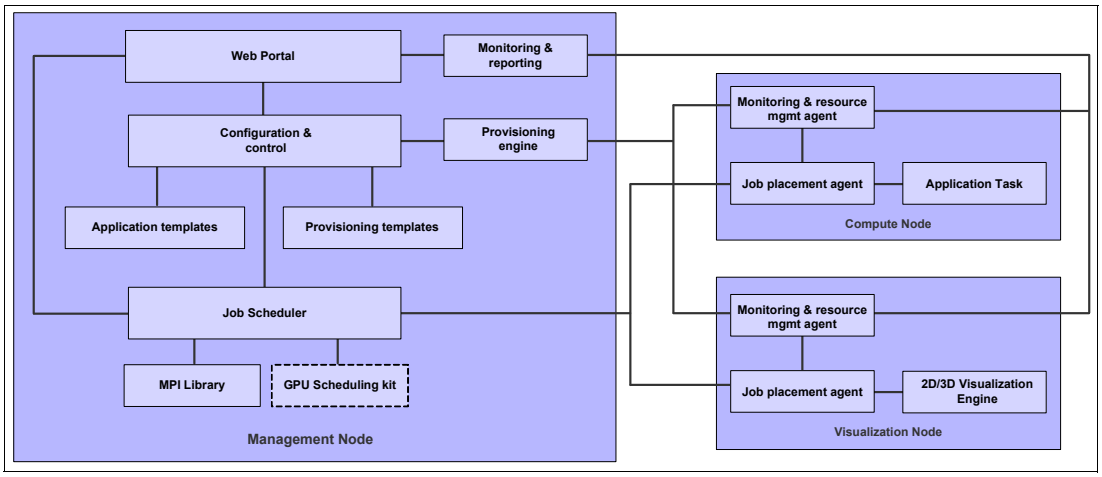

Figure 2-1 Platform HPC software components diagram

Before starting any software applications, all of the nodes must be installed with the operating system and any application-specific software. This function is provided by the provisioning engine. Here, the user creates or uses a predefined provisioning template that describes the wanted characteristics of the compute node software. This provisioning engine listens for boot requests over a selected network and installs the system with the wanted operating system and application software. After the installation is complete, the target systems are eligible to run applications.

Although the compute images can run application software, access to these images is normally controlled by the job scheduler (Platform LSF) that is running as a workload manager. This scheduler function ensures that computational resources on the compute nodes are not overused by serializing access to them. The properties of the job scheduler are normally defined during installation setup. The scheduler can be configured to allocate different workloads to be submitted to one of the job placement agents (Platform LSF agents). This job placement agent starts particular workloads at the request of the job scheduler. There are multiple job placement agents on the system, one on each of the operating system images.

The monitoring and resource agents report back to the provisioning agent and job scheduler about the state of the system on every operating system image. This provides a mechanism to provide alerts when there is a problem, and to make sure that jobs are only scheduled on operating system images that are available and have resources.

The web portal provides an easy-to-use mechanism for administrators to control and monitor the overall cluster, while for the users it provides easy-to-use access to the system for job submission, management and reporting.

Operational model

The sample high available environment that is shown in Figure 2-2 is used to show how to design a deployment of the Platform HPC cluster. This is only one of several possible configurations. In our sample, there are four networks (public, provisioning, management, and application) and one shared cluster storage that is supplemented with a two-node GPFS cluster.

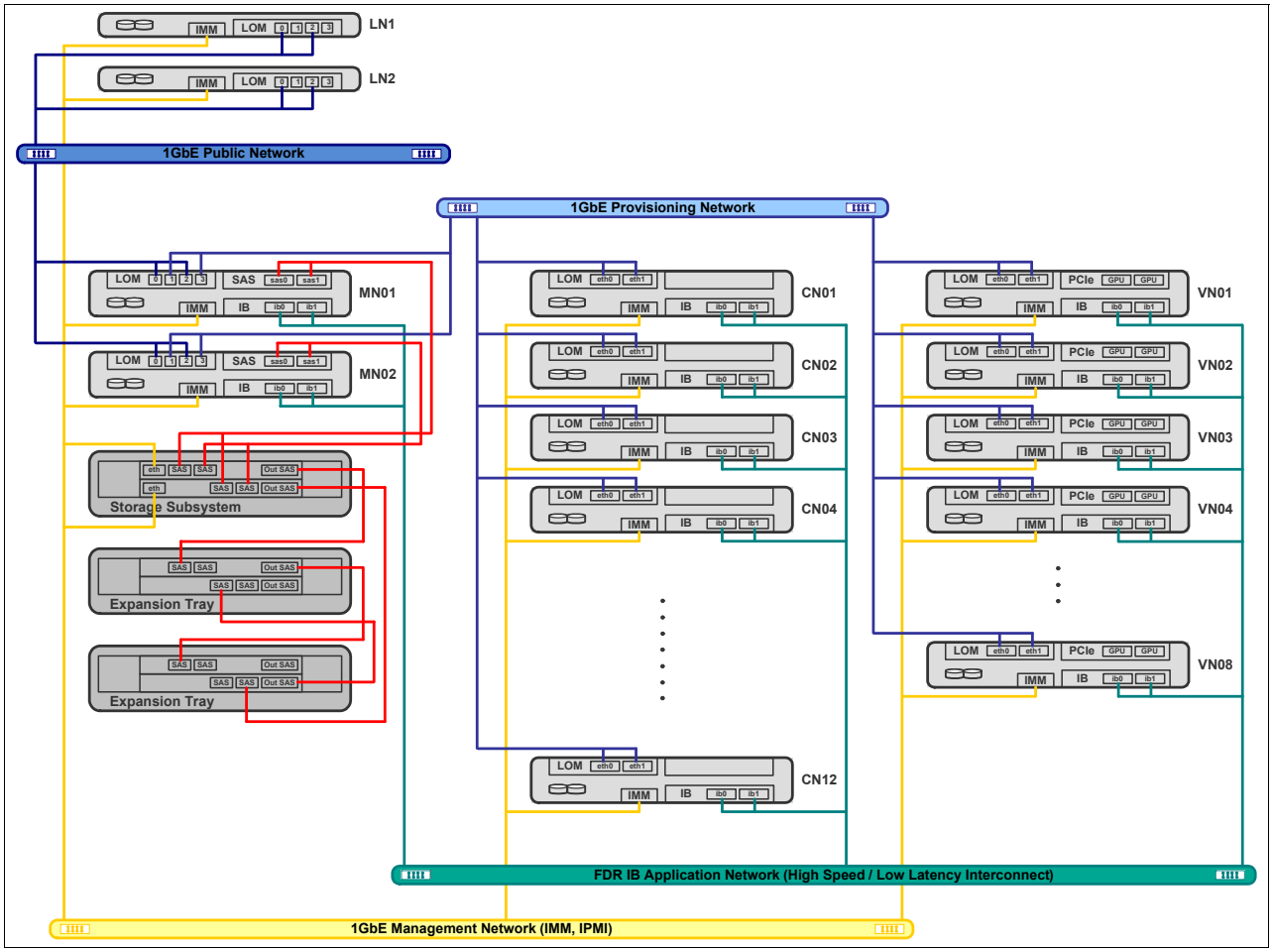

Figure 2-2 Platform HPC cluster deployment on the physical hardware

Cluster nodes

Management nodes, compute nodes, and visualization nodes can be used in the Platform HPC cluster. Each node has its own role.

Management node

A management node is the first node that is installed in your cluster. Every cluster requires a management node. It controls the rest of the nodes in the cluster. In previous versions of pHPC, this node is also called the *head node* or *master node*. A management node acts as a deployment node at the user site and contains all of the software components that are required for running the application in the cluster. After the management node is connected to a cluster of nodes, it provisions and deploys the compute nodes with client software. The software that is installed on the management node provides the following functions:

- Administration, management, and monitoring of the cluster
- Installation of compute nodes

- Stateless and stateful management
- Repository management and updates
- Cluster configuration management
- HPC kit management
- Provisioning templates management
- Application templates management
- Accelerated parallel application processing and application scaling by using the Platform MPI kit
- Workload management, monitoring, and reporting by using the Platform LSF kit
- User logon, compilation, and submission of jobs to the cluster
- Acting as a firewall to shield the cluster from external nodes and networks
- Acting as a server for many services such as DHCP, TFTP, HTTP, and optionally DNS, LDAP, NFS, and NTP

Compute node

Compute nodes are designed for computationally intensive applications to satisfy the functional requirements of planned use cases. The compute node is provisioned and updated by the management node and performs the computational work in a cluster. The workload management system (Platform LSF) sets the number of job slots on a compute node to the number of CPU cores. After the compute node is provisioned, it is installed with the operating system (OS) distribution, the Platform LSF kit (workload manager agent, monitoring, and resource management agent), the Platform MPI kit, and other custom software (as defined by user). The compute node can have some local disk for the OS and temporary storage that is used by running applications. The OS might also be configured as booted on diskless system to improve I/O performance (by using stateless provisioning).

The compute nodes also mount NFS or can be configured with GPFS for shared storage. These compute nodes can cooperate in solving problem by using MPI. This is facilitated by the connectivity to a high-speed interconnect network. Some applications do not require large disk storage on each compute node during simulation. However, large models might not fit in the available memory and must be solved out-of-core and then can benefit from robust local storage.

Visualization node

The visualization node is the same as a compute node, except it contains one or more graphics processing units (GPUs) for rendering 3D graphics, computer-aided engineering (CAE) design, validation of product parts with dynamic simulations, or stress analysis on individual components. Depending on the applications, each GPU can support several simultaneous interactive sessions. The pre- and post-processing applications are mostly serial; therefore, the processor resources in the node should be sufficient to handle their computation requirements. The visualization nodes often have some local disk space for the OS and temporary storage use by running applications. The visualization nodes also mount NFS or GPFS file systems for shared storage.

Login node

The login node functions as a gateway into the cluster. When users want to access the cluster from the public network, they must first log in to the login node before they can log in to other cluster nodes. In general, we recommend this as a best practice to prevent unauthorized access of the management node.

Cluster networks

There are several networks that are used in a pHPC cluster. Each cluster might have a dedicated network or might share a common network with others.

Public network

A public network connects the pHPC cluster to a corporate network.

Provisioning network

A provisioning network (private network) is an internal network to provision and manage the cluster nodes. The provisioning network cannot be accessed by nodes on the public network. The provisioning network often is a Gigabit Ethernet network. In general, the provisioning network serves the following purposes:

- Cluster management and monitoring
- Workload management and monitoring
- Message passing

It is common practice to perform message passing over a much faster network by using a high-speed interconnect with low latency. For more information, see “Application network” on page 17.

Management network

The management network (BMC network) is a network that provides out-of-band access to cluster nodes for hardware management. The network provides access to the CMM and the IMM of each cluster node. The management network cannot be accessed by nodes on the public network. (If public access is needed, the switch for the public network can be configured to enable routing between the public and management networks.)

Application network

This network (compute network) is used mainly by applications (for example, MPI applications) to efficiently share data among different tasks within an application across multiple nodes. This network is often used as a data path for applications to access the shared storage. The application network uses a high-speed interconnect, such as 10 Gb/40 Gb Ethernet or QDR/FDR InfiniBand. If the pHPC cluster includes a visualization node, there must be a route to the compute network from the external network. This routing is not necessary (except to the management node) if the system is intended only for batch work. It is possible to combine these networks by using virtual local area networks (VLANs).

These cluster networks can be combined into one or two physical networks to minimize the network cost and cabling in some configurations.

A typical combined deployment can be one of the following examples:

- Combined management network and provisioning network, plus a dedicated high-speed interconnect for applications. This is often the case if the high-speed interconnects are InfiniBand.
- Combined provisioning network and application network by using 10-Gigabit Ethernet, plus a dedicated management network. This network architecture can be implemented when management work has a dedicated switch on the chassis.

Both combined deployment options are available and supported by the pHPC cluster.

Cluster storage and file system

The following types of file systems are supported by the pHPC cluster:

- NFS: This file system is recommended for applications that are not I/O-intensive. The storage is connected to the management node, which acts as an NFS server.
- IBM General Parallel File Systems (GPFS): This file system is recommended for I/O-intensive applications. In this case, the management node acts as the GPFS server.

If there is an external storage system that contains data that is required by the application, the following supported access methods are available:

- Connect the storage to the cluster private network (that is, the application network). This method allows applications that are running on compute nodes to access the storage. This method should be the only option if the application requires constant changes to the files and the application performance is heavily dependent on the performance of accessing those files. If the data is stored in a database, this option should also be used.
- Connect the storage to the management node. When a job requires access to certain files, these files must be named explicitly during the job submission time. These required files are copied automatically to the compute nodes as part of the job scheduling process before the job starts. This is a viable option if the file size is small (less than 100 MB) and the data is not stored in a database. Similarly, the output files the job creates can be transferred automatically back to the management node after the job is completed.

NFS

If there is no external shared storage for the pHPC cluster, the local storage on the management server (including SAS attached disk arrays) provides the shared file system. The management node is configured and sized with enough resources to simultaneously allow file serving and other management functions. For many use cases, NFS access is sufficient to support the workload. For better performance, the external shared storage can be connected to the management node that is based on the system host connectivity option via Fibre Channel by using SAN.

GPFS

When management nodes (MN01 and MN02) are configured as a high available environment, shared storage is required to share user home directories and system working directories. We recommend building over these management nodes a two-node GPFS cluster with tiebreaker disks. We create a GPFS cluster by using two quorum nodes that are also the storage (NSD server) nodes. All the remaining compute nodes are NSD clients and do not participate in GPFS cluster quorum voting (non-quorum nodes).

Cluster NFS: In addition to the traditional exporting of GPFS file systems by using the Network file system (NFS) protocol, the use of GPFS on Linux allows you to configure a subset of the nodes in the cluster to provide a highly available solution for exporting GPFS file systems by using NFS. The participating nodes in this case acts as a GPFS client and are designated as Cluster NFS (CNFS) member nodes and the entire setup is frequently referred to as CNFS or a CNFS cluster.

High availability

A high availability cluster minimizes downtime by providing one active management node (MN01) and one standby management node (MN02). Services are only run on the active management node. If at any point a service stops or quits unexpectedly, the service is restarted on the same node. When a failover process occurs, the standby management node takes over as the management node and all the running services.

Virtual IP addresses

For the services to switch nodes, service access points are defined to enable the high availability process. A service access point defines a virtual IP address that is used by the active management node to access HPC services. In a failover, the active management node also takes over the virtual IP address. A virtual IP address for the public network and a virtual IP address for the provisioning network must be defined in the IP address ranges of your networks.

Shared file system

Shared file systems are required to set up a high availability environment on pHPC. Shared file systems are used to store user and system work data. In a high availability environment, all shared file systems must be accessible by the provisioning network for both management nodes and compute nodes.

2.1.2 IBM Platform Cluster Manager Standard Edition

IBM Platform Cluster Manager Standard Edition (PCM-SE) is easy-to-use, powerful cluster management software for technical computing users. PCM-SE 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 provisioning and maintenance of a cluster. It includes support for RedHat Enterprise Linux family operating systems for x86 64-bit and IBM POWER®.

By using the centralized user interface, system administrators can manage complex clusters as a single system and flexibility as users can add customized features that are based on specific requirements of their environment. It provides a Kit framework within an x86 ecosystem for easy software deployment, such as InfiniBand drivers and GPU runtime software.

PCM-SE provides monitoring capability for most components within a cluster so users can easily visualize the performance and condition of the cluster. The monitoring agent is the same technology that is used in IBM Platform LSF and IBM Platform Symphony and is easy to extend and customize. It also can monitor non-server components, such as chassis, network switches, IBM GPFS, GPU and co-processors, and customized devices for efficient usage of the overall infrastructure. It also adds management node automatic failover capability to ensure continuity of cluster operations.

By using xCAT technology, PCM-SE offers greater management scalability by scaling up to 2,500 nodes via the GUI. PCM-SE runs on various types of IBM servers that include the most recent iDataPlex® servers, NextScale servers, FlexSystem nodes, and System x rack-based servers. It is also supported on industry-standard non-IBM x86 hardware.

PCM-SE includes the following features:

- Quick and easy installation
- Cluster management (embedded xCAT as the provisioning engine)
- Kit framework that is designed for software deployment and maintenance
- Robust and scalable system monitoring and reporting
- Centralized Web Portal
- Cross-provisioning compute nodes

Use cases for PCM-SE

Typical use of PCM-SE is with HPC workload managers, such as IBM Platform LSF, IBM Platform Symphony, Oracle Grid Engine, PBS, Maui/Moab, and Hadoop. Its scalability is used as a commercially supported cluster manager to manage Big Data clusters, large HPC clusters, and scale out application appliances.

Component model

The PCM-SE software components that are shown in Figure 2-3 are based on Extreme Cloud Administration Toolkit (xCAT), which provides a unified interface for hardware control, discovery, and operating system and software components deployment. The back-end PCM-SE features are coded as xCAT Plug-ins complementing xCAT as a provisioning and management foundation. xCAT plug-ins store cluster configuration and user settings for the cluster inside PostgreSQL DB as a Platform Cluster Manager database (PCM DB). The cluster monitoring data is also stored in a PCM DB for reporting and analysis. The PERF service collects and aggregates performance data from the compute nodes. Other agentless monitoring data is loaded to the PCM DB by PERF data loaders.

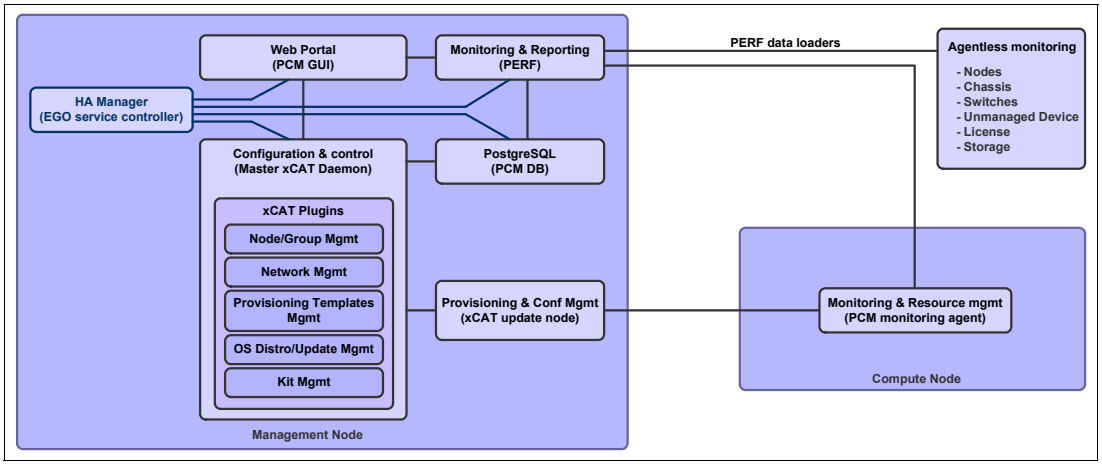

Figure 2-3 PCM-SE software components diagram

The Web Portal is the front end of PCM-SE, which provides the following capabilities:

- Resources dashboard for comprehensive view of cluster status (cluster health, cluster performance, and rack view)
- Manage hosts (nodes and node groups), unmanaged devices, licenses, and networks
- Manage provisioning templates through image profiles and network profiles (packages, kit-components, kernel modules, networks, post-install, and post-boot scripts)
- Manage networks
- Manage OS distributions
- Manage kit library
- View and manage resource reports and resource alerts

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

Operational model

From the architecture point of view, the operational model of the cluster by using PCM-SE shares topology with the pHPC cluster operational model. One of the differences is scalability. PCM-SE offers greater management scalability by scaling up to 2,500 nodes, and it can be used to deploy small and large clusters. A small cluster is considered to be a one- rack solution (maximum of 42 - 56 nodes), and a large cluster is considered to be more than a one-rack solution (up to 2,500 nodes). This affects networking design and if you want to run a cluster with more nodes than a single rack can contain, a multiple-rack setup is required. As you can see on a sample of the operational model that is shown in Figure 2-4, spine switches are used to connect top-of-rack (TOR) switches to create a single cluster from multiple machine racks.

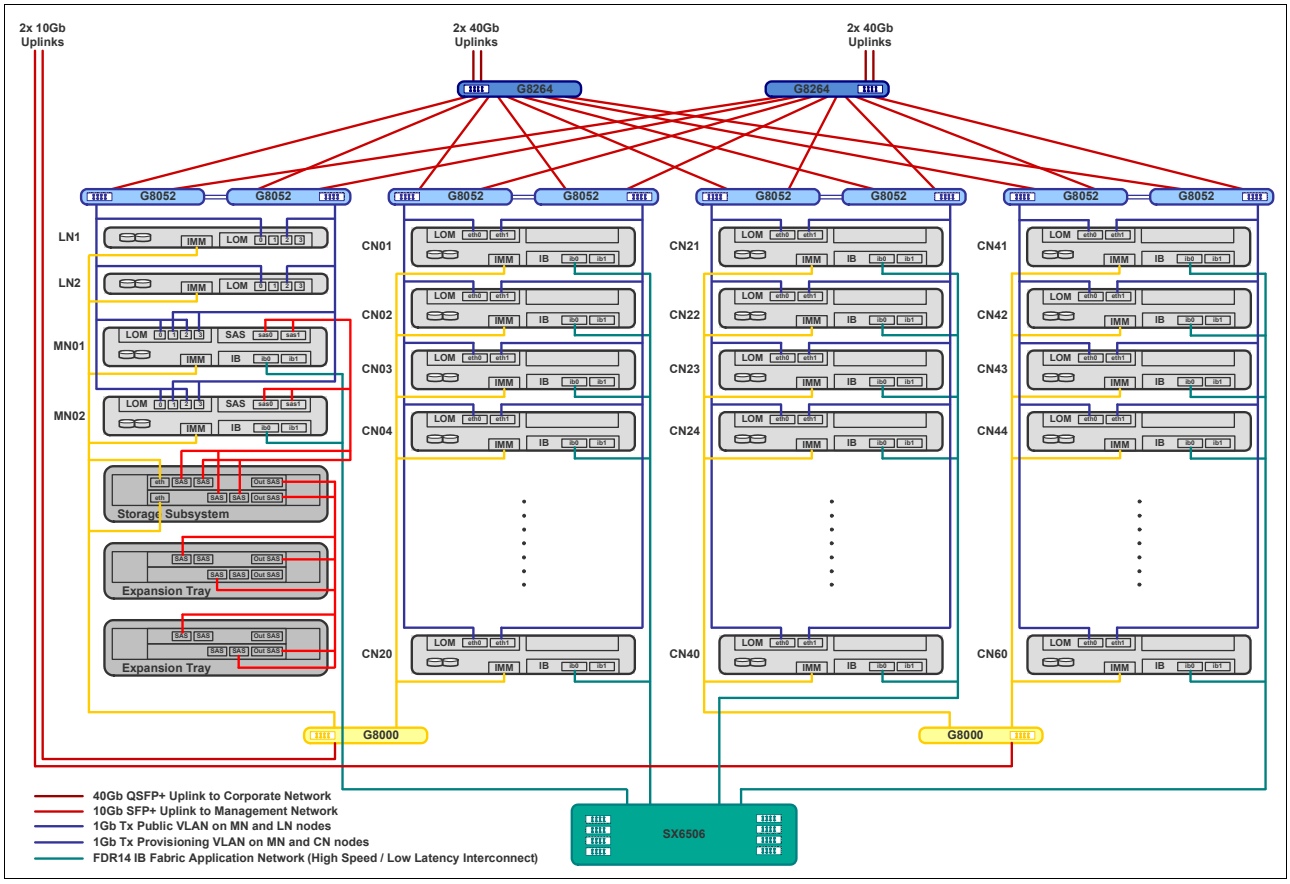

Figure 2-4 PCM-SE cluster deployment on the physical hardware

The operational model for PCM-SE is broken down into five areas: management nodes, log in nodes, compute nodes, shared storage, and networking.

Management nodes

PCM-SE has built-in failover capability for the management node. A high availability environment includes two installed PCM-SE management nodes (MN01 as active and MN02 as standby) locally with same software and network configuration (except the host name and IP address). All IP addresses (management nodes IP addresses and virtual IP address) are in the IP address range of your networks. The management node connects to public and provisioning networks.

Login nodes

As a best practice to prevent unauthorized access of the management nodes, we recommend the use of login nodes (LN01 and LN02) as a gateway into the cluster. The login nodes connect to the public and provisioning networks.

Compute nodes

The compute nodes (CN01 to CN60) are provisioned and updated by the management node and perform the computational work in a cluster. In PCM-SE, node provisioning installs an operating system and applications on a node. To provision a node, associate it with a provisioning template. The provisioning template includes an image profile, a network profile, and a hardware profile. The operating system (OS) distribution that you want to use to provision your nodes can be added to the Web Portal. When the OS distribution is added, two default image profiles are automatically created: one stateful image profile and one stateless image profile.

Stateless provisioning loads the operating system image into memory. Changes that are made to the operating system image are not persistent across compute node reboots. You can use diskless provisioning by RAM-root or by compressed RAM-root.

Stateful provisioning loads the operating system image onto persistent storage. Changes that are made to the operating system image are persistent across compute node reboots. The persistent storage can be a local disk, SAN, or iSCSI device.

Users often install homogeneous clusters where management nodes and compute nodes use the same OS distribution. In some cases, the management node and compute nodes can be different. This is called a *Cross-Distro cluster*. This is an advanced feature that is supported by PCM-SE.

Note: A mix of OS distributions in the same cluster is supported. However, a mix of x86 and Power nodes in the same cluster is not supported.

Shared storage

To create a high available environment, shared storage is required to share user home directories and system working directories. All shared file systems must be accessible by the provisioning network for both management nodes and compute nodes. For I/O-intensive applications, we recommend building a two-node GPFS cluster with tiebreaker disks over both shared storage nodes (MN01 and MN02).

We can improve GPFS performance when InfiniBand is used as a high speed and low latency interconnect. In the following cases, InfiniBand provides two modes to help increase performance:

- GPFS cluster management can use IP over InfiniBand (IPoIB).
- GPFS Network Shared Disk (NSD) communication can use Remote Direct Memory Access (RDMA) InfiniBand protocol.

GPFS can define a preferred network subnet topology; for example, designate separate IP subnets for intra-cluster communication and the public network for GPFS data. This provides for a clearly defined separation of communication traffic and allows you to increase the throughput and possibly the number of nodes in a GPFS cluster. Instead of separate IP subnets for intra-cluster communication (GPFS cluster management and heartbeat), we can use IPoIB.

GPFS on Linux supports an RDMA InfiniBand protocol to transfer data to NSD clients. GPFS has `verbsRdma` and `verbsPorts` options for the RDMA function. The InfiniBand specification does not define an API for that; however, the OpenFabrics Enterprise Distribution (OFED) is a package for Linux that includes all of the needed software (`libibverbs` package as a Verbs API) to work with RDMA. The `verbsRdma` parameter enables the use of RDMA and the `verbsPort` parameter sets the device that you want to use. These parameters are set by using the GPFS **`mmchconfig`** command. For more information about GPFS by using RDMA, see *Implementing the IBM General Parallel File System (GPFS) in a Cross Platform Environment*, SG24-7844, and the General Parallel File System (GPFS) Wiki page, which is available at this website:

https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/General%20Parallel%20File%20System%20%28GPFS%29/page/Network%20Configuration

Networking

The PCM-SE cluster uses the same networks as the p-HPC cluster (public, provisioning, monitoring, and application). Each of the clusters might have a dedicated network or share a common network with others. In a multi-rack cluster, we recommend the use of top-of-rack switches and set up VLANs for different network instead of using one top-of-rack switch per network.

When a server contains two network ports of the same speed, there is a way to tie them together by using Link Aggregation Control Protocol (LACP). Each rack has two top-of-rack switches with HA (by using inter-switch links at the top-of-rack level). This two-switch cluster with configured Virtual Link Aggregation Group (vLAG) feature allows multi-switch link aggregation, which provides higher performance and optimizes parallel active-active forwarding. From each top-of-rack switch, aggregated uplinks might be configured to the up-level spine switches to build redundant two-tier, and layer 3 fat tree network.

2.1.3 IBM Platform Cluster Manager Advanced Edition

Platform Cluster Manager Advanced Edition (PCM-AE) manages the provisioning of multiple multi-tenant analytics and technical computing clusters in a self-service and flexible fashion. PCM-AE provides secure multi-tenancy with access controls, policies, and resource limits to enable sharing. Based on assigned user roles, it provides rapid self-service provisioning of heterogeneous HPC environments and gets the clusters that you need, on physical and virtual resources.

You can deploy an HPC cluster with underlying physical servers composing your cluster infrastructure, or you can make provisioning on top of a virtualization layer on top of the bare metal, or mix both approaches to create a hybrid cluster. You can maximize the consolidation level of your infrastructure as a whole with virtualization, or you can isolate workloads by engaging only physical servers.

PCM-AE helps decrease operating costs by increasing the usage of pooled resources and the operational efficiency (managing multiple separate clusters through a single point of administration). It provides elasticity where the size of user's logical cluster can be dynamically expanded and shrunk over time based on workload and resource allocation policy. PCM-AE runs on various types of IBM servers that include the most recent iDataPlex servers, NextScale servers, FlexSystem nodes, and System x rack-based servers. It is also supported on non-IBM industry standard x86_64 hardware.

The PCM-AE environment includes the following components:

- User self-service and administration portal.
- The management server, which is responsible for running the system services and managing provisioned clusters.
- The xCAT provisioning engine, which provisions clusters with physical machines. The provisioning engine is responsible for managing the physical machines that make up provisioned clusters.
- A database to store operational data. You can specify PCM-AE to install a new MySQL, or you can use an existing MySQL or Oracle database.
- Physical machines, which are the compute nodes within a cluster.

Optionally, PCM-AE includes the following components:

- Hypervisor hosts, which run and manage virtual machines (VMs). When you are provisioning clusters with VMs, the hypervisor hosts provide the actual virtual resources that make up the clusters.
- A Lightweight Directory Access Protocol (LDAP) server for user authentication in a multi-tenant environment.
- An IBM General Parallel File System (GPFS) server for secure storage by using GPFS with PCM-AE.
- A Mellanox Unified Manager (UFM) server for an InfiniBand secure network with PCM-AE.

Use cases for PCM-AE

PCM-AE is part of a family of cluster and grid workload management solutions. PCM-AE is an enabling technology that is used to provision the cluster and grid workload managers on a shared set of hardware resources. The customer can run multiple, separate clusters and includes almost any combination of IBM and third-party workload managers.

Typical use of PCM-AE is with HPC workload managers, such as IBM Platform LSF, IBM Platform Symphony, IBM Platform Symphony MapReduce, IBM InfoSphere® BigInsights™, IBM InfoSphere Streams, Oracle Grid Engine, Altair PBS Professional, Hadoop, and others. For its capabilities, it is used as a commercially supported cluster manager to manage Big Data clusters and multi-tenant HPC clouds.

Component model

PCM-AE has its own particular internal software components architecture, and it makes use of other software components to create a manageable PCM-AE cluster infrastructure environment. Figure 2-5 shows the software components of a PCM-AE environment. We can classify them in two distinct components: PCM-AE internal software components and PCM-AE external software components.

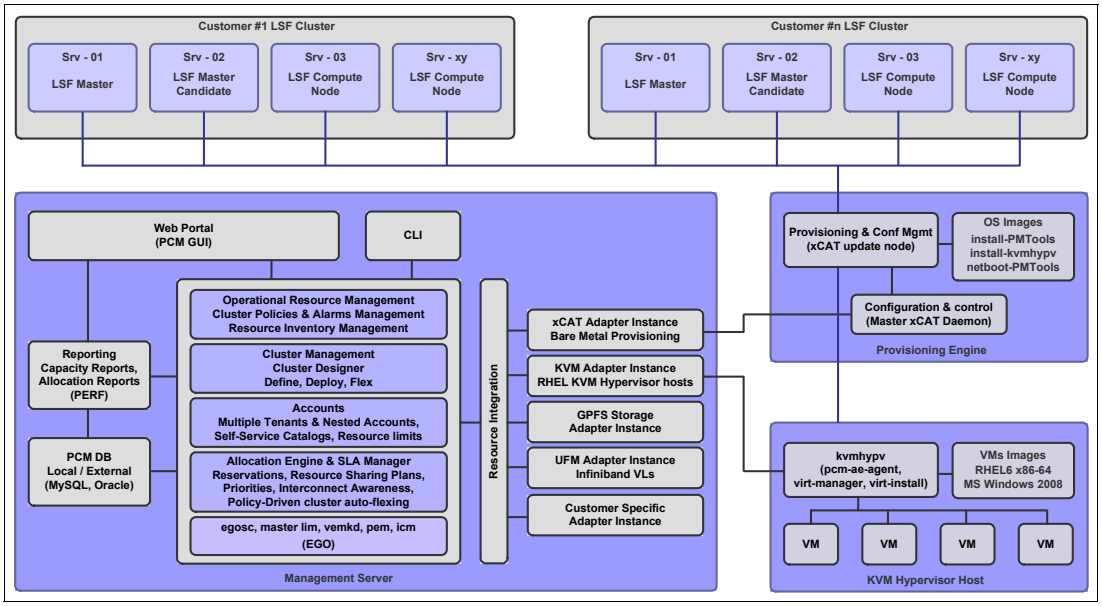

Figure 2-5 PCM-AE software components diagram

Internal software components are based on Enterprise Grid Orchestrator (EGO), which provides the underlying system infrastructure to control and manage cluster resources. EGO manages logical and physical resources and supports other software components that are in the product. PCM-AE features the following components:

- To control the multi-tenancy characteristic of PCM-AE based on accounts
- One that defines the rules for dynamic cluster growth or shrinking based on required service level agreements
- An allocation engine component that manages resource plans, prioritization and how the hardware pieces are interconnected
- To provide feedback on cluster utilization and resources that are used by tenants
- Handles overall operational resource management
- One with which you can define, deploy, and modify clusters
- To visualize existing clusters and the servers within them

The resource integrations layer allows PCM-AE to use external software components to provide resource provisioning. The integrations layer manages machine or virtual machine allocation, network definition, and storage area definition. As shown in Figure 2-5, we can use xCAT to perform bare metal provisioning of servers with Linux with dynamic VLAN configuration.

KVM also is an option as a hypervisor host that is managed by PCM-AE. Another external software component, GPFS, can be used to provide shared storage that can be used, for example, to host virtual machines that are created within the environment.

PCM-AE also can manage and provision clusters by using the Unified Fabric Manager platform with InfiniBand. PCM-AE can be used to provide a high-speed and low-latency private network among the servers while maintaining multi-tenant cluster isolation by using virtual lanes (VLs). The PCM-AE is a platform that can also offer support for the integration of other provisioning software (IBM SmartCloud® Provisioning, VMware vSphere with ESXi), including custom adapters that you might already have or need in your existing environment.

Operational model

The PCM-AE architecture provides you with the benefit and flexibility of dynamically creating HPC clusters that can be expanded later or reduced based on workload demand, or even destroyed after temporary workloads are run. PCM-AE provides cluster management that allows the provisioning of multiple clusters (supported on physical or virtual machines), which feature a self-service for minimal administrator intervention. Figure 2-6 shows how the technology infrastructure components might be set up for this use case.

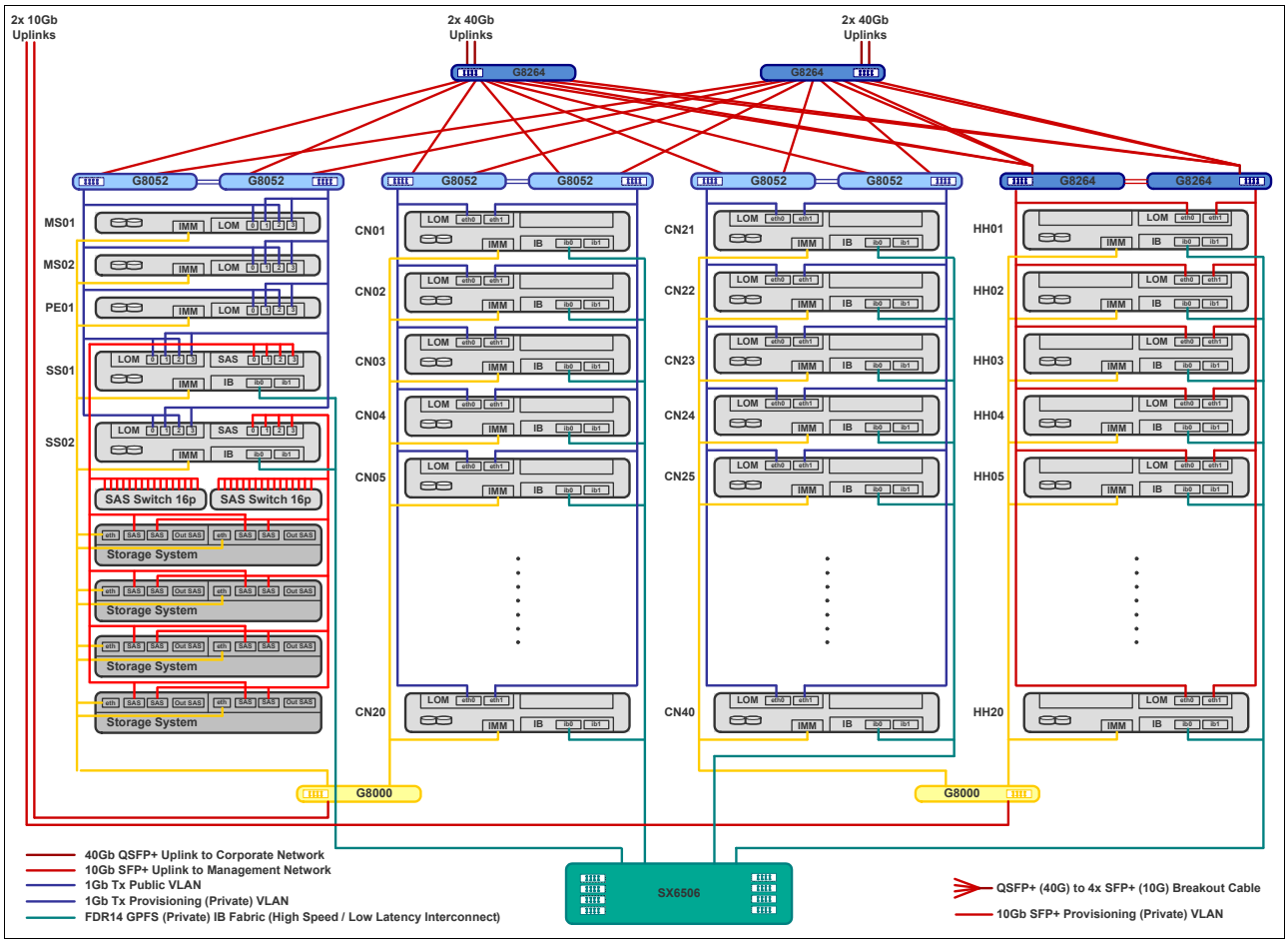

Figure 2-6 PCM-AE cluster deployment on the physical hardware

The operational model for PCM-AE is broken down into five areas: management nodes, provisioning engine, hypervisor hosts, compute nodes, shared storage, and networking. Each of these cluster components is described next.

Management nodes

To improve performance and reduce the load of using a single host, we recommend creating a multi-hosts environment and installing the management server and provisioning engine packages on separate hosts. Within the PCM-AE cluster, there is only one master management server (MS01). However, you can have more management servers for failover. This means that if the master management server fails, the system restarts on another management server that is called master candidate (MS02).

For failover to work, install the management server on each management server candidate host and configure each host to access the same shared storage location as the master management server. In that case, the PCM-AE cluster administrator account must exist as the same user on all management server candidates. The management server host is responsible for running the system services and managing provisioned clusters. If you are managing RHEL KVM hosts, the management server communicates directly with the agent that is installed on each hypervisor host. The management node connects to public and provisioning networks.

You can choose for the management server installation package to automatically deploy an Oracle Database XE. However, you can optionally use a remote database (where an Oracle database is a separate server from the management server). A separate external database is ideal for a multi-host PCM-AE environment because it allows for larger scalability for an enterprise-level database to store operational data. To support management server failover, your Oracle database cannot be on the same host as your management server.

Provisioning engine

The provisioning engine (PE01) provisions clusters with physical machines and is responsible for managing the physical machines that make up provisioned clusters. In a multi-host environment (typically, if you plan to work with many clusters in a larger environment, such as for a production environment), we recommend the use of a dedicated host for each type of management. The provisioning engine contains node groups that are templates that define how a group of machines (nodes) is configured and what software is installed. The PMTools node group is used for provisioning physical machines, while the KVM node group is used for provisioning hypervisor hosts on physical machines. The provisioning engine connects to public and provisioning networks.

Hypervisor hosts

Hypervisor hosts (HH01 to HH20) run and manage virtual machines (VMs) within a cluster. When clusters are provisioned with VMs, hypervisor hosts provide the actual virtual resources that make up the cluster. PCM-AE requires specific prerequisites for the physical machine that is used as the hypervisor host. In general, any physical x86_64 powerful machine can be used that supports RHEL (64-bit) with the KVM kernel module installed and with mounted NFS or LVM-based storage with enough disk space for the template configuration files and VM images. The amount of disk space that is required depends on the size of the VM guest operating system (the operating system that runs on the VM) and how many VMs are in the hypervisor host. The hypervisor hosts connect to the provisioning network and to the application network (high speed and low latency interconnect), if required.

Compute nodes

Compute nodes (CN01 to CN40) that are presented by physical machines are the physical resources of the shared compute infrastructure within the PCM-AE cluster. Compute nodes connect to the same private network as the provisioning and to the application network (high speed and low latency interconnect), if required.

Adding compute nodes into the xCAT adapter instance makes the nodes available for provisioning. PCM-AE is tightly integrated with Platform LSF to allow the provisioning of multiple Platform LSF clusters on demand. You can quickly deploy a complete Platform LSF cluster by creating a cluster instance from the sample Platform LSF cluster definition. You can customize the cluster definition, including rule-based policies to suit your own specific computing environment. Cluster policies help the system to deliver the required cluster environments and provide workload-intelligent allocations of clusters. Policies balance the supply and demand of resources according to your business requirements and allow you to change the available capacity (up or down), depending on workload requirements.

Note: In a secure multi-tenant environment, the tenant's users might not have access to the LAN and to their cluster machines. For example, they cannot use SSH to log in to a machine. To allow user access, an administrator must configure the network accordingly. For example, add to the compute nodes another network configuration to the public VLAN that users can connect through.

Shared storage

A shared storage repository is required to benefit from various features, such as high availability, load balancing, and migration. For failover to work, each management server host is configured to access the same shared storage location for failover logs as the master management server. To support the failover, the NFS or GPFS file system can be used as a shared file system that is connected to the private network.

If you want to share data between physical machines (for example, Platform LSF hosts), connect a shared file system to the provisioning network. By default, the system is configured to use NFS on the master management server. In the case of the PCM-AE multi-host environment, we recommend building a two-node GPFS cluster with tiebreaker disks (SS01 and SS02) as a shared storage.

If you are using hypervisor hosts to provision virtual machines (VMs), the PCM-AE supports VMs by using NFS-based file systems for storage. By default, template configuration files are stored on the master management server, VM configuration files are stored locally on the RHEL KVM hypervisor host, and VM images are stored on the NFS server. By using an NFS or GPFS as a shared storage repository, these files and images can be shared between multiple hypervisor hosts of the same type.

The PCM-AE integration with GPFS allows provisioning of the secure multi-tenant HPC clusters that are created with secure GPFS storage mounted on each server. After the storage is assigned to an account, only the users or groups of this account are given the permissions (read, write, and execution) to access the storage. To achieve this, PCM-AE communicates with the GPFS master node to add a user or group to the ACL (access control list) for the related directory in the GPFS file system.

When the cluster machines are provisioned with the GPFS file system that contains the storages, the user is not required to manually mount the file system. You also can use GPFS for PCM-AE as an extended storage, which looks like one folder in the operating system for virtual machines or physical machines. You can also use GPFS for PCM-AE as an image storage repository for virtual machines. For more information about GPFS administration, see the chapter “Secure storage using a GPFS cluster file system” in *Platform Cluster Manager Advanced Edition Version 4 Release 1: Administering*, SC27-4760-01.

Networking

The PCM-AE cluster uses the public, provisioning, monitoring, and application networks (if high speed and low latency interconnect is required). In a multi-rack cluster, we recommend the use of top-of-rack switches and setting up VLANs for different networks. When a server contains two network ports of the same speed, there is a way to tie them together by using Link Aggregation Control Protocol (LACP). For hypervisor hosts that are running and managing VMs, we recommend the use of 10 Gbps network adapters. Each rack has two top-of-rack switches with HA (that use inter-switch links at the top-of-rack level). This two-switch cluster with configured Virtual Link Aggregation Group (vLAG) feature allows multi-switch link aggregation, which provides higher performance and optimize parallel active-active forwarding. From each top-of-rack switch, aggregated uplinks can be configured to the up-level spine switches to build redundant two-tier and layer 3 fat tree networks.

In a secure multi-tenant fashion, PCM-AE should be configured to use VLAN secure networks and InfiniBand secure networks to create clusters on separate VLANs, on separate partitioned InfiniBand networks, or a combination. For more information about secure networks, see “VLAN secure networks” and “InfiniBand secure networks” chapters in *Platform Cluster Manager Advanced Edition Version 4 Release 1: Administering*, SC27-4760-01.

2.2 Workload management

IBM Platform Computing offers a range of workload management capabilities to optimize the running of various applications that use HPC clusters and ensure high resource usage with diverse workloads, business priorities, and application resource needs. Workload management uses computing resources efficiently to complete workloads as fast as possible. To enable an efficient workload allocation, an intelligent scheduling policy is required. An intelligent scheduling policy is based on understanding shared computing resources, the priority of the application, and user policies. Providing optimal service-level agreement (SLA) management and by providing greater versatility, visibility, and control of job scheduling helps reduce operational and infrastructure costs that are needed for maximum return of investment (ROI).

2.2.1 IBM Platform Load Sharing Facility

IBM Platform Load Sharing Facility (LSF) is a powerful workload management platform for demanding, distributed, and mission-critical HPC environments. IBM Platform LSF manages batch and highly parallel workloads. It provides flexible policy-driven scheduling features, which ensure that shared computing resources are automatically allocated to users, groups, and jobs in a fashion that is consistent with your service level agreements (SLAs), which improves resource usage and user productivity.

The advanced scheduling features make Platform LSF practical to operate at high usage, which translates to lower operating costs. Many features combine to reduce wait-times for users and deliver better service levels so that knowledge workers are more productive, which leads to faster, higher-quality results. Its robust administrative features make it more easily managed by a smaller set of administrators, which promotes efficiency and frees valuable staff time to work on other projects. For example, you can delegate control over a particular user community to a particular project or department manager. You also can reconfigure the cluster for one group without causing downtime for all other groups and use a new type of application that benefits from general-purpose GPUs. Having these features translates into flexibility.

Platform LSF functionality scales to meet your evolving requirements. In terms of scalability, Platform LSF is scalable in multiple dimensions. It scales to hundreds of thousands of nodes and millions of jobs. It also is scalable in other dimensions; for example, in the breadth of resources it supports. Whether you are managing Windows, Linux, GPU workloads, or floating application licenses, Platform LSF can provide flexible controls over vast numbers of users and resources across multiple data centers and geographies. It is also scalable to different workload types, whether you are managing single MPI parallel jobs that run for days across thousands of nodes, or millions of short-duration jobs that are measured in milliseconds. Platform LSF has scheduling features to meet these diverse needs and handle workloads at scale. Platform LSF is unique in its ability to solve a wide-range of scheduling problems, which enables multiple policies to be active on a cluster at the same time.

The smart scheduling policies of Platform LSF include the following features:

- Fairshare scheduling
- Topology and core-aware scheduling
- Backfill and preemption
- Resource reservations
- Resizable jobs
- Serial and parallel controls
- Advanced reservation
- Job starvation
- License scheduling
- SLA-based scheduling
- Absolute priority scheduling
- Checkpoint and resume
- Job arrays
- GPU-aware scheduling that is supported NVIDIA GPU and Intel Xeon Phi accelerators
- Tight integration with IBM Platform MPI and IBM Parallel Environment
- Plug-in schedulers

Platform LSF is available in the following editions to ensure that users have the right set of capabilities to meet their needs:

- Express Edition: Ideal for single-cluster environments and optimized for low throughput parallel jobs and simple user grouping structures.
- Standard Edition: Ideal for multi-cluster or grid environments and optimized for high throughput serial jobs and complex user grouping structures.
- Advanced Edition: Supports extreme scalability and throughput 100k+ cores and concurrent jobs.

The performance of Platform LSF depends upon many factors, including the number of nodes in the cluster, the number of concurrently running jobs, the number of pending jobs, the number of users querying the system, and the frequency of queries. As these tasks increase, the scheduling cycle and user response time increases. For high-throughput workloads, the overall system performance is dependent upon the processing power, I/O capacity, and memory of the scheduling node. Table 2-3 on page 31 provides sizing guidelines that are based on tested cluster configurations. For large clusters, it is recommended that users seek configuration assistance from IBM.

Table 2-3 Platform LSF scalability and throughput

Scalability and performance limits	Express	Standard	Advanced
Nodes	100	6,000	180,000
Cores	200	48,000	160,000
Concurrent short jobs	200	48,000	160,000
Pending jobs	10,000	500,000	2,000,000

The notion of Platform LSF heterogeneity is important because few organizations run only one operating system on only one hardware platform. Platform LSF scales from Windows to UNIX and Linux to Cray, NEC, and IBM supercomputers, which employ the world's most advanced architectures by offering customers complete freedom of choice to run the best platform for the best job with a fully supported software product.

Platform LSF is supported on any of the following operating environments and architectures:

- IBM AIX 6.x and 7.x on IBM Power 6 and POWER7
- HP UX B.11.31 on PA-RISC
- HP UX B.11.31 on IA64
- Solaris 10 and 11 on Sparc
- Solaris 10 and 11 on x86-64
- Linux on x86-64 Kernel 2.6 and 3.x
- Linux on IBM Power 6 and IBM POWER7 Kernel 2.6 and 3.x
- Windows 2003/2008/2012/XP/7/8 32-bit and 64-bit
- Apple Mac OS 10.x
- Cray XT3, XT4, XT5, XE6, and XC-30 on Linux Kernel 2.6
- glibc 2.3, SGI Performance Suite on Linux Kernel 2.5
- glibc 2.3 and ARMv7 Kernel 3.6 glibc 2.15 (Platform LSF slave host only)

For information about Platform LSF system support varies on Platform LSF Edition), see this website:

http://www-03.ibm.com/systems/technicalcomputing/platformcomputing/products/lsf/index.html

IBM Platform LSF provides optional add-ons that can be installed to extend the set of workload management capabilities. The following add-ons are designed to work together to address your high performance computing needs:

- IBM Platform Application Center (PAC): Portal management and application support that provides a rich environment for building easy-to-use, application-centric web interfaces, which simplify job submission, management, and remote 3D visualization.
- IBM Platform Process Manager (PPM): A powerful visual interface for designing complex engineering computational processes and multi-step workflows, and capturing repeatable best practices that can be used by other users.
- IBM Platform RTM: A flexible, real-time dashboard for monitoring global workloads and resources, including resource usage reporting. With better cluster visibility and cluster alerting tools, administrators can identify issues before the issues lead to outages, which helps avoid unnecessary service interruptions.
- IBM Platform Analytics: An advanced tool for visualizing and analyzing massive amounts of workload data for improved decision-making, more accurate capacity planning, optimizing asset usage and identifying and removing bottlenecks.

- IBM Platform License Scheduler: A license management tool that enables policy-driven allocation and tracking of commercial software licenses.
- IBM Platform Session Scheduler: A high-throughput and low-latency scheduling solution that is ideal for running short jobs, whether they are a list of tasks or job arrays with parametric execution.
- IBM Platform Dynamic Cluster: An innovative cloud management solution that transforms static, low-usage clusters into dynamic and shared cloud resources.

Use cases for Platform LSF

Platform LFS family products focus on the following technical computing markets:

- Electronics: Electronics design automation (EDA), electronic circuit design, and software development/QA.
- Manufacturing (automotive and aerospace and Defense): Computationally intensive simulations, crash and occupant safety, computational fluid dynamics, NVH, aerodynamics, durability, mechatronics design, engineering process and product data management, remote visualization, and materials engineering.
- Life Sciences: Human genome sequencing, QCD simulations, and therapeutic drug design.
- Energy/Oil & Gas: 3D visualization, reservoir simulation, seismic processing, and downstream chemical and mechanical engineering applications.
- Higher education and research: Electromagnetic simulations, finite element analysis, micro-scale optics, simulation, QCD simulations, visualization and image analysis, climate modeling, and weather forecast.
- Media and digital content creation: Animation, simulation, and rendering.

IBM Platform LSF is successfully deployed across many industries to manages batch and highly parallel workloads. Platform LSF use cases benefits from the key industry-leading ISV applications support. IBM Platform LSF within Platform Application Center comes complete with application templates for ANSYS Mechanical, ANSYS Fluent, ANSYS CFX, ClustalW, CMGL STARS, CMGL IMEX, CMGL GEM, HMMER, LS-DYNA, MATLAB, MSC Nastran, NCBI Blast, NWChem, Schlumberger ECLIPSE, Simulia Abaqus, STAR-CCM, and generic templates for in-house or open source applications. By standardizing access to applications, Platform Application Center makes it easier to enforce site policies and address security concerns within Role Based Access Control (RBAC).

Within Platform LSF, the computing resources are available to users through dynamic and transparent load sharing. Through its transparent remote job running, Platform LSF provides powerful remote hosts to improve application performance, which enables users to access resources from anywhere in the system.

Platform LSF architecture

Platform LSF is a layer of software services on top of heterogeneous enterprise resources. This layered service model is shown in Figure 2-7 and accepts and schedules workload for batch and non-batch applications, manages resources, and monitors all events.

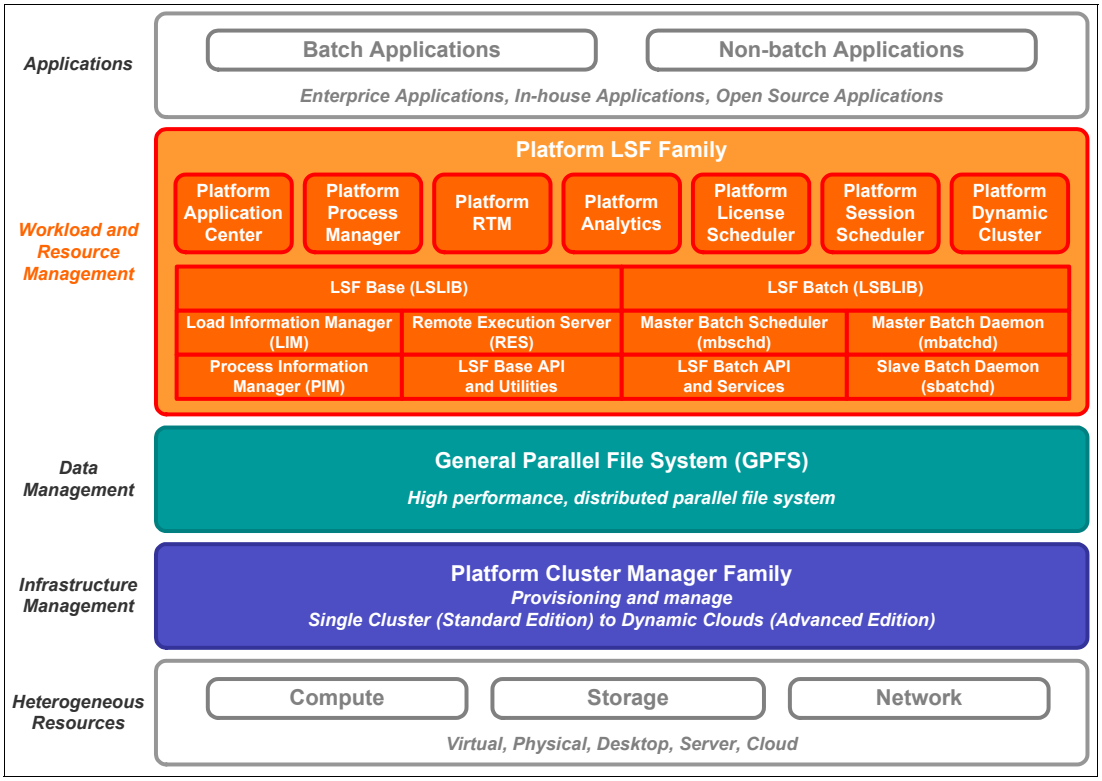

Figure 2-7 Platform LSF Layered Service Model

The three core components of the workload and resource management layer as shown in Figure 2-7 are LSF Base, LSF Batch, and LSF Libraries. Together, they help create a shared, scalable, and fault-tolerant infrastructure that delivers faster and more reliable workload performance.

LSF Base provides basic load-sharing services for the distributed system, such as resource usage information, host selection, job placement decisions, transparent remote running of jobs, and remote file option. These services are provided through the following components:

- Load Information Manager (LIM). The LIM on each host monitors its host's load and reports load information to LIM that is running on the master node. The master LIM collects information from all slave hosts that are running in the cluster and provides the same information to the applications.
- Process Information Manager (PIM). It is started by LIM and runs on each node in the cluster. It collects information about job processes that are running on the host, such as CPU and memory that is used by the job and reports the information to `sbatchd`.
- Remote Execution Server (RES). The RES on each server host accepts remote run requests and provides fast, transparent, and secure remote task running.

  There are a few utilities, such as **`lstools`**, **`lstcsh`**, and **`lsmake`** available to manage the workloads.

LSF Batch extends Platform LSF base services to provide a batch job processing system with load balancing and policy-driven resource allocation control. To provide this functionality, LSF Batch uses the following Platform LSF base services:

- Resource and load information from LIM to do load balancing
- Cluster configuration information from LIM
- The master LIM election service that is provided by LIM
- RES for interactive batch job running
- Remote file operation service that is provided by RES for file transfer

The core component of Platform LSF Batch is the scheduler framework that is based on the Master Batch Scheduler daemon (`mbschd`), which is combined with multiple plug-ins. All scheduling policies are implemented in the plug-in. For each cycle, the framework triggers scheduling, then control flow goes through each plug-in.

In different scheduling phases, the plug-in can intercept scheduling flow and influence the final decision. It means that to make scheduling decisions, Platform LSF uses multiple scheduling approaches that can run concurrently and be used in any combination, including user-defined custom scheduling approaches. This unique modular architecture makes the scheduler framework extendable to add new policy such as a new affinity plug-in.

LSF Batch services are provided by two daemons. The Master Batch daemon (`mbatchd`) runs on the master host and is responsible for the overall state of the job in the system. It receives job submission and information query requests. The daemon manages jobs that are held in queues and dispatches jobs to hosts as determined by `mbschd`. The Slave Batch daemon (`sbatchd`) runs on each slave host. The daemon receives requests to run the job from `mbatchd` and manages local running of the job. It is responsible for enforcing local policies and maintaining the state of the jobs on the hosts. The daemon creates a child `sbatchd` to handle every job run. The child `sbatchd` sends the job to the RES, which creates the environment on which the job runs.

LSF libraries provide APIs for distributed computing application developers to access job scheduling and resource management functions. The following Platform LSF libraries are available:

- LSLIB: The LSF base library that provides Platform LSF base services to applications across a heterogeneous network of computers. The Platform LSF base API is the direct user interface to the Platform LSF base system and provides easy access to the services of Platform LSF servers. A Platform LSF server host runs load-shared jobs. A LIM and a RES run on every Platform LSF server host. They interface with the host's operating system to give users a uniform, host-independent environment.
- LSBLIB: The LSF batch library gives application programmers access to the job queuing processing services that are provided by the Platform LSF batch servers. All Platform LSF batch user interface utilities are built on top of LSBLIB. The services that are available through LSBLIB include Platform LSF batch system information service, job manipulation service, log file processing service, and Platform LSF batch administration service.

Component model

The component model consists of multiple Platform LSF daemon processes that are running on each host in the distributed system, a comprehensive set of utilities that are built on top of the Platform LSF API, and relevant Platform LSF add-ons components that complement the required features. The type and number of running Platform LSF daemon processes depends on whether the host is a master node, one of the master node candidates, or a compute (slave) node, as shown in Figure 2-8.

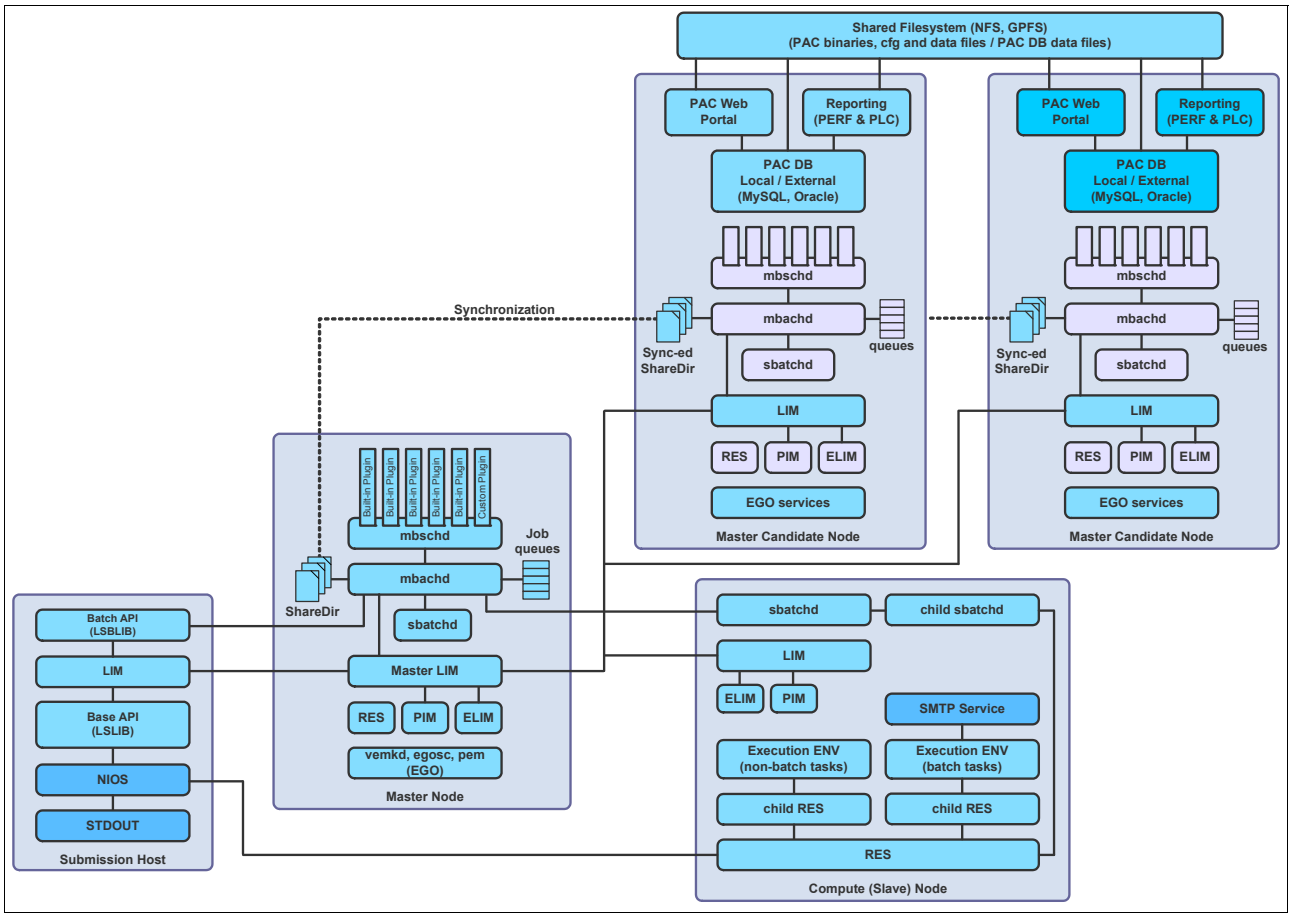

Figure 2-8 Platform LSF software components diagram

On each participating host in a Platform LSF cluster, an instance of LIM runs and collects host load and configuration information and forwards it to the master LIM that is running on the master host. The master LIM forwards load information to `mbatchd`, which forwards this information to `mbschd` to support scheduling decisions. If the master LIM becomes unavailable, a LIM on a master candidate automatically takes over. The External LIM (ELIM) is a site-definable executable file that collects and tracks custom dynamic load indexes (for example, information about GPUs). An ELIM can be a shell script or a compiled binary program that returns the values of the dynamic resources you define.

In addition to LIM, RES and PIM are other daemons that are running on each server host. RES accepts remote run requests to provide transparent and secure remote running of jobs and tasks. PIM collects CPU and memory usage information about job processes that are running on the host and reports the information to `sbatchd`.

Platform LSF can be accessed by users and administrators via the command-line interface (CLI), an API, or through the PAC Web Portal. The submission host that can be in a server host or a client host submits a job with commands by using the CLI or an application by using the API.

Platform LSF base execution (non-batch) tasks are user requests that are sent between the submission, master, and execution hosts. From the submission host, `lsrun` submits a task into the Platform LSF base. The submitted task proceeds through the Platform LSF base API (LSLIB). The LIM communicates the task's information to the cluster's master LIM. Periodically, the LIM on individual machines gathers its 12 built-in load indexes and forwards this information to the master LIM. The master LIM determines the best host to run the task and sends this information back to the submission host's LIM.

Information about the chosen execution host is passed through the Platform LSF base API back to `lsrun`, which creates network input/output server (NIOS), which is the communication pipe that talks to the RES on the execution host. Task execution information is passed from the NIOS to the RES on the execution host. The RES creates a child RES and passes the task execution information to the child RES. The child RES creates the execution environment and runs the task. The child RES receives completed task information and sends it to the RES. The output is sent from the RES to the NIOS. The child RES and the execution environment is destroyed by the RES. The NIOS sends the output to standard out (STDOUT).

In cases of Platform LSF batch execution of the (batch) tasks, the submission host does not directly interact with the execution host. From the submission host, `bsub` or `lsb_submit()` submits a job to the Platform LSF batch. The submitted job proceeds through the Platform LSF batch API (LSBLIB). The LIM communicates the job's information to the cluster's master LIM. Based on gathered load indexes, the master LIM determines the best host to run the job and sends this information back to the submission host's LIM. Information about the chosen execution host is passed through the Platform LSF batch API back to `bsub` or `lsb_submit()`.

To enter the batch system, `bsub` or `lsb_submit()` sends the job by using LSBLIB services to the `mbatchd` that is running on the cluster's master host. The `mbatchd` puts the job in an appropriate queue and waits for the appropriate time to dispatch the job. User jobs are held in batch queues by `mbatchd`, which checks the load information about all candidate hosts periodically. Then, `mbatchd` dispatches the job when an execution host with the necessary resources becomes available where it is received by the host's `sbatchd`. When more than one host is available, the best host is chosen.

After a job is sent to a `sbatchd`, that `sbatchd` controls the execution of the job and reports the job's status to `mbatchd`. The `sbatchd` creates a child `sbatchd` to handle job execution. The child `sbatchd` sends the job to the RES. The RES creates the execution environment to run the job. The job is run and results of the job are sent to the user through email system (SMTP service).

In case of the job submission through the web interface, the Platform Application Center (PAC) manages the Platform LSF Library calls. PAC components include web portal, reporting services, and database (MySQL or Oracle). To support PAC failover, the configuration files and binaries are stored on the shared file system (NFS or GPFS), and failover services are provided by EGO. Two Platform LSF master candidate hosts are used for failover (for best performance, do not use the Platform LSF master host as the Platform Application Center host). When the primary candidate host on which PAC is running fail, EGO can start PAC services and database instance on the backup candidate host.

When Platform LSF is installed without EGO enabled, resource allocation is done by Platform LSF in its core. Part of the EGO functionality is embedded on Platform LSF, which enables the application to perform the parts of the job for which EGO is responsible. When EGO is enabled (required for PAC failover), it adds more fine-grained resource allocation capabilities, high availability services to `sbatch` and RES, and faster cluster startup. If the cluster has PAC and PERF controlled by EGO, these services are run as EGO services (each service uses one slot on a management host).

Platform LSF fault tolerance depends on the event log file, `lsb.events`, which is kept on the primary file server in the shared directory LSB_SHAREDIR, which should be configured to maintain copies of these logs to use as a backup. If the host that contains the primary copy of the logs fails, Platform LSF continues to operate by using the synchronized duplicate logs. When the host recovers, Platform LSF uses the duplicate logs to update the primary copies. LSB_SHAREDIR is used for temporary work files, log files, transaction files, and spooling must be accessible from all potential Platform LSF master hosts.

The component model can be extended with installed Platform LSF add-on components for Platform License Scheduler, Platform RTM, and Platform Analytics. Other add-on components for Platform Dynamic Cluster, Platform Process Manager, and Platform Session Manager with MultiCluster configuration can be used to support extreme scalability and throughput for multi-site or geographic cluster environments.

The component model also can be extended with integrated components to support running MPI Jobs. Platform LSF supports Open MPI, Platform MPI, MVAPICH, Intel MPI, and mpich2.

Operational model

The operational model of the Platform LSF environment varies depending on the functional and non-functional requirements. One of the differences can be the number of the supported users that affects scalability and manageability. Another difference can be the requirement for multi-site deployment that is based on a customer's datacenter locations.

Also, the number of the supported applications and the type of running applications are significant when the appropriate technology platform is selected and it is decided whether to use cluster management to provision and manage or dynamically change the Platform LSF cluster environment.

The sample of the high available Platform LSF environment that is shown in Figure 2-9 shows one of the several possible configurations for MSC and ANSYS Application Ready Solution.

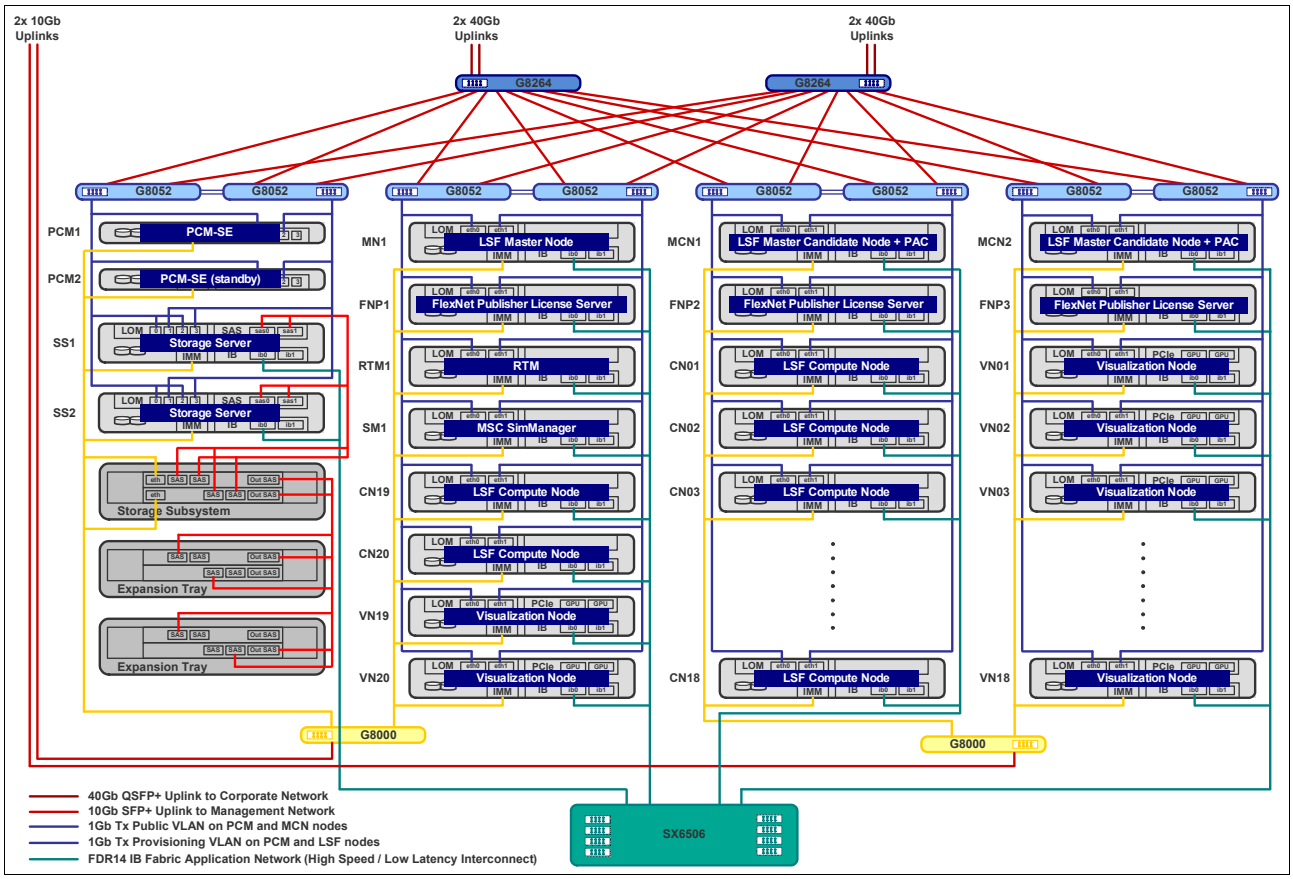

Figure 2-9 Platform LSF cluster deployment on the physical hardware

If the cluster is smaller and if cost-effectiveness is important, the cluster manager for easy provisioning should be omitted. Also, if applications other OS than Linux are required to run, the pHPC and PCM-SE cannot be used. Otherwise, for clusters that are running Linux supported applications, the cluster management solution is recommended.

Cluster management nodes

The sample solution includes one active PCM-SE management node (PCM1) and one standby node (PCM2). When a failover process occurs, the standby management node takes over as the management node with all running services. The management nodes connect to a public and provisioning network. Because the architecture is optimized for a specific application and most software deployments are fully automated, a cluster can be deployed in a short time. System administrators should use the PCM-SE web-based interface as the management console for performing daily cluster management and monitoring. This eliminates the need for a full-time system administrator with extensive technical or HPC expertise for managing the cluster. Device drivers, such as OFED for InfiniBand, VNC server, and DCV for 3D visualization, also are deployed as part of the cluster deployment. PCM-SE supports kits, which provide a framework that allows third-party packages to be configured with the system.

Shared storage

To support I/O-intensive applications, we recommend building a two-node GPFS cluster with tiebreaker disks over both shared storage nodes (SS1 and SS2). The shared storage nodes connect to public, provisioning, and application networks. GPFS provides to the LSF users secure storage to maintain user profiles, and to share compute model data files, which allows applications that are running on the cluster within high speed and low latency interconnect access any required data, which significantly improves application performance.

To create a high available PCM-SE environment, shared storage is required to share user home directories and system working directories. All shared file systems must be accessible by the provisioning network for PCM-SE management nodes and all compute nodes. The Platform LSF working directory must be available through a shared file system in the master and master candidate hosts. PAC DB data files are stored in the shared drive so the database on the secondary master candidate node uses the same data on the shared directory when it is started. The RTM DB data files are also stored in the shared drive.

LSF master node

To achieve the highest degree of performance and scalability, we strongly recommend that you use a powerful master host. There is no minimum CPU requirement (we recommend the use of multi-core CPUs) and any host with sufficient physical memory can run LSF as master host or master candidate host. Active jobs use most of the memory that LSF requires (we recommend the use of 8 GB RAM per core). The LSF master node (MN1) connects to public and provisioning networks.

LSF master candidate nodes

Two LSF master candidate nodes (MCN1 and MCN2) are used for failover of the Platform Application Center (PAC), which provides a graphic user interface to Platform LSF. Ideally, these nodes should be placed on separate racks for resiliency. Both LSF master candidate nodes connect to public and provisioning networks. PAC is the primary interface to the ANSYS and MSC applications through the application templates. For ANSYS applications, the AR-Fluent and AR-Mechanical application templates are provided. For MSC applications, the AR-NASTRAN and AppDCVonLinux (to Patran through DCV) application templates are provided.

RTM node

Platform RTM server (RTM1) is an operational dashboard for IBM Platform LSF environments that provides comprehensive workload monitoring and reporting. RTM displays resource-related information, such as the number of jobs that are submitted, the details of individual jobs (load average, CPU usage, and job owner), or the hosts on which the jobs ran. RTM also is used to monitor FlexNet Publisher License Servers and GPFS. By using GPFS ELIM, we can monitor GPFS on a per LSF host in the RTM GUI and a per LSF cluster basis as a whole or per volume level. The RTM node connects to the provisioning network.

FlexNet Publisher License nodes

The Platform License Scheduler works with master FlexNet Publisher License Server to control and monitor ANSYS and MSC license usage. To support high availability of the FlexNet Publisher License Server, the three-server redundancy configuration is used (FNP1, FNP2, and FNP3). If the master fails, the secondary license server becomes the master and serves licenses to FLEX enabled applications. The tertiary license server can never be the master. If the primary and secondary license servers go down, licenses are no longer served to FLEX enabled applications. The master does not serve licenses unless there are at least two license servers in the triad that are running and communicating. The FlexNet Publisher License Servers connects to the provisioning network.

MSC SimManager

MSC SimManager (SM1) with its `mscdb` database instance is installed on a separate node. We recommend setting the same values of the MSC variables into Platform Application Center (PAC) templates for MSC. If this information is embedded in an MSC template in PAC, it is important that when changes are made to the installation of the MSC application suite, these changes are propagated to the MSC application templates in PAC. PAC, which is complementary to SimManager, provides a robust interface to such key simulation components as MSC Nastran and MSC Patran. MSC SimManager can be used to submit jobs to the back-end LSF cluster in addition to providing project and data management. The MSC SimManager (SN1) connects to public and provisioning networks.

LSF compute node

The LSF compute nodes (CN01 to CN20) are designed for computationally intensive applications that are developed by ANSYS Mechanical, ANSYS Fluent, and MSC Nastran. They are required to satisfy the computational requirements for solving use cases. They have some local disk space for the OS and temporary storage that is used by the running applications. The compute nodes also mount the shared GPFS storage that is needed for cooperation in solving a single problem. This is facilitated by the connectivity to a high-speed, low-latency InfiniBand interconnect. Also, ANSYS parallel computing by using MPI merits the inclusion of an InfiniBand interconnect. Large models in ANSYS might not fit in the available memory and must be solved out-of-core. In that case, these nodes can benefit from robust local storage that is attached to each compute server.

In addition to the operating system, the application and runtime libraries are installed on all LSF compute nodes. The monitoring and resource management agents are connected to the cluster management software and the workload management software is installed.

The performance of the LSF compute node (depending on the configuration of ANSYS Fluent) does not require a large amount of memory per node when multiple nodes are used to run large problems. We recommend the use of 8 GB RAM per core node with two eight-core Intel Xeon E5-2600 v2 CPUs. ANSYS Fluent does not perform any I/O locally at each node in the cluster. Therefore, a robust shared file system satisfies the I/O requirements of most ANSYS Fluent applications. ANSYS Mechanical requires a large amount of memory so that the simulations can run within memory most of the time. We recommend the use of 16 GB RAM per core node with two eight-core Intel Xeon E5-2600 v2 CPUs. MSC Nastran also requires a large amount of memory so that the simulations can run within memory most of the time. We recommend the use of 16 GB RAM per core node with two eight-core Intel Xeon E5-2600 v2 CPUs. The LSF compute nodes connects to provisioning and application networks.

Visualization node

A visualization node is required to support the remote 3D visualization. A pool of nodes (VN01-VN20) is designed as visualization nodes, which are excluded from the computational work queues. These nodes are equipped with GPUs, which are officially supported by ANSYS Workbench and MSC Patran. Each visualization node can be equipped with up to four GPUs. A single visualization node can support several simultaneous interactive sessions. Platform LSF keeps track of the allocation of these interactive sessions. When a user requests an interactive login session through the Platform Application Center web portal, a free session is allocated to the user. Through remote visualization, the rendering of a graphics application is done on the graphics adapter. The rendered graphics are compressed and transferred to the thin client. The remote visualization software engine that is supported on the visualization node is Desktop Cloud Visualization (DCV), which is produced by the NICE Software.

When an interactive session that uses DCV is requested and allocated, PAC downloads session-related information of the DCV to the client workstation that is requesting the session. The client component of the visualization software, which is on the client workstation, uses this session information and connects with the visualization node on which Platform LSF allocated the session. The client part of the visualization prompts the user for credentials for authentication. If the login is successful, the interactive session is established to start graphics-oriented applications, such as ANSYS Workbench or MSC PATRAN.

In addition to the operating system, the application and runtime libraries are installed on all visualization nodes. The monitoring and resource management agents are connected to the cluster management software and the workload management software is installed.

The performance of the visualization node depends on the configuration because the pre- and post-processing of the applications require large amounts of memory. We recommend the use of 256 GB RAM and one eight-core Intel Xeon E5-2600 v2 product family CPU for two NVIDIA GPU adapters or 512 GB RAM and two eight-core Intel Xeon E5-2600 v2 product family CPUs for four NVIDIA GPU adapters per visualization node. The visualization nodes connect to provisioning and application networks.

Networking

The Platform LSF cluster uses the same networks as the PCM-SE cluster (public, provisioning, monitoring, and application networks). In a multi-rack cluster, we recommend the use of top-of-rack switches and set up VLANs for different networks. When a server contains two network ports of the same speed, there is a way to tie them together by using LACP. Each rack has two top-of-rack switches with HA (that uses inter-switch links at the top-of-rack level). This two-switch cluster with configured Virtual Link Aggregation Group (vLAG) feature allows multi-switch link aggregation, which provides higher performance and optimizes parallel active-active forwarding. From each top-of-rack switch, aggregated uplinks can be configured to the up-level spine switches to build redundant Two-Tier and Layer 3 Fat Tree network. The Equal-Cost Multi-Path Routing (ECMP) L3 implementation is used for scalability if Virtual Router Redundancy Protocol (VRRP) is not applicable.

2.2.2 IBM Platform Symphony

IBM Platform Symphony is the most powerful enterprise-class management for running distributed applications and Big Data analytics on a scalable and shared grid. Platform Symphony's efficient, low-latency middleware and scheduling architecture is designed to provide the performance and agility that is required to predictably meet and exceed throughput goals for running diverse workloads. It accelerates various compute-intensive and data-intensive applications for faster results and better use of all available resources.

Platform Symphony features the following characteristics:

- Low latency and high throughput: Platform Symphony provides submillisecond responses. The sending throughput per task was bench marked repeatedly over 17,000 tasks per second per application.
- Large scale: An individual session manager (per application) can schedule a task up to 10k cores. Platform Symphony manages up to 40k cores per grid and allows multiple grids to be linked. Platform Symphony Advanced Edition with multi-cluster feature can manage up to 100k cores.
- Performance enhancements: Low Latency 'Push' Infrastructure reduces the wait time for task allocation. Service-oriented architecture Framework that is written in C++ reduces the wait time for jobs to start running, a sophisticated scheduling engine that reduces the wait time for pending jobs/tasks, and data management reduces the wait time for getting data to the jobs or tasks.

High availability and resiliency reduce the wait time on recovery of jobs and tasks, Sharing resources across application boundaries reduces the server wait time for new work when there are pending tasks, shared memory logic for MapReduce reduces data movement, single service instance for multiple tasks (MTS), and parallel EGO service starts (30,000 services in under two minutes).

- Dynamic resource management: Slot allocation changes dynamically based on job priority and server thresholds. Lending and borrowing resources from one application to another within advanced resource sharing ensures SLAs while encouraging resource sharing.
- Application lifecycle: Support for rolling upgrades of the Platform Symphony software. Support for multiple versions of Hadoop co-existing on the same cluster.
- Reliability: Platform Symphony makes all MapReduce and HDFS-related services highly available (name nodes, job trackers, task trackers, and so on).
- Sophisticated scheduling engine: Platform Symphony has a fair share of scheduling with 10,000 levels of prioritization. Also, preemptive and resource threshold-based scheduling with runtime change management.
- Open: Platform Symphony supports multiple APIs and languages. Fully compatible with Java, Pig, Hive, and other MR applications. Platform Symphony also supports multiple data sources, including HDFS and GPFS.
- Management tools: Platform Symphony provides a comprehensive management capability for troubleshooting, alerting, and tracking jobs and rich reporting capabilities.

Platform Symphony is available in the following editions, which gives users the best set of capabilities to meet their needs:

- IBM Platform Symphony Developer Edition: Builds and tests applications without the need for a full-scale grid (available for download at no cost).
- IBM Platform Symphony Express Edition: For departmental clusters where this is an ideal, cost-effective solution.
- IBM Platform Symphony Standard Edition: This version is for enterprise class performance and scalability.
- IBM Platform Symphony Advanced Edition: This is the best choice for distributed compute and data intensive applications, including optimized and low-latency MapReduce implementations.

Platform Symphony clients and services can be implemented on different operating environments, languages, and frameworks. Clusters also can consist of nodes that are running multiple operating systems. For example, 32- and 64-bit Linux hosts can be mixed that are running different Linux distributions and multiple Microsoft Windows operating systems can be deployed. Platform Symphony can manage all of these different types of hosts in the same cluster and control what application services run on each host.

For more information about system support (which varies on Platform Symphony Edition), see *IBM Platform Symphony 6.1.1: Supported System Configurations*, SC27-5373-01, which is available at this website:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.platform_product_libraries.doc/platform_product.htm

Table 2-4 on page 43 summarizes the features that are associated with each Platform Symphony edition and provides sizing guidelines that are based on tested cluster configurations. For advanced clusters, it is recommended that customers seek configuration assistance from IBM.

Table 2-4 Platform Symphony features and scalability

Features	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	X	-	-	X
Multi-cluster management	-	-	-	X
Max hosts/cores	2 Hosts	240 Cores	5k Hosts, 40k Cores	5k Hosts, 40k Cores
Application managers	-	5	300	300

The following add-on resource harvesting tools can be used with Platform Symphony Standard and Advanced Editions:

- IBM Platform Symphony Desktop Harvesting: This add-on harnesses the resources from available idle desktops and adds them to the pool of potential candidates to help complete tasks. Platform Symphony services do not interfere with other applications that are running on the desktops and harvested resources are managed directly through the integrated management interface.
- IBM Platform Symphony Server/VM Harvesting: To use more of your enterprise's resources, this addition allows you to tap idle or under-used servers and virtual machines (VM). Instead of requiring new infrastructure investment, Platform Symphony locates and aggregates these server resources as part of the grid whenever more capacity is needed to handle larger workloads or when the speed of results is critical.
- IBM Platform Symphony GPU Harvesting: To unleash the power of general-purpose graphic processing units (GPUs), this tool enables applications to share expensive GPU resources more effectively and to scale beyond the confines a single GPU. Sharing GPUs more efficiently among multiple applications and detecting and addressing GPU-specific issues at run time helps improve service levels and reduce capital spending.
- IBM Platform Analytics: An advanced analysis and visualization tool for analyzing massive amounts of workload and infrastructure usage data that is collected from IBM Platform Symphony clusters. It enables you to easily correlate job, resources, and license data from multiple Platform Symphony clusters for data driven decision making.

The following complementary products can be used with Platform Symphony:

- IBM InfoSphere BigInsights (IBM Distribution of Apache Hadoop)
- IBM Algorithmics® (Full-featured Enterprise Risk Management Solution)
- IBM General Parallel File Systems (High-performance enterprise file management platform)
- IBM Platform Process Manager (Design and automation of complex processes)
- IBM Intelligent Cluster™ (Pre-configured, pre-integrated, optimized, and fully supported HPC solution)

Use cases for Platform Symphony

Platform Symphony is successfully deployed across many industries to manages multiple compute and data intensive workloads. Platform Symphony fits all time critical use cases where service-oriented applications are running and services are calling programmatically by using APIs. Whereas a batch scheduler can schedule jobs in seconds or minutes, Platform Symphony can schedule tasks in milliseconds. Because of this difference, Platform Symphony can be described as supporting online or near real-time requirements. Well-documented APIs enable fast integrations for applications that are written in C, C++, C#, .NET, Visual Basic, Java, Excel COM, R, and various popular scripting languages. Platform Symphony provides flexible distributed run time to support various Big Data and analytics applications that benefit from the best-in class Hadoop MapReduce implementation (Enhanced Platform Symphony MapReduce framework).

Platform Symphony targets the following markets:

- Financial Services: Market risk (VaR) calculations, credit risk including counterparty risk (CCR), Credit Value Adjustments (CVA), equity derivatives trading, stochastic volatility modeling, actuarial analysis and modeling, ETL process acceleration, fraud detection, and mining of unstructured data.
- Manufacturing: Data warehouse optimization, predictive asset optimization, process simulation, Finite Elements Analysis, and failure analysis.
- Health and Life Sciences: Health monitoring and intervention, Big Data biology emerging as a MapReduce workload, genome sequencing and analysis, drug discovery, protein folding, and medical imaging.
- Government and Intelligence: Weather analysis, collaborative research, MapReduce logic implementation, data analysis in native formats, enhanced intelligence and surveillance insight, real-time cyber attack prediction and mitigation, and crime prediction and protection.
- Energy, Oil, and Gas: Distribution load forecasting and scheduling, enable customer energy management, smart meter analytics, advanced condition monitoring, drilling surveillance and optimization, and production surveillance and optimization.
- Media and Entertainment: Optimized promotions effectiveness, real-time demand forecast, micro-market campaign management, and digital rendering.
- E-Gaming: Game-and Player-related analytics.
- Telco: Network analytics, location-based services, pro-active call center, and smarter campaigns.
- Retail: Customer behavior and trend analysis driving large and complex analytics, merchandise optimization, and actionable customer insight.

Platform Symphony architecture

Platform Symphony is a layer of software services on top of the heterogeneous enterprise resources that provide workload and resource management. This layered service model (as shown in Figure 2-10 on page 45) presents a simplified Platform Symphony architecture.

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 workloads. The Enterprise Grid Orchestrator (EGO), as a resource manager, manages the supply and distribution of resources, which makes them available to applications. EGO provides resource provisioning, remote execution, high availability, business continuity, and cluster management tools.

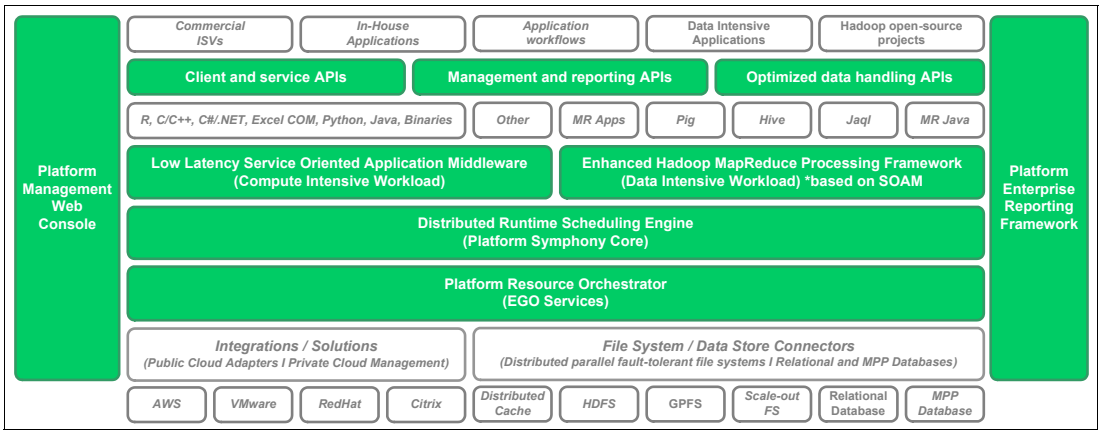

Figure 2-10 Platform Symphony Layered Service Model

A workload manager interfaces directly with the application, receiving work, processing it, and returning the results. A workload manager provides a set of APIs or can interface with more runtime components to enable the application components to communicate and perform work. Platform Symphony works with the Service-Oriented Application (SOA) model. In a SOA environment, workload is expressed in terms of messages, sessions, and services. SOAM (SOA Middleware) works as a workload manager within the Platform Symphony.

When a client submits an application request, the request is received by SOAM. SOAM manages the scheduling of the workload to its assigned resources, requesting more resources as required to meet service-level agreements. SOAM transfers input from the client to the service, returns results to the client, and then releases excess resources to the resource manager.

Within Platform Symphony, the Enhanced Hadoop MapReduce Processing Framework supports data-intensive workload management by using a special implementation on SOAM for MapReduce workload. The MapReduce Processing Framework is available only with the Advanced Edition. Significant performance improvement for the Symphony's MapReduce framework is attained for most of the MapReduce jobs when compared with the open source Hadoop framework, and especially for the short-run jobs. This is based mainly on the low latency and the immediate map allocation and job startup design features of the SOAM (JobTracker and TaskTracker components of Hadoop are replaced with Symphony SOAM components, which are much faster at allocating resources to MapReduce jobs).

For more information, see the following publications:

- *Platform Symphony Version 6 Release 1.1: Platform Symphony Foundations*, SC27-5065-02
- *Platform Symphony Version 6 Release 1.1: User Guide for the MapReduce Framework*, GC27-5072-02

These publications are available at this website:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.platform_product_libraries.doc/platform_product.htm

Component model

The component model that is shown in Figure 2-11consists of multiple Platform Symphony processes that are running on each host in a distributed system, a comprehensive set of utilities that are built on top of the Platform Symphony API, and relevant Platform Symphony add-on components complement the required features. The type and number of running Platform Symphony processes depend on whether the host is a master node, master node candidate, one of the management nodes, or a compute node. Other considered nodes are Platform Symphony clients and nodes (relational database servers and GPFS servers) that provide high available services that are related to the production environment.

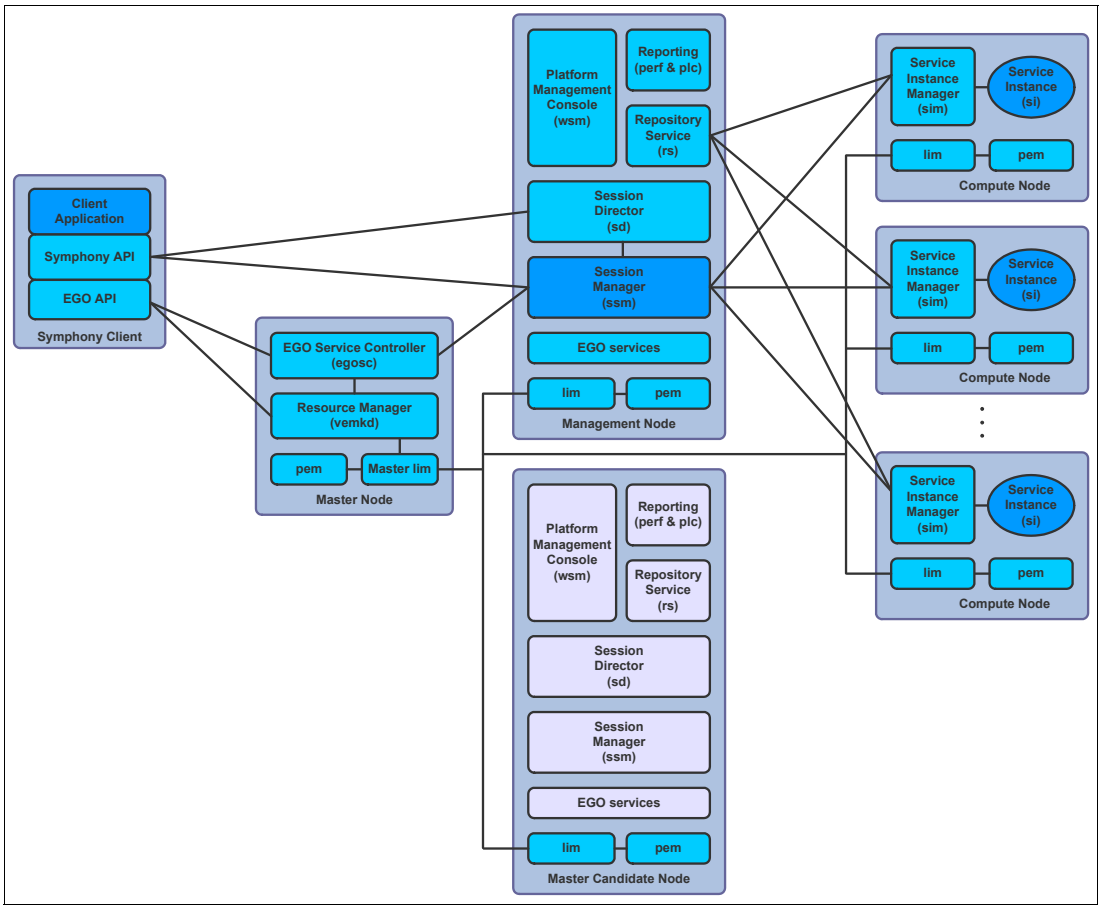

Figure 2-11 Platform Symphony software components diagram

Management nodes are designated to run the management components of the grid. By default, these nodes do not run the workload for users. The master node is the first node that is installed and the resource manager of the grid is here. There is only one master node at a time. The master candidate node act as the master if the master fails and usually is configured as one of the possible management nodes. The management node (or management nodes) run session managers (there is one session manager per available slot on a management host and one session manager per application) and provide an interface to the clients. Compute nodes are designated to run work and provide computing resources to users. Client nodes (Platform Symphony client) are used for submitting work to the grid and normally they are not members of the grid.

On the master node, the master lim starts vemkd and process execution monitor (pem). There is one master lim per grid. The vemkd (EGO kernel) starts the service controller egosc, maintains security policies (allowing only authorized access), and maintains resource allocation policies (distributing resources accordingly).

There is one vemkd per cluster and it runs on the master node. The pem monitors vemkd, and notifies the master lim if vemkd fails. The EGO service controller (egosc) is the first service that runs on top of the EGO kernel. It functions as a bootstrap mechanism for starting the other services in the cluster. It also monitors and recovers the other services. After the kernel boots, it reads a configuration file to retrieve the list of services to be started. There is one egosc per cluster, and it runs on the master node.

On other management nodes, the session director (sd) acts as a liaison between the client application and the session manager (ssm). There is one session director process per cluster, and it can run on the master or other management node. The repository service (rs) provides a deployment mechanism for service packages to the compute nodes in the cluster. There is one repository service per cluster, and it can run on the master or other management node.

The load information manager (lim) monitors the load on the node, and starts pem. The pem starts Platform Symphony processes on the node. The ssm is the primary workload scheduler for an application. There is one session manager per application. The web service manager (wsm) runs the Platform Management Console. The PERF loader controller (plc) loads data into the reporting database. The PERF data purger (purger) purges reporting database records. If the cluster has wsm and plc controlled by EGO (required for high availability), the relevant services are run as EGO services.

On the master candidate nodes, the lim monitors the load on the master candidate node and starts pem. The lim also monitors the status of the master lim. If the master node fails, lim also elects a new master node. The pem starts relevant Platform Symphony processes on the node. At a minimum, three management nodes are needed to provide high available management services. If the management node fails, the master candidate node (as one of the management nodes) must be configured to start all relevant EGO services.

On the compute nodes, the lim starts pem on the node, monitors the load on, and passes the configuration information and load information to the master lim on the master node. The pem monitors the lim process. The service instance manager (sim) is started on the compute node when the workload is submitted to the node if the application is preconfigured. The sim then starts service instance (si). There is one sim per service instance.

A Platform Symphony client connects to Platform Symphony's session director (sd) and ssm servers and to EGO's kernel daemon (vemkd) and the service controller (egosc). This is because a Platform Symphony client indirectly uses the EGO API to communicate with EGO. More specifically, a Platform Symphony client is linked with the Platform Symphony SDK library that uses the Platform Symphony API (for sessions). The Platform Symphony API internally uses the EGO API to communicate with EGO, so the Platform Symphony SDK client internally is also an EGO client.

The Service-Oriented Architecture Middleware (SOAM) is responsible for the role of workload manager and manages service-oriented application workloads within the cluster, which creates a demand for cluster resources. The SOAM components consist of the sd, ssm, sim, and si.

To support failover for multi-node clusters, a shared file system is required to maintain the configuration files, binaries, deployment packages, and logs. To enable this shared file system, you must create a shared directory that is fully controlled by the cluster administrator and is accessible from all management nodes.

To eliminate single point of failure (SPOF), the file system should be high-available by using HA-NFS or IBM GPFS, for example. For the best performance, parallel multi-node access for shared data and fast inter-node replication, we recommend the use of GPFS.

For the compute nodes, we recommend the use of the GPFS File Placement Optimizer (FPO) feature that is designed for Big Data applications that process massive amounts of data. GPFS implements the POSIX specification natively, which means that multiple applications (MapReduce and non-MapReduce applications) can share the same file system, which improves flexibility. The use of GPFS eliminates the HDFS node name as a SPOF, which improves file system reliability and recoverability. Within GPFS, you can employ the right storage architecture, depending on the application need by using GPFS FPO with n-way block replication for Hadoop workloads and traditional GPFS for non-Hadoop workloads to improve flexibility and minimize costs.

The Platform Symphony Multi-Cluster (SMC) feature can be used to extend scalability of the grid to distribute workload execution among clusters, and to repurpose hosts between clusters to satisfy peak load. This feature is available within Platform Symphony Advanced Edition and allows spanning up to 20 silo clusters that are scaling up to the 100,000 hosts with 800,000 cores and up to 1,000 application managers in total.

The MultiCluster management system (SMC-Master) that is shown in Figure 2-12 is composed of software components that run outside of individual clusters to manage movement of resources between clusters and coordinate and track them. The SMC Master cluster is on a set of hosts to ensure failover and uses EGO technology for high availability and to manage services within the MultiCluster. In each cluster that you want to manage, you enable the MultiCluster proxy (SMC-Proxy) on a management host. The MultiCluster proxy is the SMCP service that discovers hosts in the cluster and triggers actions to repurpose them.

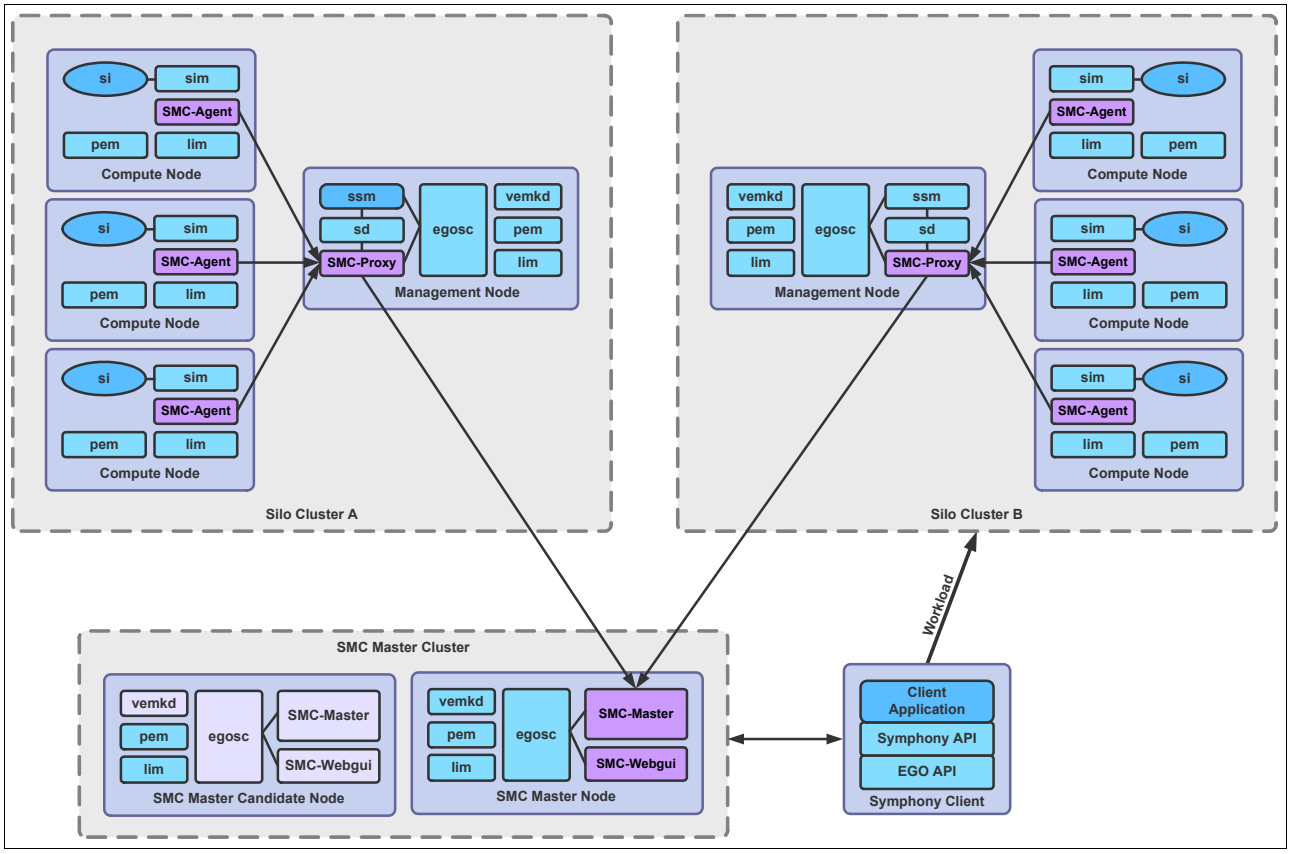

Figure 2-12 Platform Symphony Multi-Cluster software components diagram

There is one MultiCluster proxy per silo cluster. The MultiCluster agent (SMC-Agent) is a daemon that runs on demand on a host that is repurposed. It is responsible for stopping and starting the `lim` and starting move-in and move-out scripts and reporting their results to the MultiCluster proxy. By using MultiCluster, you can monitor all clusters from one central console (SMC-Webgui).

For more information, see *Platform Symphony Version 6 Release 1.1: MultiCluster User Guide*, SC27-5083-02, which is available at this website:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.platform_product_libraries.doc/platform_product.htm

Operational model

The sample of a Platform Symphony cluster as a high available environment is shown in Figure 2-13. The number of supported applications and type of running applications (compute-intensive versus data-intensive) are significant to select the appropriate technology platform and required scalability, and determine whether to use the cluster management to provision the cluster. If the cluster is smaller and if cost-effectiveness matters, the cluster manager for easy provisioning should be omitted.

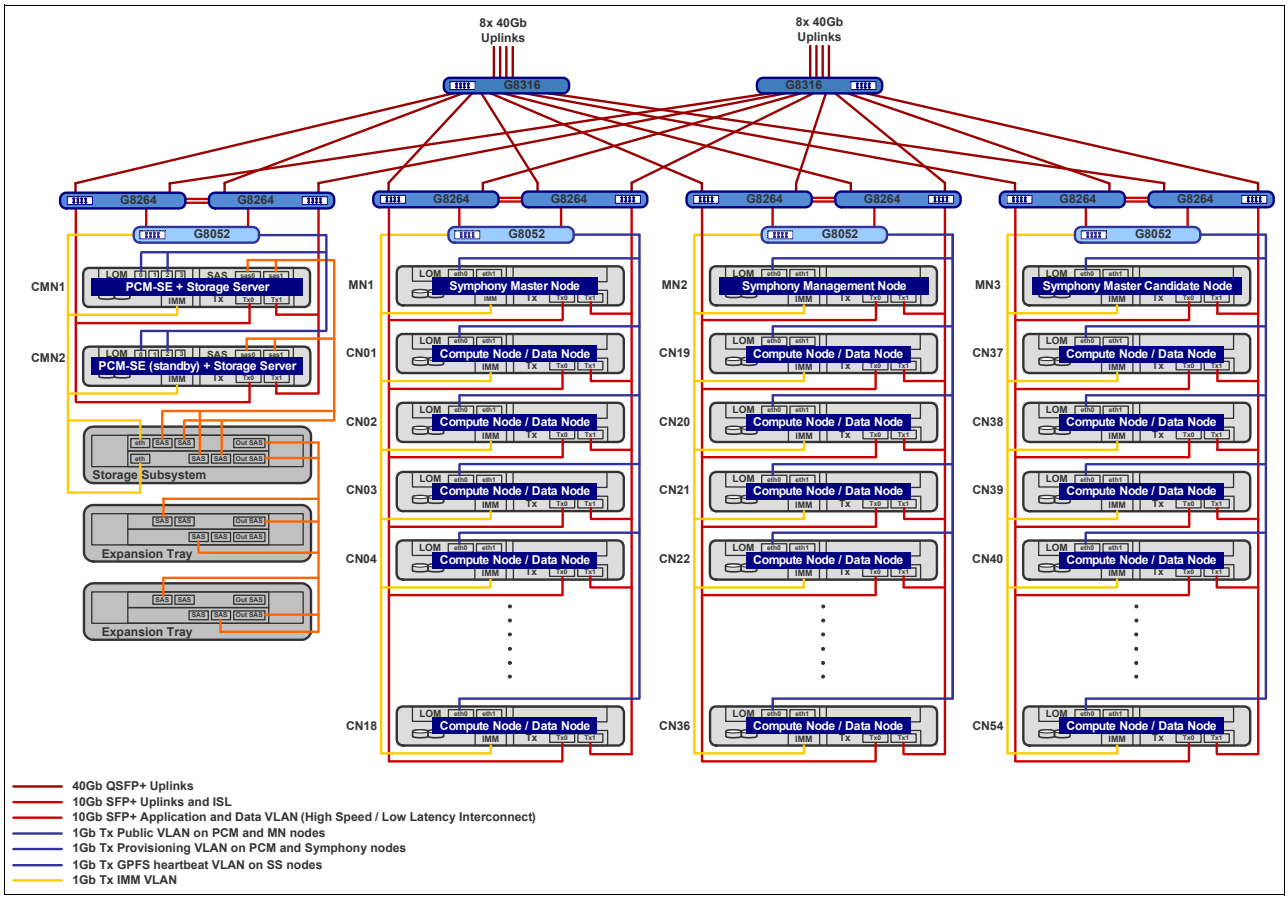

Figure 2-13 Platform Symphony cluster deployment on the physical hardware

Also, if applications are required to run on an operating system other than Linux, PCM-SE cannot be used. Otherwise, for clusters that are running Linux supported applications, the cluster management solution is recommended. In our case (which is one of several possible configurations), we use PCM-SE for provisioning, including one shared cluster storage that is supplemented with two-node GPFS cluster.

Cluster management nodes

The sample solution includes one active PCM-SE management node (CMN1) and one standby node (CMN2). When a failover process occurs, the standby management node takes over as the management node with all running services. Because the software deployments are fully automated, the Platform Symphony cluster can be deployed in a short time. System administrators should use the PCM-SE web-based interface as the management console for performing daily management and monitoring of the cluster. PCM-SE supports kits, which provide a framework that allows third-party packages to be configured with the system. The management nodes connect to the public and the provisioning networks.

Shared storage for cluster management

To create a high available PCM-SE environment, shared storage is required to share home and system working directories. All shared file systems must be accessible by the provisioning network and all provisioned Platform Symphony nodes. This requirement can be covered by a two-node GPFS cluster with tiebreaker disks on the storage subsystem over both storage (NSD server) nodes (CMN1 and CMN2).

Shared storage for Platform Symphony

To support Platform Symphony management failover, the `ego.shared` directory (which contains configuration files, binaries, deployment packages, and logs) must be accessible to all Platform Symphony management nodes. This requirement is covered by Platform Symphony's GPFS cluster.

For data-intensive workloads, we recommend building a GPFS FPO cluster over all Platform Symphony nodes. GPFS FPO makes applications aware of where the data chunks are kept. It helps data-aware scheduling (data affinity) to intelligently schedule application tasks and improve performance by taking into account data location when dispatching the tasks (a GPFS API maps each data chunk to its node location). Data affinity is a feature that is available in the Advanced Edition version of Platform Symphony. Data chunks allow applications to define their own logical block size; therefore, GPFS FPO via optimized variable block-sizes provides good performance across diverse types of workloads.

GPFS is flexible in how its node roles might be assigned. In the context of GPFS FPO, the following recommendations are suggested:

- Assign all Platform Symphony management nodes as GPFS nodes. All of these nodes require a GPFS server license.
- Platform Symphony's GPFS cluster is distributed over several physical racks with both management and compute nodes also distributed evenly across these racks.
- The GPFS cluster primary and secondary cluster configuration nodes should be on separate racks for resiliency.
- Ideally, there should be at least one GPFS quorum node per rack. There should be an odd number of quorum nodes to a maximum of seven.
- Ideally, there should be at least one GPFS metadata disk node per rack.
- Each GPFS metadata node should have at least two disks for metadata, and all metadata nodes should have the same number of metadata disks. For better performance, consider the use of solid-state drives (SSDs) for the metadata. For better fault tolerance, it is recommended to have at least four nodes with metadata disks when you are using metadata replication that is equal to three.
- All Platform Symphony compute nodes should be GPFS FPO data disk nodes. All of these nodes require a GPFS FPO license.

- Use hybrid allocation treating metadata and data differently. Metadata is suited for regular GPFS and data is suited for GPFS FPO. Make the allocation type as a storage pool property accordingly. For metadata pools, use the no write affinity and for data pools use the write affinity.

Platform Symphony's GPFS cluster architecture is shown in Figure 2-14, which relies on local disks to store data. The use of local disks provides performance and cost improvement over other solutions. Because a local disk is available to one server only, data replication is also used for data redundancy and to enable scheduling tasks to where the data exists. For more information, see *Best Practices: Configuring and Tuning IBM Platform Symphony and IBM GPFS FPO*, which is available at this website:

https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20Platform%20Symphony%20Wiki/page/Focus%20on%20Big%20Data

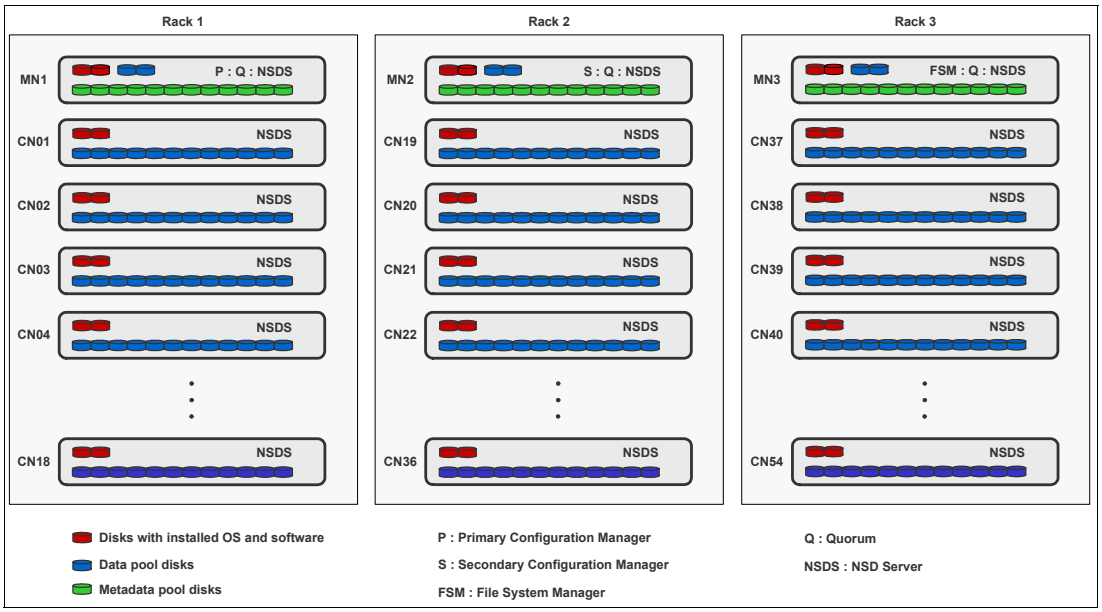

Figure 2-14 Sample GPFS FPO cluster nodes and disks

Platform Symphony master node

The Platform Symphony master node (MN1) provides grid resource management. If it fails, this service fails over to other master candidate nodes. A similar failover strategy can be implemented for the SSM, which is running on the Platform Symphony management node. A shared file system among the management nodes facilitates the failover strategy. Within Platform Symphony's GPFS cluster, the server acts as a GPFS quorum node and primary configuration manager node.

To achieve the highest degree of performance and scalability, we recommend using a powerful master host. We recommend the use of multi-core CPUs with sufficient physical memory (we recommend the use of 8 GB RAM per core). The Platform Symphony master node connects to the public, provisioning, and high-speed interconnect networks.

Platform Symphony management node

The Platform Symphony management node (MN2) runs session managers (there is one session manager per available slot on a management host, and one session manager per application), and provides multiple ways for workload submission. From the administration perspective, the management tasks can be performed from the web-based Platform Management console GUI that is running on the Platform Symphony management node and the command line.

The GUI is a modern and complete web-based portal for management, monitoring, reporting, and troubleshooting purposes. As a differentiator feature, it offers a high level of interactivity with the running jobs (suspend, resume, and kill job and tasks, and can even change the priority of a running job). If the Platform Symphony management node fails, the services fail over to a master candidate node. A shared file system among the management nodes facilitates the failover strategy.

Within Platform Symphony's GPFS cluster, the server acts as a GPFS quorum node and secondary configuration manager node. To achieve the highest degree of performance and scalability, we recommend the use of a powerful master host. We also recommend the use of multi-core CPUs with sufficient physical memory (8 GB RAM per core is recommended). The Platform Symphony management node connects to the public, provisioning, and high-speed interconnect networks.

Platform Symphony master candidate node

The Platform Symphony master candidate node (MN3) is used for failover of any of the management nodes (MN1 or MN2). The configuration of the master candidate node is the same as the Platform Symphony master node. Within Platform Symphony's GPFS cluster, the server acts as a GPFS quorum node and file system manager node. The master candidate node connects to the public, provisioning, and high-speed interconnect networks.

Platform Symphony compute node and data node

The Platform Symphony compute and data nodes (CN01 to CN54) are designed to run Platform Symphony's SOA execution management services to support running and managing compute-intensive or data-intensive applications that are scheduled by the upper-level SOA workload management services. Platform Symphony EGO services that are running on all data nodes collect computational resource information and help Platform Symphony SOA workload management service to schedule jobs more quickly and efficiently.

Within Platform Symphony's GPFS cluster, the GPFS NSD service is running on all compute and data nodes to consolidate all local disks, and provides an alternative distributed file system (DFS) to HDFS. The GPFS FPO function and its specific license is enabled on all compute and data nodes. To achieve the highest degree of performance and scalability, we recommend the use of a powerful master host. We also recommend using multi-core CPUs (one processor core to one local data disk) and sufficient physical memory (a minimum of 8 GB RAM per core, explicitly for analytic applications, is recommended). The Platform Symphony compute and data nodes connect to the provisioning and high-speed interconnect networks.

Networking

The Platform Symphony cluster uses similar networks as the PCM-SE cluster (public, provisioning, monitoring, and application networks). The corporate network (public) represents the outside customer environment. The admin network (provisioning) is a 1 GbE network that is used for the management of all Platform Symphony nodes. The management network (monitoring) is a 1 GbE network that is used for out-of-band hardware management that uses Integrated Management Modules (IMM). Based on customer requirements, the service and management links can be separated into separate VLANs or subnets.

The management network often is connected directly into the client's management network. Data network (application) is a private 10GbE cluster data interconnect among compute and data nodes that are used for data access, moving data across nodes within the cluster, and importing data into GPFS. The Platform Symphony cluster often connects to the client's corporate data network by using one or more management nodes that are acting as interface nodes between the internal cluster and the outside client environment (data that is imported from a corporate network into a cluster, for example). The data network is important to the performance of data-intensive workloads. Use of a dual-port 10GbE adapter in each compute and data node is recommended to provide higher bandwidth and higher availability.

In a multi-rack cluster, we recommend the use top-of-rack switches and set up VLANs for different networks. Each cluster node has two aggregated 10 Gb links that use Link Aggregation Control Protocol (LACP) to the data network, one 1 Gb link to the admin network and 1 Gb link to the IMM network. Each rack has two, 10 Gb top-of-rack switches with HA (that uses inter-switch links at the top-of-rack level). These 10 Gb switches with configured Virtual Link Aggregation Group (vLAG) feature allows multi-switch link aggregation, which provides higher performance and optimizes parallel active-active forwarding.

Each rack also has one, 1 Gb top-of-rack switch that is dedicated to the administration and IMM networks with two 10 Gb uplinks to the up-level switches. From each top-of-rack switch, aggregated uplinks can be configured to the up-level spine switches to build a redundant Two-Tier and Layer 3 Fat Tree network. The Equal-Cost Multi-Path Routing (ECMP) L3 implementation is used for scalability if a Virtual Router Redundancy Protocol (VRRP) is not applicable.

Note: There is no definitive rule between a Layer 2 and a Layer 3 network configurations. However, a reasonable measure is that L3 should be considered whether the cluster is expected to grow beyond 5 - 10 racks.

The important task is to find out how the Platform Symphony cluster fits in a customer environment. We recommend that you to work with a network architect to collect the customer network requirements and to customize the network configurations. You need to know the following information:

- The different methods s of data movement in and out of the Platform Symphony cluster
- Customer network bandwidth requirements
- Customer corporate network standards
- If the segment allocated in the corporate network has enough room for IP allocation growth

2.3 Reference architectures

IBM Application Ready Solutions for Technical Computing are based on IBM Platform Computing software and powerful IBM systems, which are integrated and optimized for leading applications and backed by reference architectures. IBM created Application Ready Solution reference architectures for target workloads and applications. Each of these reference architectures includes recommended small, medium, and large configurations that are designed to ensure optimal performance at entry-level prices. These reference architectures are based on powerful, predefined, and tested infrastructure with a choice of the following systems:

- IBM Flex System™ provides the ability to combine leading-edge IBM POWER7®, IBM POWER7+™ and x86 compute nodes with integrated storage and networking in a highly dense, scalable blade system. The IBM Application Ready Solution supports IBM Flex System x240 (x86), IBM Flex System p260, and p460 (IBM Power) compute nodes.
- IBM System x helps organizations address their most challenging and complex problems. The Application Ready Solution supports IBM NeXtScale System, a revolutionary new x86 high-performance system that is designed for modular flexibility and scalability, System x rack-mounted servers and System x iDataPlex dx360 M4 systems are designed to optimize density, performance, and graphics acceleration for remote 3-D visualization.
- IBM System Storage DS3524 is an entry-level disk system that delivers an ideal price and performance ratio and scalability. You also can choose the optional IBM Storwize® V7000 unified for enterprise-class, midrange storage that is designed to consolidate block-and-file workloads into a single system.
- IBM Intelligent Cluster is a factory-integrated, fully tested solution that helps simplify and expedite deployment of x86-based Application Ready Solutions.

The solutions also include the following pre-integrated IBM Platform Computing software that is designed to address technical computing challenges:

- IBM Platform HPC is a complete technical computing management solution in a single product, with a range of features that are designed to improve time-to-results and help researchers focus on their work rather than on managing workloads.
- IBM Platform Cluster Manager Standard Edition provides easy-to-use yet powerful cluster management for technical computing clusters that simplifies the entire process, from initial deployment through provisioning to ongoing maintenance.
- IBM Platform LSF provides a comprehensive set of tools for intelligently scheduling workloads and dynamically allocating resources to help ensure optimal job throughput.
- IBM Platform Symphony delivers powerful enterprise class management for running Big Data, analytics, and compute-intensive applications.
- IBM General Parallel File System (GPFS) is a high-performance enterprise file management platform for optimizing data management.

2.3.1 IBM Application Ready Solution for Abaqus

The IBM Application Ready Solution for Abaqus is a technical computing architecture that supports linear and nonlinear structural mechanics and multiphysics simulation capabilities in Abaqus, which is part of the SIMULIA realistic simulation software applications that are available from Dassault Systèmes. Abaqus provides powerful structural and multiphysics simulation capabilities that are based on the finite element method. It is sold globally by Dassault Systèmes and their reseller channel.

For more information, see the following resources:

- *IBM Application Ready Solution for Abaqus: An IBM Reference Architecture based on Flex System, NextScale and Platform Computing v1.0.1*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12368usen/DCL12368USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.2 IBM Application Ready Solution for Accelrys

Designed for healthcare and life sciences, the Application Ready Solution for Accelrys simplifies and accelerates mapping, variant calling, and annotation for the Accelrys Enterprise Platform (AEP) NGS Collection. It addresses file system performance (the biggest challenge for NGS workloads on AEP) by integrating IBM GPFS for scalable I/O performance. IBM systems provide the computational power and high-performance storage that is required, with simplified cluster management to speed deployment and provisioning.

For more information, see the following resources:

- *IBM Application Ready Solution for Accelrys: An IBM Reference Architecture based on Flex System, System x, and Platform Computing V1.0.3*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12370usen/DCL12370USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.3 IBM Application Ready Solution for ANSYS

The IBM Application Ready Solution for ANSYS is a technical computing architecture that supports software products that are developed by ANSYS in the areas of computational fluid dynamics and structural mechanics. ANSYS computational fluid dynamics (CFD) software solutions (including ANSYS Fluent and ANSYS CFX) are used to predict fluid flow and heat and mass transfer, chemical reactions, and related phenomena by numerically solving a set of governing mathematical equations (conservation of mass, momentum, energy, and others). ANSYS Structural Mechanics software (including ANSYS Mechanical) offers a comprehensive solution for linear or non-linear and dynamic analysis. It provides a complete set of elements behavior, material models, and equation solvers for various engineering problems.

For more information, see the following resources:

- *IBM Application Ready Solution for ANSYS: An IBM Reference Architecture based on Flex System, NeXtScale, and Platform Computing v2.0.1*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12372usen/DCL12372USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.4 IBM Application Ready Solution for CLC bio

This integrated solution is designed for clients who are involved in genomics research in areas ranging from personalized medicine to plant and food research. Combining CLC bio software with high-performance IBM systems and GPFS, the solution accelerates high-throughput sequencing and analysis of next-generation sequencing data while improving the efficiency of CLC bioGenomic Server and CLC Genomics Workbench environments.

For more information, see the following resources:

- *IBM Application Ready Solution for CLC bio: An IBM Reference Architecture based on Flex System, System x, and Platform Computing V1.0.2*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12371usen/DCL12371USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.5 IBM Application Ready Solution for Gaussian

Gaussian software is widely used by chemists, chemical engineers, biochemists, physicists, and other scientists who are performing molecular electronic structure calculations in various market segments. The IBM Application Ready Solution is designed to help speed results by integrating the latest version of the Gaussian series of programs with powerful IBM Flex System POWER7+ blades and integrated storage. IBM Platform Computing provides simplified workload and resource management.

For more information, see the following resources:

- *IBM Application Ready Solution for Gaussian: An IBM Reference Architecture based on POWER Systems V1.0.1* is available at:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12373usen/DCL12373USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.6 IBM Application Ready Solution for InfoSphere BigInsights

The Application Ready Solution for IBM InfoSphere BigInsights provides a powerful Big Data MapReduce analytics environment and reference architecture that is based on IBM PowerLinux™ servers, IBM Platform Symphony, IBM GPFS, and integrated storage. The solution delivers balanced performance for data-intensive workloads, with tools and accelerators to simplify and speed application development. The solution is ideal for solving time-critical, data-intensive analytics problems in various industry sectors.

For more information, see the following resources:

- *IBM Application Ready Solution for InfoSphere BigInsights: An IBM Reference Architecture V1.0*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12376usen/DCL12376USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.7 IBM Application Ready Solution for mpiBLAST

The mpiBLAST is a freely available, open source, parallel implementation of National Center for Biotechnology Information (NCBI) Basic Local Alignment Search Tool (BLAST). IBM Application Ready Solution for mpiBLAST simplifies the deployment of a life sciences open source parallel BLAST simulation environment. It provides an expertly designed, tightly integrated, and performance optimized architecture based on Flex System, System x, and Platform Computing for simplified workload and resource management.

For more information, see the following resources:

- *IBM Application Ready Solution for mpiBLAST: An IBM Reference Architecture based on Flex System, System x, and Platform Computing Version 1.0*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12377usen/DCL12377USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.8 IBM Application Ready Solution for MSC Software

The IBM Application Ready Solution for MSC Software features an optimized platform that is designed to help manufacturers rapidly deploy a high-performance simulation, modeling, and data management environment, complete with process workflow and other high-demand usability features. The platform features IBM systems (IBM Flex System and IBM NeXtScale System), Platform HPC workload management, and GPFS parallel file system that are seamlessly integrated with MSC Nastran, MSC Patran, and MSC SimManager to provide clients robust and agile engineering clusters for accelerated results and lower cost.

For more information, see the following resources:

- *IBM Application Ready Solution for MSC: An IBM Reference Architecture V1.0.1*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12367usen/DCL12367USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.9 IBM Application Ready Solution for Schlumberger

Fine-tuned for accelerating reservoir simulations that use Schlumberger ECLIPSE and INTERSECT, this Application Ready Solution provides application templates to reduce set up time and simplify job submission. Designed specifically for Schlumberger applications, the solution enables users to perform more iterations of their simulations and analysis, which ultimately yields more accurate results. Easy access to Schlumberger job-related data and remote management improves user and administrator productivity.

For more information, see the following resources:

- *IBM Application Ready Solution for Schlumberger: An IBM Reference Architecture based on Flex System, System x, and Platform Computing V1.0.3* is available at:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12374usen/DCL12374USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.10 IBM Application Ready Solution for Technical Computing

The IBM Application Ready Solutions for Technical Computing architecture supports both compute and data intensive applications. Technical computing users are often technical within their respective field (engineers who are designing automobile parts, for example), but not experts in computer and software technology. Independent software vendors (ISVs) made significant investments to increase their application's performance and capability by enabling them to run in a distributed computing cluster environment. However, many users are unable to fully use the capabilities of these applications because they do not have the technical ability to efficiently deploy and manage a technical or HPC cluster.

For more information, see the following resources:

- *IBM Application Ready Solutions for Technical Computing: An IBM Reference Architecture based on Flex System, NextScale, System x, and Platform Computing V2.0*, which is available at this website:

  http://public.dhe.ibm.com/common/ssi/ecm/en/dcl12369usen/DCL12369USEN.PDF

- IBM Platform Computing:

  http://www.ibm.com/technicalcomputing/appready

2.3.11 IBM System x and Cluster Solutions configurator

The IBM System x and Cluster Solutions configurator (x-config) is the hardware configurator that supports the configuration of Cluster Solutions. The reference architectures for IBM Application Ready Solutions are provided as a predefined template within x-config. The configurator is a stand-alone, Java based application that runs on a workstation after it is downloaded and does not require an internet connection. However, if an internet connection is available, the most recent version is installed automatically when the tool is started.

The configurator is available at this website:

http://www.ibm.com/products/hardware/configurator/americas/bhui/asit/index.html

Complete the following steps to access the predefined templates:

1. From the x-config starter window (as shown on Figure 2-15), select **Express** and **Cluster Support ON**.

Figure 2-15 x-config starter page

2. In the next window, select **No-options solutions** from the Solution drop-down menu.

3. Select **IBM Application Ready Solutions** from the Type drop-down menu.
4. Select the applicable Application Ready Solution from the Template drop-down menu.
5. After the template is loaded into configurator, click **View details of this configuration** to see a complete list of parts and list prices.
6. Click **Configure** to customize the predefined solution that is based on your requirements.

The x-config provides support for the configuration of Intelligent Clusters, iDataPlex systems, and stand-alone System x servers. Its intention is to translate rack-level architectural design into orderable configurations. It is also a single tool to support the design and configuration of multiple product lines (Blade, iDPx, Power, Flex, System x Rack Mount servers, racks, switches, cables, PDUs, and so on).

It also provides rack-level diagramming so that you can visualize the component location within a rack, which is important within a data center design; for example, where items are in the rack, airflow, and how much power you put in the rack. It also expands out-to-floor layouts, which is critical, especially in the high-performance computing area when you have specific networks that are expensive. Therefore, the ability to model the actual data center and floor can give you specific cable lengths. You can also view list prices as they are configured so that you can estimate more accurately where you are at list prices.

The x- config should not be considered a design tool because. Instead, it allows architects to translate designs into priceable, orderable, and buildable solutions and aids the user by providing the following benefits:

- Performing system-level checks and validation (SOVA)
- Aiding in calculating cable lengths that are based on the floor locations of the racks
- Having a high-level rule enforcement that is based on best-practices

2.4 Workload optimized systems

Workload optimized hardware and software nodes are key building blocks for every technical computing environment. To review the potential of the use of workload optimized systems with IBM Platform Computing products, you can use the reference architectures as part of an overall assessment process with a customer. While you are working on a proposal with a client, you can discover and analyze the client's technical requirements and expected usage (hardware, software, data center, workload, current environment, user data, and high availability).

The following hardware evaluations must be considered:

- Determine data storage requirements, including user data size and compression ratio.
- Determine shared storage requirements.
- Determine whether data node OS disks require mirroring.
- Determine memory requirements.
- Determine throughput requirements and presumable bottlenecks.

The following software aspects must be considered:

- Identify cluster management strategy, such as node firmware and OS updates.
- Identify a cluster rollout strategy, such as node hardware and software deployment.
- Determine use of the GPFS performance.

The following data center aspects must be evaluated and considered:

- Determine cooling requirements, such as airflow and BTU requirements.
- Determine server spacing, racking, networking and electrical cabling, and cooling.

The following workload aspects must be considered:

- Determine workload characteristics, such as performance sensitive, compute-intensive, data-intensive, or a combination.
- Identify workload management strategy.
- Determine business-driven scheduling policies.

The following current environment aspects must be considered:

- Determine customer corporate networking requirements, such as networking infrastructure and IP addressing.
- Determine data storage and memory existing shortfalls.
- Identify system usage inefficiencies.

The following user data aspects must be considered:

- Determine the current and future total data to be managed.
- Determine the size of a typical data set.
- In case of import, specify the volume of data to be imported and import patterns.
- Identify the data access and processing characteristics of common jobs and whether they are query-like frameworks.

The following high availability aspects must be considered:

- Determine high availability requirements.
- Determine multi-site deployments, if required.
- Determine disaster recovery requirements, including backup and recovery and multi-site disaster recover requirements.

Recommendation: To design an HPC cluster infrastructure, conduct the necessary testing and proof of concepts against representative data and workloads to ensure that the proposed design achieves the necessary success criteria.

2.4.1 NeXtScale System

NeXtScale System is the next generation dense system from System x for clients that require flexible and scale-out infrastructure. The building blocks include dense 6U chassis (NextScale n1200) and contain 12 bays for half-wide compute (NeXtScale nx360 M4), storage (Storage NeX), and planned acceleration via graphics processing unit (GPU) or Intel Xeon Phi coprocessor (PCI NeX). NeXtScale System uses industry-standard components, including I/O cards and top-of-rack networking switches for flexibility of choice and ease of adoption.

From the performance point of view, the NeXtScale server solution benefits from the use of (IVB EP) next generation Intel processors top bin E-5 2600 v2 processors (up to 24 cores, 48 threads per server), fasted memory that is running at 1866 MHz (up to 256 GB per server), choice of SATA, SAS, or SSD on board (ultimate in I/O throughput that uses SSDs), and open ecosystem of high-speed I/O interconnects (10 GB Ethernet, QDR/FDR10, or FDR14 InfiniBand via slotless Mezz adapter).

From the operation point of view, the NeXtScale solution benefits from the use of S3, which allows systems to come back into full production from a low-power state much quicker (only 45 seconds) than a traditional power-on (270 seconds). When you know that a system will not be used because of time of day or state of job flow, it can be sent into a low-power state to save power and bring it back online quickly when needed.

By using 80+ Platinum power supplies that are operating at 94% efficiency to save power and shared in the chassis with 80 mm fans, shared power and cooling is provided for all nodes installed. NeXtScale reduces the total number of parts that are needed for power and cooling solution, which saves money in part cost and reduces the number of PSUs and fans, which reduces power draw.

Servicing NeXtScale from the front of the rack is an easier task. Everything is right there in front of you, including the power button, cabling, alert leds, and node naming and tagging. It reduces chances of missing cabling or pulling a wrong server. Having the cables arranged in the back of the rack also is good for air flow and good for energy efficiency.

The hyper-scale server NeXtScale nx360 M4 provides a dense, flexible solution with a low total cost of ownership. The half-wide, dual-socket NeXtScale nx360 M4 server is designed for data centers that require high performance but are constrained by floor space. By taking up less physical space in the data center, the NeXtScale server significantly enhances density.

NeXtScale System can provide up to 84 servers (or 72 servers with space for six 1U switches) that are installed in a standard 42U rack. Supporting Intel Xeon E5-2600 v2 series up to 130 W and 12-core processors provides more performance per server. The nx360 M4 compute node contains only essential components in the base architecture to provide a cost-optimized platform.

Native expansion means that we can add function and capabilities seamlessly to the basic node. There is no need for exotic connectors, unique components, or high-speed back or mid planes. NeXtScales Native Expansion capability adds hard disk drives (HDDs) to the node with a simple storage NeX (tray) + standard RAID card, SAS cable, and HDDs. Adding GPUs to a node is done through a PCI NeX supplementing PCI riser and a passively cooled GPU from nVidia or Intel. You also have a powerful acceleration solution for HPC, virtual desktop, or remote graphics (two GPUs per server in 1U effective space).

For more information about how to implement NeXtScale System and positioning within other System x platforms (iDataPlex, Flex System, rack-mounted System x servers), see *IBM NeXtScale System Planning and Implementation Guide*, SG24-8152, which is available at this website:

http://www.redbooks.ibm.com/abstracts/sg248152.html

For more information about the NeXtScale System product, see this website:

http://www.ibm.com/systems/x/hardware/highdensity/nextscale/index.html

2.4.2 iDataPlex

IBM System x iDataPlex is an innovative data center solution that maximizes performance and optimizes energy and space efficiencies. The building blocks include the iDataPlex rack cabinet, which offers 100 rack units of space (up to 84 servers per iDataPlex rack, 8 top-of-rack switches, 8 PDUs), 2U FlexNode chassis that supports up to two half-depth 1U dx360 M4 compute nodes, which can be extended with PCIe trays that support I/O- and GPGPU-intensive applications.

The iDataPlex rack has 84U slots for server chassis and 16 vertical slots for network switches, PDUs, and other appliances. The rack is oriented so that servers fit in side-by-side on the widest dimension. For ease of serviceability, all hard disk, planar, and I/O access is from the front of the rack. In addition, the optional liquid-cooled Rear Door Heat eXchanger that is mounted to the back of the rack can remove 100% of the heat that is generated within the rack, which draws it from the data center before it exits the rack. It can also help to cool the data center and reduce the need for Computer Room Air Conditioning (CRAC) units. This allows racks to be positioned much closer together, which eliminates the need for hot aisles between rows of fully populated racks.

Note: IBM also supports the installation of iDataPlex servers in standard 19-inch racks.

The highly dense iDataPlex dx360 M4 server is a modular solution with a low total cost of ownership. The unique half-depth, dual-socket iDataPlex dx360 M4 server is designed for data centers that need energy efficiency, optimized cooling, extreme scalability, high density at the data center level, and high performance at an affordable price. Supporting Intel Xeon E5-2600 v2 series up to 130 W processors provides more performance per server (up to 24 cores, and 48 threads) and maximize the concurrent execution of multi-threaded applications.

Each 2U chassis is independently configurable to maximize compute, I/O, or storage density mix configurations for a tailored dense solution. You can use faster memory that is running at 1866 MHz (up to 512 GB per server), choice of SATA, SAS, or SSD on board, select high-speed I/O interconnects (two 10 GB Ethernet ports, or two FDR14 InfiniBand ports via slotless Mezz adapter). The chassis use a shared fan pack with four 80 mm fans and redundant highly efficient (80 PLUS Platinum) power supplies. With the iDataPlex chassis design, air needs to travel only 20 inches front to back (shorter distance means better airflow). This shallow depth is part of the reason that the cooling efficiency of an iDataPlex server is high.

For more information about the iDataPlex System and liquid-cooling, see these resources:

- *Implementing an IBM System x iDataPlex Solution*, SG24-7629-04, which is available at this website:

  http://www.redbooks.ibm.com/abstracts/sg247629.html
- http://www.ibm.com/systems/x/hardware/highdensity/dx360m4/index.html
- http://www.redbooks.ibm.com/abstracts/tips0878.html

2.4.3 Intelligent Cluster

The IBM Intelligent Cluster is a factory-integrated, interoperability-tested system with compute, storage, networking, and cluster management that is tailored to your requirements and supported by IBM as a solution. With optimized solution design and by using interoperability-tested best-of-industry technologies, it simplifies complex solutions and removes the time and risk within the deployment.

The Intelligent Cluster solution includes the following building blocks:

- IBM System x:
  - IBM NeXtScale System: nx360 M4
  - IBM Flex System: x220, x240, x440 compute nodes
  - Blade servers: HX5, HS23
  - Enterprise servers: x3850 X5, x3690 X5
  - iDataPlex servers: dx360 M4
  - Rack servers: x3550 M4, x3630 M4, x3650 M4, x3750 M4, and x3650 M4 HD

- Interconnects:
  - Ethernet Switches: IBM System Networking, Brocade, Cisco, Mellanox, Edgecore,
  - Ethernet Adapters: Chelsio, Mellanox, Emulex, Intel
  - InfiniBand Switches and Adapters: Mellanox, Intel
  - Fibre Channel: Brocade, Emulex, and Intel
- Storage systems (System Storage): DS5020, DS5100, DS5300, DS3950, DS3500, DS3512, DS3524, and IBM Storwize V3700
- Storage expansions: EXP5000 Storage Expansion Unit, EXP 2512 Storage Expansion Unit, EXP 2524 Storage Expansion Unit, EXP 520 Storage Expansion Unit, and EXP 395 Storage Expansion Unit
- OEM storage solution: DDN SFA 12000 InfiniBand (60 and 84 drive enclosures)
- Graphic Processing Units (GPUs): NVIDIA: Quadro 5000, Tesla K10, Tesla M2070Q, Tesla M2090, Tesla K20, and Tesla K20X
- Operating systems: Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES)
- Cluster management software:
  - IBM Platform HPC
  - IBM Platform Cluster Manager
  - IBM Platform LSF, xCAT (Extreme Cloud Administration Toolkit)
  - Moab Adaptive HPC SuiteMoab Adaptive Computing Suite
  - IBM General Parallel File System (GPFS) for Linux
  - IBM LoadLeveler®
  - IBM Parallel Environment

For more information about Intelligent Cluster, see this website:

http://www.ibm.com/systems/x/hardware/highdensity/cluster/index.html

2.4.4 Enterprise servers

The new IBM X6 enterprise servers are high-end servers that are designed for heavy vertical workloads, virtualization, and legacy system replacements. IBM invested in its enterprise X Architecture to deliver industry-leading performance, scalability, and reliability on industry standard x86-based systems. IBM X6 rack-mount servers are available in four-socket (x3850 X6) and eight-socket (x3950 X6) and incorporate a new book design.

The IBM X6 server offer pay-as-you-grow scalability. The System x3850 X6 server features a modular design that includes so-called books for each of the three subsystems (I/O, storage, and compute). Front and rear access means that you can easily add and remove the various components without removing the server from the rack, which is a revolutionary concept in rack servers. You add components as you need them, and some components, such as storage and I/O adapters, are hot-swappable, so you do not need to power off the server to add them. With a design that helps prevent component failures from bringing down the entire machine, you can feel confident that an X6 server is an ideal platform for any mission-critical application.

The following building blocks for new X6 servers are available:

- Compute books

  Each compute book contains one processor (Intel Xeon E7-4800v2 or E7-8800v2) and 24 DIMM slots. It is accessible from the front of the server. The x3850 X6 has up to four compute books. The x3950 X6 has up to eight compute books (with support for 64 GB LRDIMMs, you can have up to 6 TB of memory in the x3850 X6 or 12 TB in the x3950 X6).

- Storage books

  The storage book contains standard 2.5-inch drives or IBM eXFlash 1.8-inch SSDs (up to 12.8 TB of SAS 2.5-inch disk or up to 6.4 TB of eXFlash 1.8-inch SSDs). It also provides front USB and video ports, and has two PCIe slots for internal storage adapters. The storage book is accessible from the front of the server.

- I/O books

  The I/O book is a container that provides PCIe expansion capabilities. I/O books are accessible from the rear of the server. The rear contains the primary I/O book, optional I/O books, and up to four 1400 W/900 W AC or 750 W DC power supplies.

The following types of I/O books are available:

- Primary I/O book

  This book provides core I/O connectivity, including a dedicated mezzanine LOM (ML) slot for an onboard network, three PCIe slots, an Integrated Management Module II, and four rear ports (USB, video, serial, and management).

- Full-length I/O book

  This hot-swappable book provides three full-length PCIe slots. This book supports a co-processor or GPU adapter up to 300 W if needed.

- Half-length I/O book

  This hot-swappable book provides three half-length PCIe slots.

The X6 offering also includes new IBM eXFlash memory-channel storage and IBM FlashCache Storage Accelerator, two key innovations that eliminate storage bottlenecks. This new IBM eXFlash memory-channel storage brings storage closer to the processor subsystem, which improves performance considerably.

These storage devices have the same form factor as regular memory DIMMs, are installed in the same slots, and are directly connected to the memory controller of the processor. IBM eXFlash DIMMs are available in 200 GB and 400 GB capacities and you can install 32 of them in a server.

New eXFlash DIMMs have latency values lower than any SSD or PCIe High IOPS adapter. This represents a significant advantage for customers who need the fastest access to data. FlashCache Storage Accelerator is intelligent caching software and uses intelligent write-through caching capabilities to achieve better IOPS performance, reduced I/O load, and ultimately increased performance in primary storage.

All these features mean that the new X6 enterprise servers offer excellent upgradability and investment protection.

For more information, see these websites:

- http://www.ibm.com/systems/x/x6/index.html
- http://www.ibm.com/systems/info/x86servers/ex5/index.html

2.4.5 High volume systems

The volume space is over half the total x86 server market and IBM has a broad portfolio of rack servers to meet various client needs, from infrastructure to technical computing specially for the analytics (compute-intensive workloads) and Big Data (data-intensive workloads) where the new System x3650 M4 server types (HD and BD) are focused.

The new System x3650 M4 HD (High Density) two-socket 2U server is optimized for high-performance, storage-intensive applications, including data analytics or business-critical workloads. Supporting Intel Xeon E5-2600 v2 series up to 130 W processors provides more performance per server (up to 24 cores, and 48 threads) and maximizes the concurrent execution of multi-threaded applications.

You can use faster memory that is running at 1866 MHz (up to 768 GB per server with 30 MB cache), which yields low latency for data access and faster response times. It has 12 Gb RAID on board, which doubles the bandwidth of the x3650 M4 for optimized performance for data protection. It also allows up to 4 RAID adapters, which provides flexible storage controller configurations for up to 4x performance increase for demanding storage intensive workloads versus single RAID adapter design.

It provides flexible internal storage options, including 26 x 2.5-inch HDD or SSD or 16 x 2.5-inch HHD or SSD + 16 x 1.8-inch SSD, up to 41 TB. You can select up to 16 HDDs + 16 SSDs for optimum storage performance through storage tiering and use IBM FlashCache Storage Accelerator option to deliver high I/O performance without the need for tuning.

It also provides the ability to boot from the rear HDDs with a separate optional RAID adapter to keep OS and business data separate, which means easy set up, management, and configuration. You can use standard and high-speed I/O interconnects (4 x 1 Gb Ethernet ports + 1 Gb IMM + optional 2 x 10 Gb Ethernet NIC design where no PCIe slot is required).

Extensions are provided through up to six PCIe 3.0 ports (you can add up to two optional GPUs) or optional 4 PCI-X. The server optionally supports up to two GPUs (NVIDIA adapters). On the rear, it can use up to 4x Hot swap redundant fans and 2x Hot swap redundant Efficient 80+ Platinum Power Supply Units.

For more information about the System x3650 M4 HD product, see this website:

http://www.ibm.com/systems/x/hardware/rack/x3650m4hd/index.html

For more technical information about the System x3650 M4 HD, see this website:

http://www.redbooks.ibm.com/abstracts/tips1049.html

The new System x3650 M4 BD (Big Data) two-socket 2U server is optimized for the capacity, performance, and efficiency you need for Big Data workloads. Supporting Intel Xeon E5-2600 v2 series up to 115 W processors provides more performance per server (up to 24 cores, and 48 threads) for fast response time and business outcomes for Big Data workloads. You can use faster memory that is running at 1866 MHz (up to 512 GB per server with 30 MB cache). For instance, it supports 1+1 RAID, which enables the ability to boot from the separate rear drives, which keeps the OS and business data separate.

It also offers flexible options, including a choice of JBOD for maximum capacity or RAID for up to 12 Gb for optimal data protection that is supported by up to 1 GB, 2 Gb, or 4 GB flashed-back cache. Another option is 6 Gb RAID with 200 or 800 GB flash for caching or data or boot volume.

It provides flexible internal storage options by choosing 12+2 x3.5-inch Hot Swap SATA HDD, and 2.5-inch SSDs up to 56 TB. You can use standard and high-speed I/O interconnects (4 x 1 Gb Ethernet ports + 1 Gb IMM + optional 2 x 10 Gb Ethernet NIC design where no PCIe slot is required). Extensions are provided through up to five PCIe 3.0 ports. On the rear, it can use up to 4x Hot swap redundant fans and 2x Hot swap redundant Efficient 80+ Platinum PSU.

For more information about the System x3650 M4 BD product, see this website:

http://www.ibm.com/systems/x/hardware/rack/x3650m4bd/index.html

For more technical information about System x3650 M4 BD, see this website:

http://www.redbooks.ibm.com/abstracts/tips1102.html

3

IBM Platform High Performance Computing implementation scenario

Implementing IBM Platform high performance computing (HPC) can be easily accomplished, and the process is described in this chapter. To better describe its implementation, you can find a case study (scenario) that implements the most common features when a cluster is built with IBM Platform HPC. The aim of this scenario is to guide you through the key building blocks of a complex HPC cluster that is built by using the IBM Platform HPC suite. Even if you are not using all of the components that are described in this chapter, it is worth reading it completely as the chapter brings a broader view of building a cluster and can help you build your own solution.

This chapter is not intended to replace the installation or administration manuals, but to give you a guideline of how to install and manage a cluster with the IBM Platform HPC from the residency team perspective. If you do not find all of the steps to install the cluster, see the product's installation manuals (for more information, see "Related publications" on page 185).

The objective of this scenario focuses on a high availability implementation for the IBM Platform HPC (pHPC) with GPFS. Every section starts with a description of the concepts that are used and continues with practical examples. For example, in the next sections we describe the following tasks:

- Installing pHPC and other software, such as GPFS and ISV application.
- Provisioning GPFS server nodes and compute nodes.

This chapter includes the following topics:

- Application Ready Solution versus scenario-based implementation
- Scenario-based implementation

Note: In this book, we interchange the terms *IBM Platform HPC* and *pHPC*.

3.1 Application Ready Solution versus scenario-based implementation

IBM has several bundled offers for IBM Platform HPC, which are named Application Ready Solutions. These bundles are most commonly represented by specific IBM hardware (Flex System, NeXtScale, and System x), a specific version of IBM Platform HPC, and an ISV application (or applications), such as Abaqus, ANSYS, Accelrys, InfoSphere BigInsights, CLC bio, Gaussian, mpiBLAST, MSC Software, and Schlumberger.

If you purchase an Application Ready Solution, everything is delivered, including the hardware and a `.tar` archive file with all of the requirements to automatically install your cluster, along with all the customizable scripts. For example, if you purchase GPFS for your cluster, you receive in this bundle all of the scripts to automatically provision the GPFS cluster.

IBM Application Ready Solution provides an expertly designed, tightly integrated, and performance-optimized architecture for technical computing applications. These bundles are targeted mostly for domain experts, which require technical or HPC clusters that can be quickly deployed without extensive computer skills.

This document is not intended to overlap with the Application Ready Solutions guide. Instead, it is intended to be a more generic guide. The idea is to give you an overview of how to install IBM Platform HPC.

3.2 Scenario-based implementation

In building this scenario, we consider the following key aspects of an HPC cluster and the applications that are running on it:

- The application is highly I/O intensive and requires high throughput.
- The application needs low latency communication between cluster nodes.
- There is a need for high availability of the cluster components.
- All cluster components need monitoring.

Because of these aspects, our cluster uses the following components:

- A low latency network, such as InfiniBand.
- A high-performance parallel file system, such as GPFS.
- IBM Platform LSF for workload management.
- IBM Platform Cluster Manager for high availability.

In the next sections, we describe the scenario that we build. We start by installing the management node of the cluster (single head), continue with the GPFS servers installation and configuration, followed by compute nodes provisioning with all of the necessary software installed, including pHPC software stack and the GPFS client software.

Figure 3-1 shows the components of the cluster we build in this chapter.

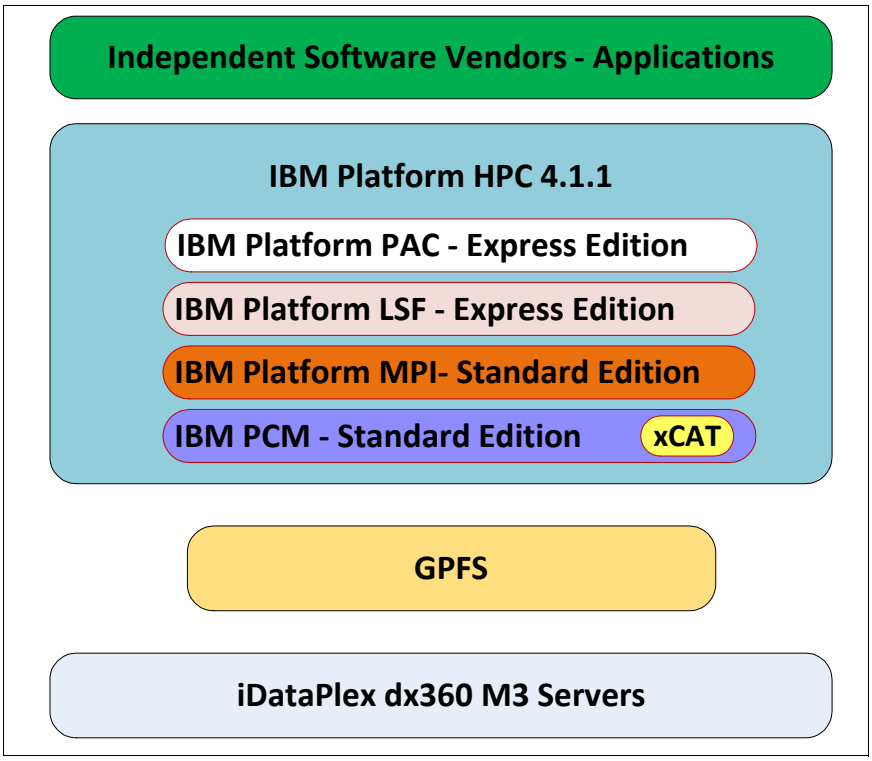

Figure 3-1 Cluster components

All of the software that is installed on the compute nodes is packed as distribution kits. Later in this section, we describe how to build your own kits. Moreover, if your cluster workload management requirements increases over time and you need more advanced Platform LSF features, we show you how to upgrade to the Platform LSF Enterprise Edition (for more information, see “Upgrading Platform LSF” on page 130).

Starting with version 4.1.1.1, IBM Platform HPC supports high availability for the management nodes. To show you this feature while showing the sample scenario, we reconfigure the initial cluster to provide high availability by adding a secondary management node.

Figure 3-2 shows the architecture of the cluster we build, the roles that are assigned to each node, and how they are wired together. We have two management nodes (49 and 50) that are installed with pHPC in high availability configuration, two nodes that act as GPFS server nodes, which are Network Shared Disk (NSD) servers, and the rest of the nodes are used as compute nodes. The GPFS servers can have local disks, SAS-attached, or Fiber Channel-attached disks that are shared with its clients over InfiniBand.

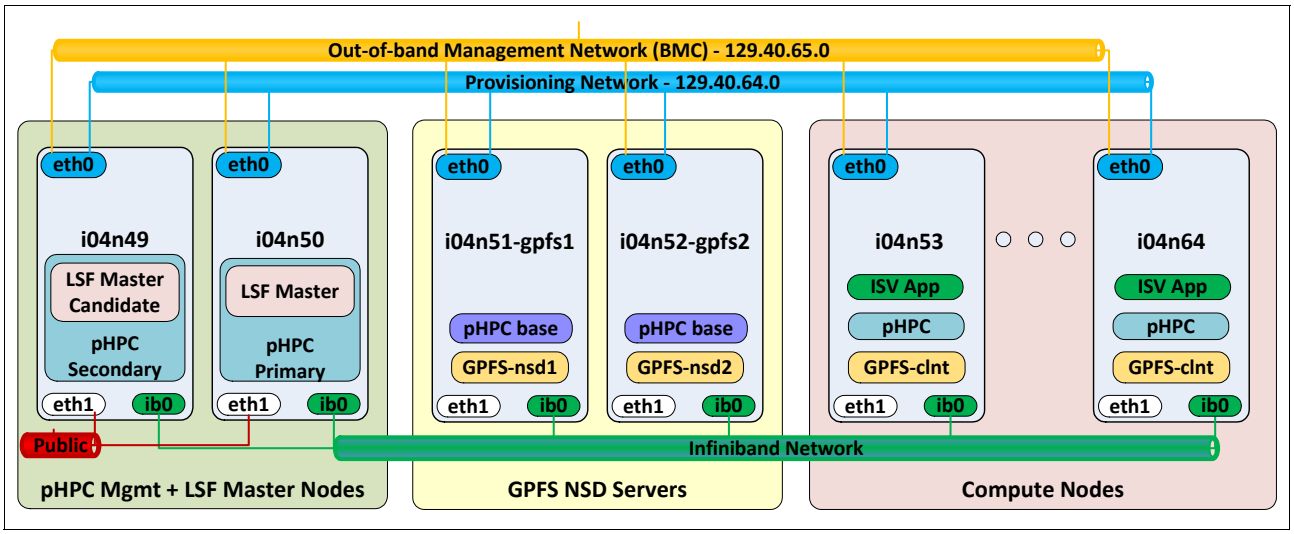

Figure 3-2 Cluster architecture

3.2.1 Cluster hardware

The cluster is built on 16 IBM iDataPlex nodes, which are all placed in the same rack. They are a subset of nodes that are borrowed from a larger cluster, which has 64 nodes. For easy tracking, we do not change their names. Therefore, our cluster node names are in a range i04n[49 - 64].

Note: There is no integration with any clusters that are installed on nodes i04n[1 - 48].

Each iDataPlex server in the cluster is equipped with 2 x CPU (6 cores each), 24 GB RAM, two 1-Gb Ethernet ports and one InfiniBand adapter. The two 1-Gb ports are connected to Ethernet switches for provisioning, out-of-band management, or for public network access (access to management nodes). Later in this chapter, we describe the meaning of each of these networks types. For low latency communication at the application level and for the GPFS file system, the servers are equipped with one port InfiniBand adapter that is connected to a dedicated InfiniBand switch.

In the next section, we describe each component of the cluster, their role and functionalities, along with best practices for a smooth integration into the cluster.

3.2.2 Management nodes

The management nodes play many important roles in the cluster, but the most important roles are the provisioning and the software lifecycle management functions (updates, reinstall, uninstall) of the compute nodes. Starting with version 4.1, IBM Platform HPC uses xCAT as the provisioning engine. To accomplish the provisioning function, IBM Platform HPC provides to the cluster the following suite of services:

- DHCP: The management node listens on the defined provisioning interface and answers to DHCP requests from the compute nodes.
- TFTP: Serves the PXE boot file in early stages of the boot over the network.
- DNS: If you do not have an external DNS server, you can define one on the management node.
- NFS: The management node can export local directories to be mounted on compute nodes. For example, the compute nodes can mount these exports as `/home` for users or `/data` for application data.
- HTTP: Used to share the yum repositories for operating systems or kit distributions to the compute nodes for initial installation or later updates.
- NTP: The management node can act as a time server of the cluster. If you decide not to use it as an NTP server, you can set an external NTP server.

In addition to the provisioning functions, the management node acts as a repository, which stores files and scripts for provisioning compute nodes. The repository can store the following files:

- Kickstart files
- Network boot files (PXE)
- Operating system (OS) installation files and application kits
- Postinstallation and post-boot scripts

At the same time, the management node can act as a web portal to submit, manage, and monitor jobs in the cluster, and runs services for workload management (Platform LSF). Additionally, it performs cluster monitoring of different components, such as Platform LSF (monitoring agent), hardware monitoring services (metric collection, aggregation, alerting, purging), and GPFS monitoring.

The management node also can act as a firewall to shield the cluster from external (public) networks.

3.2.3 Compute nodes

The compute nodes run workload as assigned by the workload manager (Platform LSF). Everything happens on these nodes and is a result of an action that is triggered from the management node (pHPC) or workload-manager node (Platform LSF). In our scenario, the cluster management and workload manager functions overlap on the first two nodes.

Compute node provisioning is the process of installing the operating system and related software on a compute node, sometimes with the initial software configuration. For provisioning to occur, the compute node must boot over the network (PXE).

The following provisioning types of compute nodes are available:

- Stateful provisioning is a method that deploys the operating system and related software onto persistent storage (local disk, SAN disk, or iSCSI device) such that the changes that are made to the operating system are persistent across node reboots.

- Stateless provisioning deploys the operating system into memory; therefore, the changes are not persistent across reboots. This method is faster because at provisioning time, the compute node does not spend time installing all of the packages. Instead, it uses a pre-generated stateless image on the management node. This pre-generated stateless image includes the operating system and related components.

3.2.4 Networking

Ethernet switches must be prepared before they are integrated into our cluster. For this process, the spanning-tree function and multicasting must be disabled in the switch. To speed up the process of PXE booting of compute nodes, the switch should begin forwarding the packets as it receives them. To ensure this, you should enable port-fast on the switch. If your switch does not support this feature, search for the specific function (command) that changes the forwarding scheme of your switch.

If you want to monitor the switch and add it to your pHPC monitoring portal, you must enable SNMP traps on the switch.

InfiniBand networks have many similarities to a traditional Ethernet network that is based on standard (traditional) network concepts, such as layers. In addition to these similarities, there are many differences between an InfiniBand network and an Ethernet network. The main difference is that InfiniBand provides a messaging-service that can be accessed directly by the applications.

Traditional networks are considered network-centric networks because they are mostly built with the focus on underlying switches and wires. In HPC, most of the applications need low latency for interprocess communications. This is why there is the need for a more application-centric network.

Building an application-centric network can be done by removing all unnecessary layers and by offloading the I/O operations from the CPU to the host bus adapter (HBA). This application-centric theory was transformed into a network architecture called InfiniBand. Therefore, the InfiniBand network architecture was a result of trying to determine how to make application's communication as simple, efficient, and direct as possible.

The InfiniBand network architecture provides to the applications an easy-to-use messaging-service. With the help of this messaging-service, the application can access the network directly, instead of relying on the operating system to transfer messages. Relying on the operating system to transfer the messages adds more overhead and, as a result, more latency. This latency is caused by the time that is needed for a message to pass from the application's virtual buffer space, through the network stack, and out into the wire. At the destination, there is an identical process in a reverse order that adds even more latency. InfiniBand avoids this through a technique that is known as *stack bypass*.

InfiniBand provides the messaging-service by creating a channel connecting applications that must communicate. The two applications can be in disjoint physical address spaces that are hosted by different servers. The channel links virtual address space of an application to the virtual address space of its peer application. The applications that are using the messaging-service can be user space applications or kernel space application (for example, the file system).

On each of the endpoints of the channel, there is a buffer to accumulate the network traffic. Therefore, any channel has a queue (buffer) at each end, and this forms a queue pair (QP). At each end, each queue has its own send and receive queue. Because the InfiniBand architecture requires avoiding the communication through the operating system and network stacks (as shown in Figure 3-3), the applications at each end of the channel are directly connected to the QP on their own host. This direct mapping of the applications to the QPs is done through linking the virtual address space of the application with the QP. The two applications at each endpoint of the channel can access directly the virtual address space of the peer application at the other end of the channel.

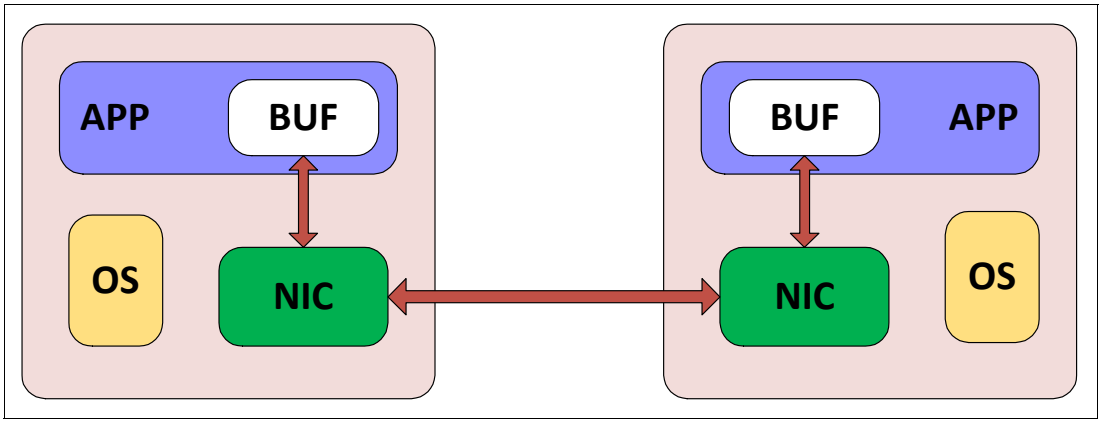

Figure 3-3 InfiniBand network communication

In essence, the infiniBand architecture creates communication channels between two disjoint virtual address spaces of peer applications by using QP, which provides a mean for local application to transfer messages directly without involving the local operating system (as shown in Figure 3-3).

Having established this communication channel, there is a need to implement methods for transferring the messages between the peers. The infiniBand architecture implemented the following transfer semantics:

- Channel semantic (SEND/RECEIVE): With this transfer method, each application accesses its own buffer only. The sending application SENDs the message and the receiving application RECEIVEs the message. When the message is received, the message is pre-posted on its receive queue.
- Memory-pair semantic (RDMA READ/WRITE): This is the opposite of the previous method. In this case, the receiving side application registers a buffer in its virtual memory space and then it passes the control of that buffer to the sending side, which uses RDMA READ or RDMA WRITE operations to either read or write the data in that buffer.

For more information about infiniBand architecture, see *HPC Clusters Using InfiniBand on IBM Power Systems Servers*, SG24-7767.

pHPC networking

In pHPC, you mostly manage logical networks while you are performing network management. In most of the cases, the logical networks overlap the physical networks, but not in every case. For example, in our scenario we chose to build two logical networks over one physical network: provisioning and out-of-band management (BMC).

You can define the following types of logical networks in pHPC:

- The provisioning network is used to connect the management nodes to the compute nodes, usually over a Gigabit network. The cluster performs its administration and monitoring functions over this network. For example, the DHCP service from the management node listens on the interface that is associated with this logical network and answers to the requests from the compute nodes. Moreover, all the PXE boot files and installation files are served over this network.
- The out-of-band management network (BMC) is used by the management nodes for remote hardware management of the compute nodes. The operations that are run over this network include: power control (on, off, reset, and soft-off); turn LEDs on and off; get hardware event logs or inventory information; collect sensor information, such as CPU temperature and fan speed; get energy usage, and serial console access.
- The public network is used to connect management nodes to the outside world, and to give users on the corporate network access to IBM Platform HPC web portal for cluster administration or job submission.
- The application (high-speed) network is used by applications to exchange data among its tasks that are in different nodes. In some cases, it can be a data path for applications to access the shared storage (for example, GPFS over RDMA).

In pHPC, a logical network is represented as a TCP/IP subnetwork. Such a network can be instantiated by specifying its name, an IP address range, and a gateway.

During the installation phase, the pHPC installer can configure by default a maximum of three types of networks: provisioning, out-of-band (BMC), and public. At a minimum, provisioning and out-of-band (BMC) networks must be configured. If you must configure other networks (for example, public or application), but not necessarily at installation time, you can configure this later through the pHPC web interface.

IBM Platform HPC can use two methods to add nodes to the cluster: the use of a host file or the use of the discovery feature. If you decide to use both, the provisioning subnetwork is divided into two subranges, one for each method. For example, if your provisioning subnetwork is `19.168.1.0/24`, the range that is used for nodes that are added to pHPC through a host file is `192.168.1.3` - `192.168.1.200`. The range for nodes that are added by the discovery method are `192.168.1.200` - `192.168.1.254`.

3.2.5 Cluster prerequisites

Before cluster deployment, the following prerequisites must be met:

- The primary management node must be installed by using traditional methods with a supported operating system; for example, Red Hat Enterprise Linux 6.4 (RHEL). For more information about checking whether your partitioning layout is correct, if you installed all of the necessary packages, verifying that you disabled all unnecessary services, and so on, see the “Preparing to install Platform HPC” chapter of the *IBM Platform HPC Installation Guide*, SC22-5380-00.
- During the installation, the pHPC installer must access an operating system media or image file (`.iso`) because the installer builds a first image profile (OSimage), which is used later to deploy the compute nodes with an operating system, as shown in Figure 3-4 on page 75.

Management Node	Compute Node	
	Certified	Supported
Red Hat Enterprise Linux (RHEL) 6.4 x86 64-bit	• RHEL 6.4 • CentOS 6.4 • RHEL 5.9	• RHEL 6.3 • RHEL 5.8 • CentOS 6.3 • CentOS 5.9 • CentOS 5.8

Figure 3-4 Management nodes and compute nodes operating systems

- All servers must boot over the network. You can check this by using the IBM Advanced Settings Utility (ASU) if your servers have this capability. The setting `BootOrder.BootOrder` contains the boot order and the value of “PXE Network” should be set ahead of any “Hard Disk” value in the ordered list. The tool can be downloaded from this website:

  `http://www-947.ibm.com/support/entry/portal/docdisplay?lndocid=TOOL-ASU`
- All servers must have an IP address assigned to their IMM to control the BIOS/uEFI firmware remotely via the command line or the ASU. This tool can be used in later stages of your cluster deployment when you conduct performance tests because from a central point, you can tweak different parameters into the firmware to boost the cluster performance.
- To access the IBM Platform HPC web portal after installation, you need the following programs:
  - Adobe Flash version 8 or higher and JRE 1.6 version or higher
  - IE 8 or 9 or Firefox 21.x or higher on Windows and Firefox 10.x or higher on Linux
- For a comprehensive list of the minimal hardware and software requirements for the management nodes and compute nodes, see *IBM Platform HPC Installation Guide*, SC22-5380-00.

Note: Our cluster was built for demonstration purposes. If you plan to deploy a production cluster, make proper plans according to your environment requirements. Before you proceed with the installation, see *IBM Platform HPC Installation Guide*, SC22-5380-00.

3.2.6 Cluster deployment

Now that our cluster is defined and all the prerequisites are fulfilled, we can proceed with the cluster deployment. We start to build the cluster by installing the primary management node with the IBM Platform HPC. Before you start the installation script, remember to check all of the requirements for the management node as described in the “Preparing to install Platform HPC” chapter of *IBM Platform HPC Installation Guide*, SC22-5380-00.

In the following sections, you can follow the details of our cluster installation for reference, as shown in Example 3-1 on page 76.

Example 3-1 IBM Platform HPC installation

```
[root@i04n50 install]# ./phpc-installer.sh
   Preparing to install 'phpc-installer'...                [  OK  ]
******************** WARNING ********************
A partial Platform HPC installation is detected.
Continuing will remove the existing installation.

Do you want to continue? (Y/N) [Y] Y
   Checking for any packages                               [  OK  ]
   Cleaning database...                                    [  OK  ]
   Finding the product entitlement file...                 [  OK  ]
   ================================================================
   Welcome to the IBM Platform HPC 4.1.1.1 Installation
   ================================================================
   The complete IBM Platform HPC 4.1.1.1 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.

   ================================================================
   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.
1

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

   Checking hardware architecture...                       [  OK  ]
   Checking OS compatibility...                            [  OK  ]
   Checking free memory...                                 [  OK  ]
   Checking if SELinux is disabled...                      [  OK  ]
   Checking if Auto Update is disabled...                  [  OK  ]
   Checking if NetworkManager is disabled...               [  OK  ]
   Checking if PostgreSQL is disabled...                   [  OK  ]
   Checking for DNS service...                             [  OK  ]
   Checking for DHCP service...                            [  OK  ]
   Checking management node name...                        [  OK  ]
   Checking static NIC...                                  [  OK  ]
```

```
Probing DNS settings...                                  [  OK  ]
Probing language and locale settings...                  [  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...   [  OK  ]

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

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:  /tmp/rhel6.4_x64.iso

Select a network interface for the provisioning network from the following options:
1) Interface: eth0, IP: 129.40.64.50, Netmask: 255.255.255.0
Enter your selection [1]:  1

Enter IP address range used for provisioning compute nodes
[129.40.64.3-129.40.64.200]: 129.40.64.51-129.40.64.92

Do you want to provision compute nodes with node discovery? (Y/N) [Y]: 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. [129.40.64.201-129.40.64.254]: 129.40.64.230-129.40.64.254

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

Select a network to use as your BMC network
1) Create a new network
2) Provision network
Enter your selection [1]:  1

Specify a subnet for the BMC network [xxx.xxx.xxx.xxx]: 129.40.65.0
Specify a subnet mask for the BMC network [255.255.255.0]: 255.255.255.0
Specify a gateway IP address for the BMC network [None]: 129.40.65.254
Specify an IP address range for the BMC network
[129.40.65.3-129.40.65.254]: 129.40.65.51-129.40.65.92

Specify a hardware profile for the BMC network:
1) IBM_System_x_M4  IBM System x3550 M4, x3650 M4, x3750 M4
2) IBM_Flex_System_x  IBM Flex System x220, x240, and x440
3) IBM_iDataPlex_M4  IBM System dx360 M4
4) IPMI  Any hardware with IPMI device
     5) IBM_NeXtScale_M4  IBM System nx360 M4
Enter your selection [1]:  4

Enter a domain name for the provisioning network [private.dns.zone]:  pbm.ihost.com
Enter the IP addresses of extra name servers that are separated by commas
[129.40.106.1]: 129.40.106.1
```

```
Enter NTP server [pool.ntp.org]:  i04mgr.pbm.ihost.com

Synchronizing management node with the time server...   [  OK  ]

Do you want to export the home directory on the management node
and use it for all compute nodes? (Y/N) [Y]: Y

Do you want to change the root password for compute nodes and the
default password for the Platform HPC database? (Y/N) [Y]: N

================================================================
 Platform HPC Installation Summary
================================================================
You have selected the following installation settings:

Provision network domain:               pbm.ihost.com
Provision network interface:            eth0, 129.40.64.0/255.255.255.0
Depot (/install) directory mount point: /install
OS media:                               /tmp/rhel6.4_x64.iso
Network Interface:                      eth0
eth0 IP address range for compute nodes: 129.40.64.51-129.40.64.92
eth0 IP address range for node discovery:129.40.64.230-129.40.64.254
NTP server:                             i04mgr.pbm.ihost.com
Name servers:                           129.40.106.1
Database administrator password:        ************
Compute node root password:             ************
Export home directory:                  Yes
Enable default BMC network with:        129.40.65.0/255.255.255.0
Default BMC network Gateway:            129.40.65.254
Default BMC network IP range:           129.40.65.51-129.40.65.92
Default BMC Hardware Profile:           IPMI

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

Copying Platform HPC core packages...                  [OK]
Copying Platform HPC kits...                           [ OK ]
Adding OS from media '/mnt3'...
* Verifying that the OS distribution, architecture,
  and version are supported...                         [  OK  ]
* Detected OS: [rhel 6 x86_64]                         [  OK  ]
* Copying OS media. This can take a few minutes...     [\]
Successfully added operating system.                   [  OK  ]
Preparing the bootstrap environment...                 [  OK  ]
Installing packages. This can take several minutes...  [/]
                                                       [  OK  ]
Initializing Platform HPC 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 dhcpd:                                       [  OK  ]
   Setting up named:                                       [  OK  ]

   Setting up shared NFS export:                           [  OK  ]
   Setting up ntpd:                                        [  OK  ]
   Setting up LSF configuration:                           [  OK  ]
   Setting up web portal:                                  [  OK  ]
                                                           [  OK  ]
   Setting up Platform HPC configuration:                  [  OK  ]
   Updating management node:                               [  OK  ]

Platform Initalization Complete

IBM System X iDataPlex dx360 M3 Server
UEFI Build Ver: 1.19    IMM Build Ver: 1.35   Diagnostics Build Ver: 9.27

2 CPU Packages Available at 6.4GT/s Link Speed             [  OK  ]
24576MB Memory Available at 1333MHz in Independent Channel Mode
SLOT ID  LUN VENDOR   PRODUCT          REVISION   INT13 SIZE \ NV
---- --- --- -------- ---------------- ---------- ----- ---------r.log.
  0    1  0  IBM-ESXS VPBA146C3ETS11 N A496        Boot    136 GB

LSI Corporation MPT boot ROM successfully installed your web browser, you can access the
Web Portal at http://129.40.64.50:8080.

Enter 'q' to quit or close this terminal.
source /opt/pcm/bin/pcmenv.sh
Enter 'q' to quit or close this terminal.
q
[root@i04n50 install]#
```

After the installation, you can connect to the web portal of your first management node by using the link at the bottom of the output of the installation script. In our case, the following link is used:

http://129.40.64.50:8080

The dashboard is a panel that displays the health of your cluster (as shown in Figure 3-5 on page 80) and the resources available to your cluster (CPU and cores, RAM, and storage space) or in some cases, which resources are unavailable, by raising alarms.

Also displayed in the panel are Cluster performance metrics, such as CPU and memory load average, compute node availability over time, job slot usage, or power consumption. Spend time in this panel to get familiar with all the metrics that can help you understanding how efficiently you use your cluster resources, or if there is any issue with some components of the cluster.

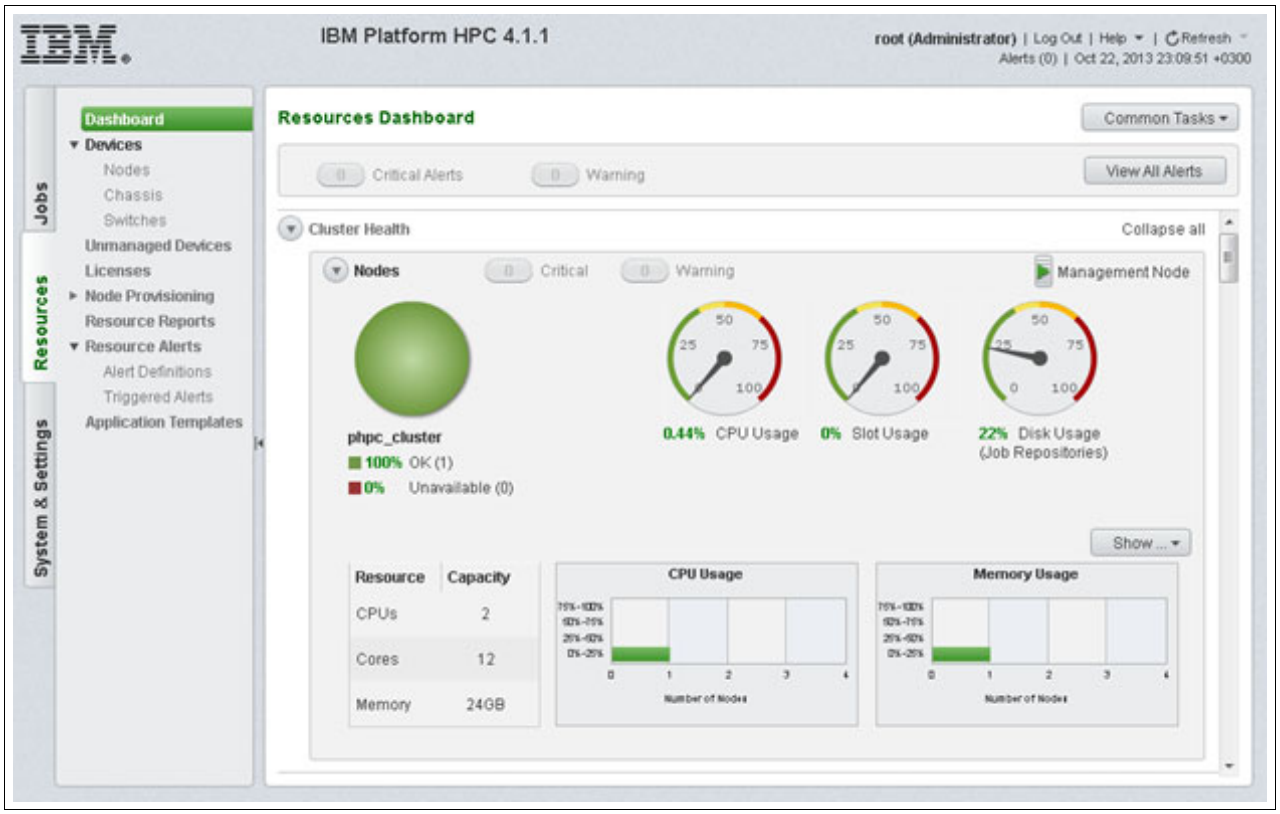

Figure 3-5 IBM Platform HPC 4.1.1 Resource Dashboard

A third main panel in this dashboard is a Rack view, which by default is provisioned with four 42U racks that can be populated with your cluster nodes or chassis. You can increase or decrease the number of the available racks, add or remove chassis, or change their position in the racks only by command line by using the commands **`pcmrackexport`** and **`pcmrackimport`**.

For example, to add a chassis into one of your racks, one option is to export the current configuration into a file by using the **`pcmrackexport`** command, adding the details of your chassis to this file, and then importing it into your cluster by using the **`pcmrackimport`** command. For more information about adding a chassis, see 3.2.7, “Provisioning cluster resources” on page 80.

The compute nodes can be provisioned into a Rack view at an unspecified or specific location. When a node is added to the cluster, if you do not specify any location, it is placed into a pool named “location unspecified”, which is outside any defined racks. Later on, you can add this node to a free location in any of the available racks by updating the node’s location through the GUI (Relocate) or the CLI.

3.2.7 Provisioning cluster resources

Now that we explored the dashboard, we start provisioning our cluster resources. The resource is a generic term that can refer to devices (compute nodes, switches, chassis), operating systems distributions, software (kits), licenses, and networks.

Devices can be of two types: managed and unmanaged.

Unmanaged devices are devices that are known by the pHPC cluster but are not provisioned, updated, or monitored by the cluster. Defining a node as unmanaged in the HPC cluster disables the node from being allocated by the provisioning engine. An example of such device can be an NAS device or NFS server, which exists in your network and exports resources to your cluster but it is not managed by pHPC.

Managed devices are devices on which the pHPC cluster can perform actions, such as provisioning, updating, monitoring, power on and off actions, inventory, and many others. There are three types of managed devices that you can define and manage: nodes, chassis, and switches.

To add node devices, you must complete some preliminary preparation tasks. For now, we add only switches and chassis devices; after all the necessary elements are defined later, we describe you how to add node devices.

Adding chassis

Adding a chassis to the cluster can be done only through the command-line interface (CLI). As a chassis is always placed into a rack, we add this device to the cluster by using rack commands. To add it, you must collect all of the parameters by which this chassis can be identified and create a chassis definition to be imported into the pHPC database. After the import finishes, you must restart the GUI interface to reflect the changes you made to the pHPC database. After the GUI restarts, you should see your chassis by clicking **Devices** → **Chassis** and under **Dashboard** → **Rack View**.

Example 3-2 shows how to add a chassis to your cluster.

Example 3-2 Add chassis to your pHPC cluster

```
[root@i04n50 ~]# pcmrackexport > rack.info
[root@i04n50 ~]# vi rack.info
Add your chassis definition parameters to the end of your file (rack.info). For
our chassis the definition is:

chassis3:
       objtype=chassis
       displayname=nextscale1
       height=6
       ip=129.40.65.80
       username=USERID
       password=PASSWORD
       rack=rack1
       slots=16
       unit=37
       urlpath=http://129.40.65.80
[root@i04n50 ~]# pcmrackimport -f rack.info
```

Note: To reflect these changes in your user interface (Dashboard), restart the GUI by running the **`plcclient.sh -d pcmnodeloader`** command.

For our scenario, there is no need to add a chassis because we do not have a chassis. After you add switches to your cluster definition, you can monitor and collect performance statistics by using SNMP traps.

Adding a node device

Every node you add to your cluster is unique in some aspects (MAC/IP address, name, location into the rack, and so on), and it has many similarities with other cluster nodes. For example, your cluster can be built on two different hardware types with different processing capabilities and different network adapter layout. As a consequence, you install different operating systems and applications on each node type. Therefore, you split your hardware pool in two main categories, each category having compute nodes with the same capabilities. This means that you profile the compute nodes.

For each of these profiles, you can create a template that can help provision each type of compute nodes. The provisioning template has predefined components that are determined by hardware type and network and software requirements, as shown in Figure 3-6. Further, for each of these predefined components, we use the terms hardware profile, network profile, and image profile.

Note: You can profile the nodes with other criteria, not only with the ones that are used for exemplification.

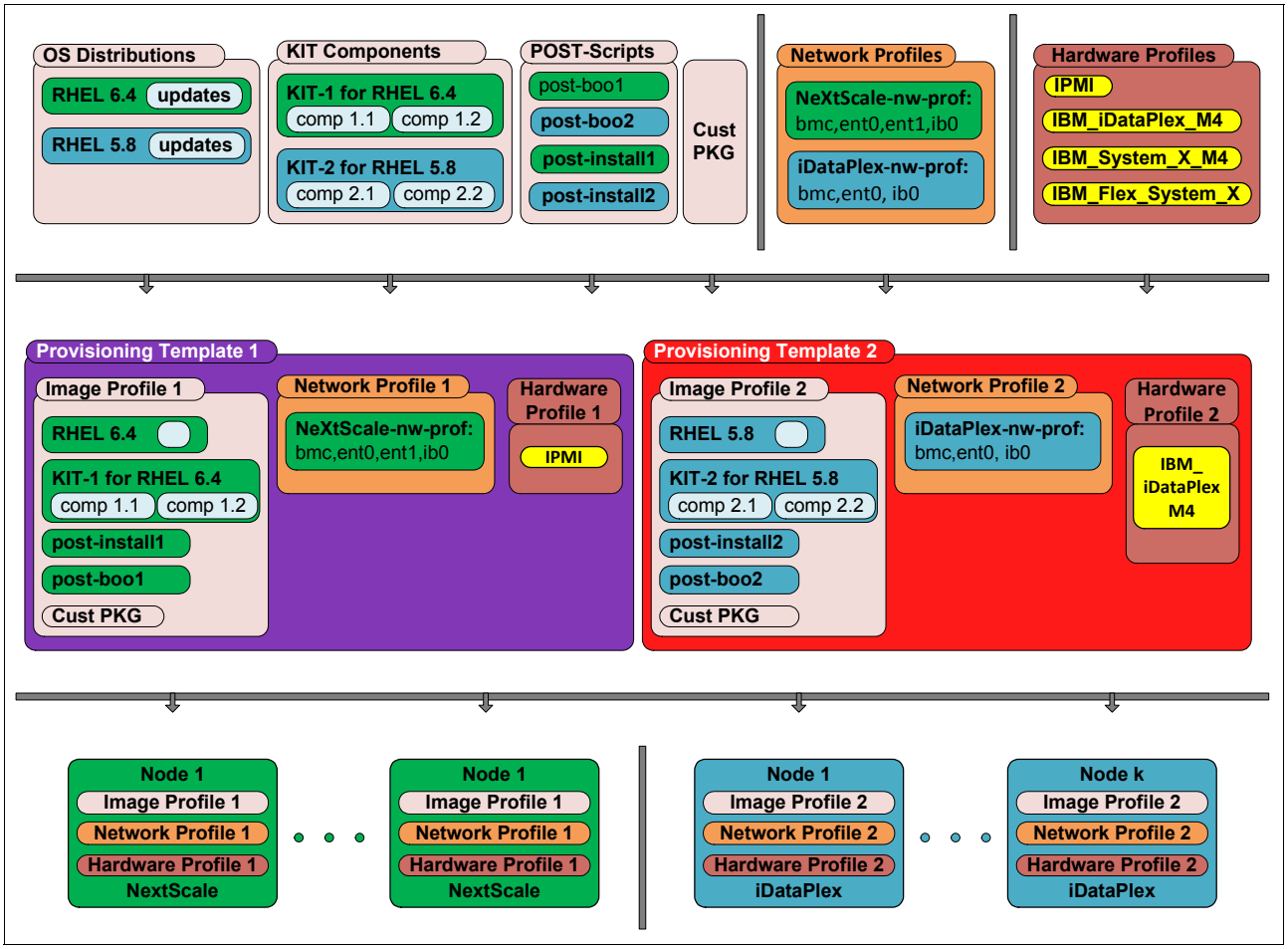

Figure 3-6 Provisioning template

As shown in Figure 3-6, each node at the end assigned an image profile, a network profile, and a hardware profile, which are part of node definition. The profiles cannot be changed without reprovisioning the node.

To add a node, it is mandatory to have all of the components (image, network, and hardware profiles) defined and functional, as shown in Figure 3-7. By default, the pHPC installer creates a few predefined profiles that are based on the information that was provided at installation time. For example, you have an image profile built that is based on an `.iso` image that was provided during the pHPC installation with the pHPC default kit-components, a network profile built on the networks provided (at minimum the provisioning network), and a predefined hardware profile.

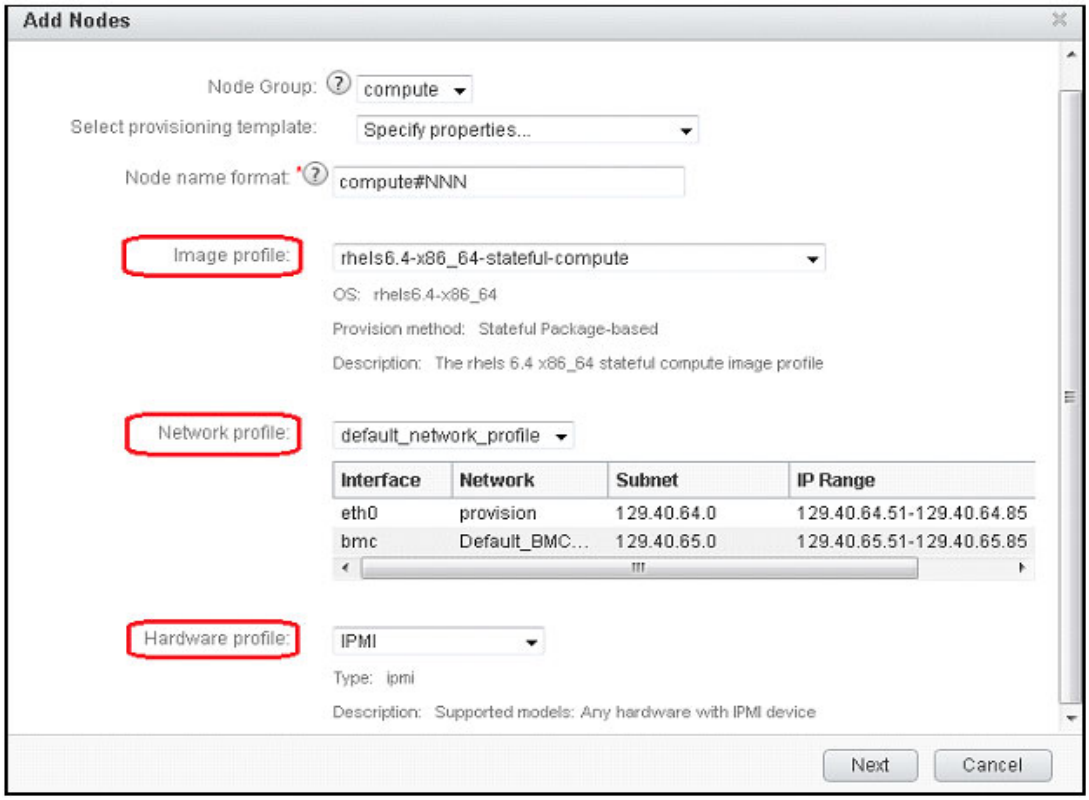

Figure 3-7 Profile for adding a note

As shown in Figure 3-7, you can choose from a drop-down menu the different profiles or use the defaults. Because these profiles are common to many nodes, pHPC can create out of them a provisioning template by linking all of these profiles for later node add processes or for node provisioning. A second option is to add nodes by selecting a predefined (or user-created) provisioning template (see the submenu Select provisioning template in Figure 3-7), and not selecting each time all the component profiles. This is the default method that is used when nodes are added.

The adding nodes wizard can provision more than one node at a time. For that process, you are requested to introduce the Node name format that is used to automatically name the nodes. Nodes are managed devices represented by compute nodes, which can be added to the cluster by two methods: Discovery feature or host file (node information) method.

Adding nodes to the cluster by using the host file method allows you to add nodes with a specific host name, IP address for management, and IP address for BMC, MAC address, the position in rack, and how many units it occupies in the rack.

Now that we defined all the common components (profiles) of our nodes, we can add the details for each individual node. In the next window of the wizard, you are prompted to choose between two methods of node discovery: Auto discovery by PXE boot or import node information file.

The unique details of each node can be obtained by using one of the discovery methods. For the import node information file method, preconfigured hardware is delivered with a file that contains all of the details about each compute node (including rack details). If your hardware is not preconfigured but you know all of the details of your nodes, add them into a file and import these details by using the wizard. The file should have the format as shown in Example 3-3.

Example 3-3 Node definition example

```
<hostname>:
   mac=<mac-address>
   ip=<IP for primary NIC>
   nicips=<IPs for other NICs>
   rack=<rack name>
   height=<server height>
   unit=<server's position in rack>
```

Note: If the NICs for the other networks (for example, `bmc` and `eth1`) are not defined in the node information file, the NICs are assigned auto-generated IP addresses. The IP address of the NIC that is associated with the BMC must be assigned accordingly on the nodes for remote power management to function correctly.

This method should be used when you want to assign static host names and IP addresses to your compute nodes.

The auto discovery by PXE boot method (as shown in Figure 3-8) is used most often to add nodes to the cluster when you do not know the details of your compute nodes and the name and IP address that your compute node is assigned is not important. However, you can control the name and IP address assignment by powering on the nodes individually in the order that you want. Your nodes have names and IP addressed that are assigned in a sequential order.

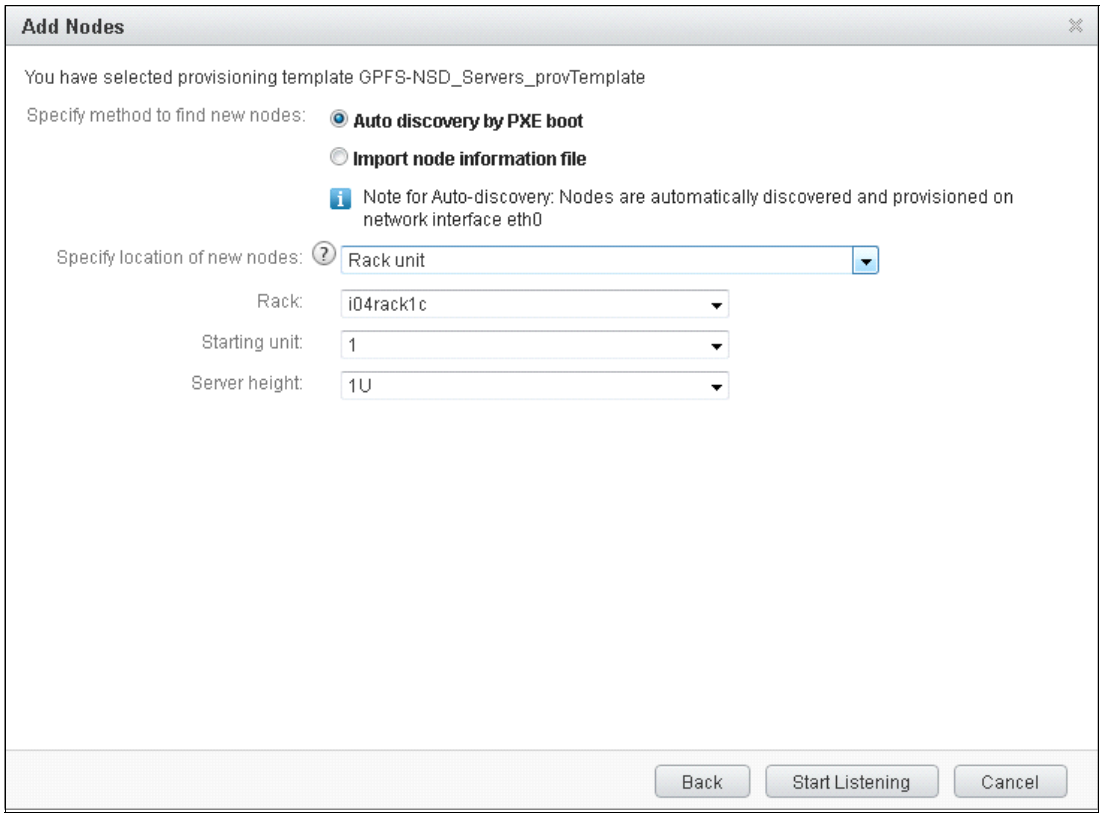

Figure 3-8 Provisioning template GPFS-NSD_Servers_provTemplate

Note: To use this provisioning method, you must define a discovery IP address range into the provisioning network at installation time. This discovery IP address range is used to assign a temporary IP address for collecting information about the host by using the auto node discovery method.

To define the cluster for our scenario, we used a host information file. As shown in Example 3-4, the first two compute nodes are used as GPFS server nodes (NSD servers) to share the disks to the cluster. We prefer to provision the GPFS nodes by using pHPC because all internal communication of GPFS relies on SSH, and it must exchange keys with all GPFS clients, which is automatically done by pHPC. Because the IP addresses of the NIC that is associated with the BMC of the nodes are configured, they are explicitly defined in the host-information-file. The nodes can have non-consecutive static IP addresses, as shown in Example 3-11 on page 91.

Example 3-4 Adding nodes by using host information file method

```
i04n51-gpfs1:
mac=E4:1F:13:84:40:F8
ip=129.40.64.51
nicips=bmc!129.40.65.51
i04n52-gpfs2:
mac=E4:1F:13:84:46:8C
ip=129.40.64.52
nicips=bmc!129.40.65.52
i04n53:
mac=E4:1F:13:4D:87:D8
ip=129.40.64.53
nicips=bmc!129.40.65.53
i04n54:
mac=E4:1F:13:4D:90:00
ip=129.40.64.54
nicips=bmc!129.40.65.54
i04n63:
mac=E4:1F:13:84:31:98
ip=129.40.64.63
nicips=bmc!129.40.65.63
i04n64:
mac=E4:1F:13:84:41:88
ip=129.40.64.64
nicips=bmc!129.40.65.64
```

You can start adding nodes to the cluster immediately after the pHPC installation by using the default provisioning template and default profiles, which is enough if you want to evaluate the product. In production, you must define your own provisioning template and all of its constituent components because you must add your other software, provision your custom networks, or modify hardware profiles according to your hardware.

We define all of the components of a provisioning template as is shown Figure 3-6 on page 82 from left to right and top to bottom.

The operating system (OS) distribution is a copy of the Linux distribution and service levels that are obtained from the operating system ISO files. OS distributions are packages that are used to distribute operating system software onto nodes. At creation time, each OS distribution is copied to a default destination directory under `/install/OS-name/OS-arch`, where OS-name is the distribution name and OS-arch is the OS architecture; for example: `/install/rhels6.4/x86_64`.

Platform HPC can create OS updates for an OS distribution that exists in the system. OS updates are stored in the `/install/osdistroupdates` directory and can be applied to compute nodes. For example, after the OS update is created, it is stored in the `/install/osdistroupdates/OS-update-name` directory. For provisioned compute nodes, the OS update can change the system configuration files or upgrade some software packages to a higher version.

Complete the following steps to add OS updates from a local directory:

1. Determine the name of the OS distribution you want to update, as shown in Example 3-5.

Example 3-5 Operating system distribution to be updated

```
[root@i04n50 ~]# lsdef -t osdistro
rhels6.4-x86_64  (osdistro)
[root@i04n50 ~]#
```

2. Copy the OS updates (Rational Portfolio Manager packages) to a directory (for example, `/tmp/updates`).
3. Create an OS update definition, as shown in Example 3-6.

Example 3-6 Creating an OS distribution update

```
[root@i04n50 ~]# pcmosdistroupdate -c rhels6.4-x86_64 -p /tmp/updates
Creating the OS distribution update for rhels6.4-x86_64 from /tmp/updates
Created OS distribution update rhels6.4-x86_64-2013-11-14_10-06.
[root@i04n50 ~]#
```

4. List the new OS update, as shown in Example 3-7.

Example 3-7 Listing the OS update

```
[root@i04n50 ~]# pcmosdistroupdate -l rhels6.4-x86_64
osdistroupdate name: rhels6.4-x86_64-2013-11-14_10-06
    osdistroname=rhels6.4-x86_64
    dirpath=/install/osdistroupdates/rhels6.4-x86_64-2013-11-14_10-06
    downloadtime=2013-11-14_10-6
    disable=0
[root@i04n50 ~]#
```

5. For the OS update to be accessible from the Web Portal, you must trigger the PERF loader. To trigger the PERF loader to update the database, run the command as shown in Example 3-8.

Example 3-8 Loaders start

```
[root@i04n50 ~]# plcclient.sh -d "pcmosdistroloader"
Loaders startup successfully.
[root@i04n50 ~]#
```

Cross Distribution cluster

Users often install homogeneous clusters where management nodes and compute nodes use the same OS Distribution. In some cases, the compute nodes can be different, so you can mix OS distributions in the same cluster. This is a Cross Distribution cluster.

To add a new OS Distribution, use the GUI (as shown in Figure 3-9 on page 88) and an ISO file. When a new OS Distribution is defined, two image profiles also are created: one stateful and one stateless. The image profiles are stored in the `/install/osimages/*` directory. For more information about an image profile, see “Image profiles” on page 102.

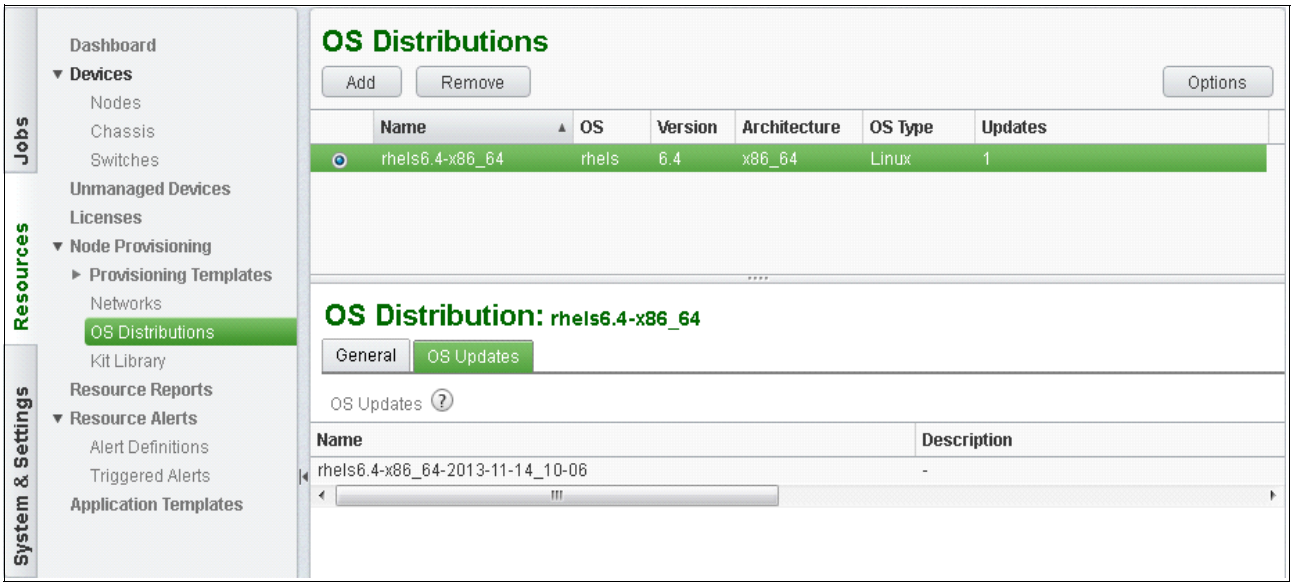

Figure 3-9 OS distributions window

Kit and kit components

In the IBM Platform HPC web interface, you see a menu that is named Kit Library, as shown in Figure 3-10.

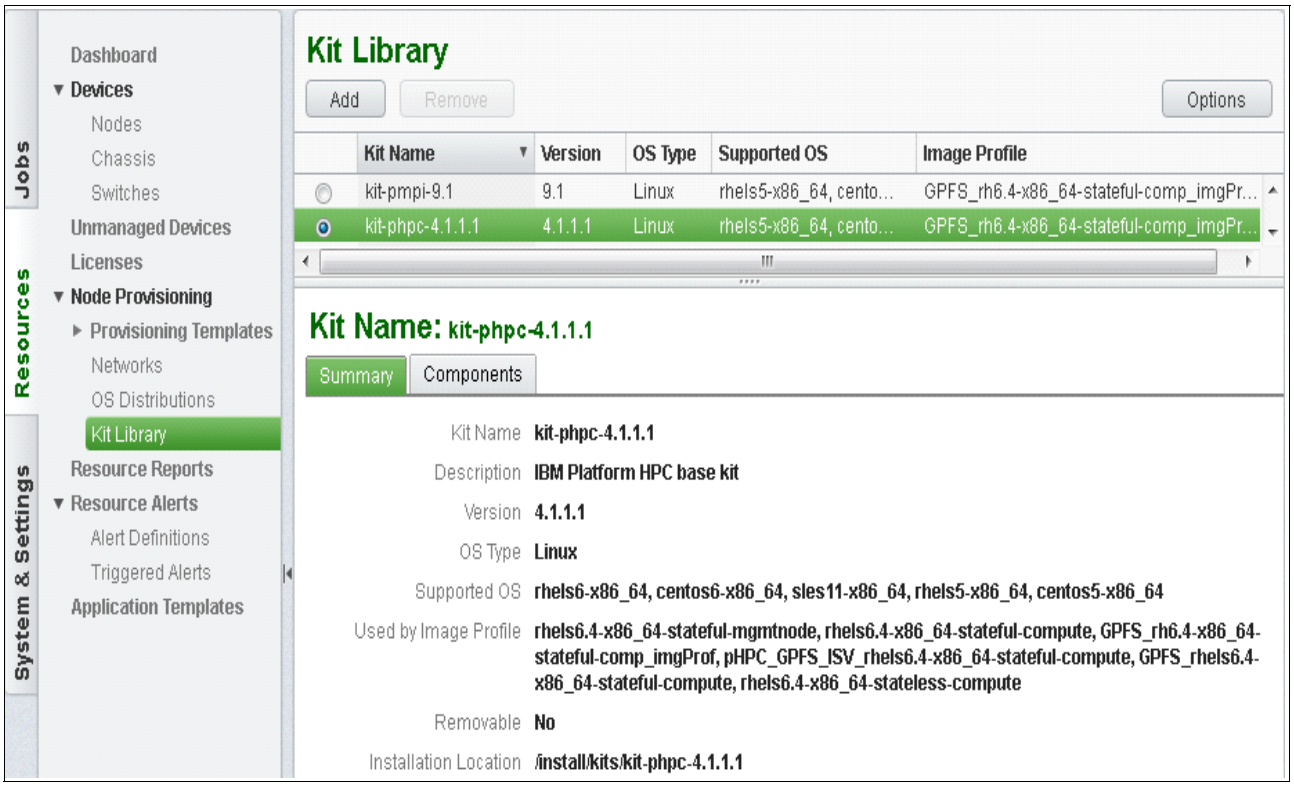

Figure 3-10 IBM Platform HPC Kit Library menu

A Kit Library is a central repository in which you can store kits. These kits can be applied to different hardware architectures (for example, x86 or x64) or different OS distributions (for example, RHEL or SLES).

A *kit* is a pre-packed bundle that contains an application and its related components. A kit represents a mechanism to bundle an application with its installation scripts to make it easier to manage and deploy your cluster. Kits are used to deploy a single application on compute nodes. IBM Platform HPC ships with kits, such as Platform LSF, Platform MPI, and Platform PAC.

A *kit component* represents different parts of an application to be deployed into your cluster. You can have a client component and server component of an application bundled into the same kit. Platform LSF is a practical example; it is bundled into a kit that contains two kit components: Master component (installed on management nodes) and compute component (installed on compute nodes).

Kit components are used to install Platform products, but also can be used by users to install their own applications on the compute nodes. Within each kit component, a user can define which packages are installed and which scripts are run to configure the component.

If the kit is deployed on different hardware architectures (for example, x86 or x64) or different OS distributions (for example, RHEL or SLES), you must define a kit component that is installed only on that OS distribution for each architecture or each OS distribution.

A kit is an archive file (`.tar`) that contains the following components:

- Kit repositories: A directory for each operating system version in which this kit is supported. Each directory contains all of the product software packages that are required for that environment with repository metadata.
- Kit components: A metapackage that is built to ask for all product software dependencies and to automatically run installation and configuration scripts. It relies on the packages that are stored into kit repositories or OS distribution repositories.
- Kit component files: Scripts, deployment parameters, exclusion lists, and other files that are used to install and configure the kit components and product packages.
- Kit configuration file: A file that describes the contents of the kit that contains the following information:
  - Kit name, version, description, supported OS distributions, license information, and deployment parameters.
  - Kit repository information, including name, supported OS distributions, and supported architectures.
  - Kit component information, including name, version, description, server roles, scripts, and other data.
- Documentation: Product documentation that is shipped as HTML files that can be displayed by using the GUI.
- Plug-ins: Used for other product configuration and customization during image management and node management.

To build a kit, you must use the tool that is shipped with pHPC that is called **`buildkit`**. This tool is used to generate a skeleton (template) for your new kit. It then compiles your modifications that were made into the skeleton and bundles all of your components into a kit. The following steps are used:

1. Building the skeleton of the kit is the process that is used to create a predefined subdirectories tree that are automatically populated with samples in which you can add your packages, scripts, parameters, and so on, as shown in Example 3-9 on page 90. In the next sections, we describe each subdirectory and file.

Example 3-9 Command syntax for buildkit

```
# buildkit create <kit_basename>
```

The command that is shown in Example 3-9 creates a subdirectory called `kit_basename` in your current directory, as shown in Figure 3-11.

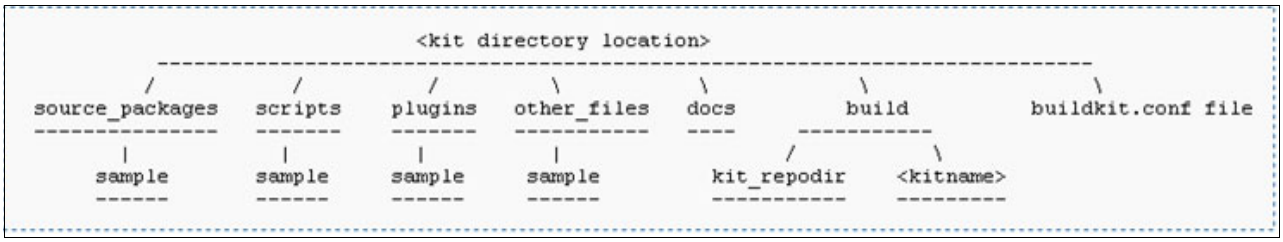

Figure 3-11 Kit directory location

The subdirectory contains the following components:

- `<kit directory location>`: The full path name of the location that is specified on the command line or the current working directory in which you ran the **`buildkit`** command and the `<kit_basename>` that you provided.
- `buildkit.conf`: The sample kit build file.
- source_packages: This directory stores the source packages for kit packages and non-native packages. The **`buildkit`** CLI searches these directories for source packages when packages are built. This directory stores RPM spec and tarballs, source RPMs, pre-built RPMs, and non-native packages.
- scripts: Stores the kit deployment scripts.
- plug-ins: Stores the kit plug-ins. Samples are provided for each type of plug-in.
- docs: Stores the kit documentation files.
- other_files:
  - `kitdeployparams.lst`: The kit deployment parameters file.
  - `exclude.lst`: Contains files and directories to exclude from the stateless image.
- build: Stores files when the kit is built. For example, the kit `.tar` file `kitname.tar.bz2`. This directory is created only after you create the following repositories (**`buildkit buildrepo all`**):
  - `build/kit_repodir` stores the fully built kit package repositories
  - `build/<kitbasename>` stores the contents of the kit `.tar` file before it is archived in a `.tar` file

2. Modify the skeleton by adding your components. This is the phase in which you prepare your kit with all its components before it is bundled.

   In our cluster scenario, we must install GPFS on two types of nodes: GPFS server nodes and GPFS client nodes. Our GPFS rpm files are of two types, as shown in Example 3-10.

Example 3-10 GPFS base release

```
[root@i04n50 gpfs.base]# ll
-rw-r--r-- 1 root root 12405953 Oct 25 21:05 gpfs.base-3.5.0-3.x86_64.rpm
-rw-r--r-- 1 root root   230114 Oct 25 21:05 gpfs.docs-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root   509477 Oct 25 21:05 gpfs.gpl-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root    99638 Oct 25 21:05 gpfs.msg.en_US-3.5.0-3.noarch.rpm
[root@i04n50 gpfs.base]#
```

As shown in Example 3-11, in the `gpfs_updates` directory, there is a file named `gpfs.gplbin-*`. This file is the GPFS portability layer.

Example 3-11 GPFS updates

```
[root@i04n50 gpfs.update]# ll
-rw-r--r-- 1 root root 13156422 Oct 25 21:13 gpfs.base-3.5.0-13.x86_64.update.rpm
-rw-r--r-- 1 root root   254205 Oct 25 21:13 gpfs.docs-3.5.0-13.noarch.rpm
-rw-r--r-- 1 root root   554372 Oct 25 21:13 gpfs.gpl-3.5.0-13.noarch.rpm
-rw-r--r-- 1 root root  1866576 Oct 25 21:13
gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
-rw-r--r-- 1 root root   106817 Oct 25 21:13 gpfs.msg.en_US-3.5.0-13.noarch.rpm
[root@i04n50 gpfs.update]#
```

GPFS portability layer and how to build it

On Linux platforms, GPFS uses a loadable kernel module that enables the GPFS daemon to interact with the Linux kernel. This kernel module is specific to each version of Linux and it must be built from the source code for each type of Linux. When GPFS is installed on Linux, you must build the portability layer (kernel module) that is based on your particular hardware platform and Linux distribution to enable communication between the Linux kernel and GPFS.

To make the deployment simpler to all compute nodes, after compiling the source code on a template node, you can build an rpm package (binary module) and reuse this rpm on all compute nodes that have the same hardware and Linux distribution.

Complete the following steps to build the portability layer:

1. Install all of the prerequisites, as shown in Example 3-12.

   Example 3-12 Installing prerequisites

   ```
   [root@i04n50]# yum install kernel-devel kernel-headers xorg-x11-xauth gcc-c++
   imake libXp compat-libstdc++-33 compat-libstdc++-296 libstdc++
   ```

2. Compile the GPFS portability layer for the current environment, as shown in Example 3-13.

   Example 3-13 Compiling to create the portability layer

   ```
   [root@i04n50 /usr/lpp/mmfs/src]# make Autoconfig
   [root@i04n50 /usr/lpp/mmfs/src]# make World
   [root@i04n50 /usr/lpp/mmfs/src]# make InstallImages
   ```

   Note: Building the portability layer assumes the GPFS main binaries are installed.

3. Build the RPM, as shown in Example 3-14.

   Example 3-14 Building the rpm

   ```
   [root@i04n50 /usr/lpp/mmfs/src]# make rpm
   ... (output truncated)
   Wrote:
   /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
   ... (output truncated)
   [root@i04n50 /usr/lpp/mmfs/src]#
   ```

You cannot have in the same kit more than one component that refers to the same package name but different rpm name. For example, if you define two components into the same kit (one for base and one for updates) and make a dependency for updates to depend on the base, you might end up installing updates first and then base, which fails. In Example 3-15, you can see that the package name is different and the name inside is the same. This is correct because when you perform an update, the rpm tool knows exactly what to update.

Example 3-15 Running an update

```
[root@i04n50 gpfs.update]# rpm -qip gpfs.base-3.5.0-13.x86_64.update.rpm
Name        : gpfs.base                    Relocations: (not relocatable)
Version     : 3.5.0                        Vendor: IBM Corp.
Release     : 13                       Build Date: Mon 30 Sep 2013 03:51:14 PM EDT
Install Date: (not installed)          Build Host:
bldlnxbc2b11.pok.stglabs.ibm.com
Group       : System Environment/Base       Source RPM: gpfs.base-3.5.0-13.src.rpm
Size        : 38338955                         License: (C) COPYRIGHT
International
Signature   : (none)
Packager    : IBM Corp. <gpfs@us.ibm.com>
URL         : http://www.ibm.com/systems/software/gpfs/index.html
Summary     : General Parallel File System
[root@i04n50 gpfs.update]#


[root@i04n50 gpfs.base]# rpm -qip gpfs.base-3.5.0-3.x86_64.rpm
Name        : gpfs.base                Relocations: (not relocatable)
Version     : 3.5.0                         Vendor: IBM Corp.
Release     : 3                          Build Date: Mon 20 Aug 2012 03:57:46 PM
EDT
Install Date: (not installed)           Build Host: bldlnxbc2b11.ppd.pok.ibm.com
Group       : System Environment/Base   Source RPM: gpfs.base-3.5.0-3.src.rpm
Size        : 36313403 License: (C) COPYRIGHT International
Signature   : (none)
Packager    : IBM Corp. <gpfs@us.ibm.com>
URL         : http://www.ibm.com/servers/eserver/clusters/software/gpfs.html
Summary     : General Parallel File System
[root@i04n50 gpfs.update]#
```

In building the GPFS kits for our cluster, we chose to build two kits. The first kit is the base kit and it is installed on all nodes (server and client nodes) in the same manner, and a second kit for updates. In the updates kit, we added two kit-components, one for updating the GPFS server nodes and the second for updating GPFS client nodes. The difference between them is that for the component of the GPFS client nodes (compute nodes), we added a post script that has the role of adding the node as a GPFS client.

In the next sections, you can find the details of both kits that we built. We start with the GPFS base kit and then continue with the GPFS updates kit. For the GPFS base kit, we added to the skeleton all the rpm files that we want to have into the kit (see `/source_packages/gpfs.base`, as shown in Example 3-16 on page 93).

Example 3-16 Source rpm files

```
[root@i04n50 gpfs.base]# ls –ltR
-rw-r--r-- 1 root root     1257 Oct 27 22:26 buildkit.conf
drwxr-xr-x 2 root root     4096 Oct 22 15:40 docs
drwxr-xr-x 3 root root     4096 Oct 22 15:40 other_files
drwxr-xr-x 8 root root     4096 Oct 29 14:38 plugins
drwxr-xr-x 3 root root     4096 Oct 22 15:40 scripts
drwxr-xr-x 4 root root     4096 Oct 25 21:18 source_packages
. . . (truncated)
./source_packages/gpfs.base:
-rw-r--r-- 1 root root 12405953 Oct 25 21:05 gpfs.base-3.5.0-3.x86_64.rpm
-rw-r--r-- 1 root root   230114 Oct 25 21:05 gpfs.docs-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root   509477 Oct 25 21:05 gpfs.gpl-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root    99638 Oct 25 21:05 gpfs.msg.en_US-3.5.0-3.noarch.rpm
[root@i04n50 gpfs.base]#
```

The skeleton includes a template configuration file that must be modified to reflect your needs. (In the documentation, you find in the `buildkit.conf` file.) For our GPFS base kit, Example 3-17 shows the kit configuration file that we used.

Example 3-17 buildkit.conf file

```
[root@i04n50 gpfs.base]# cat buildkit.conf
kit:
   basename=gpfs
   description=GPFS 3.5.0-3 base realease.
   version=3.5.0
   release=3
   ostype=Linux
   kitlicense=EPL

kitrepo:
   kitrepoid=rhels6.4
   osbasename=rhels
   osmajorversion=6
   osminorversion=4
   osarch=x86_64

kitcomponent:
   basename=component-gpfs-base
   description=GPFS 3.5.0-3 base release
   version=3.5.0
   release=3
   serverroles=mgtnode,compute
   kitrepoid=rhels6.4
   kitpkgdeps=gpfs.base,gpfs.docs,gpfs.msg.en_US,gpfs.gpl

kitpackage:
   filename=gpfs.base-3.5.0-3.x86_64.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.base

kitpackage:
   filename=gpfs.docs-3.5.0-3.noarch.rpm
   kitrepoid=rhels6.4
```

```
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.base

kitpackage:
   filename=gpfs.gpl-3.5.0-3.noarch.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.base

kitpackage:
   filename=gpfs.msg.en_US-3.5.0-3.noarch.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.base

[root@i04n50 gpfs.base]#
```

For the GPFS updates, we built a kit that contains two kit components: one for the GPFS server nodes and one for the GPFS client nodes. The kit component for the client nodes is configured with another script that configures compute nodes as GPFS client nodes to access the network shared disks (NSD) servers, as shown in Example 3-18.

Example 3-18 Kit components for GPFS server and client nodes

```
[root@i04n50 gpfs.update]# cat buildkit.conf
kit:
   basename=gpfs-update
   description=GPFS 3.5.0-13 with portability layer
   version=3.5.0
   release=13
   ostype=Linux
   kitlicense=EPL

kitrepo:
   kitrepoid=rhels6.4
   osbasename=rhels
   osmajorversion=6
   osminorversion=4
   osarch=x86_64

kitcomponent:
   basename=component-gpfs-updates
   description=GPFS 3.5.0-13 updates, and portab. layer built into kernel
   version=3.5.0
   release=13
   serverroles=mgtnode,compute
   kitrepoid=rhels6.4
   kitcompdeps=component-gpfs-base
   kitpkgdeps=gpfs.base,gpfs.docs,gpfs.msg.en_US,gpfs.gpl,gpfs.gplbin

kitcomponent:
   basename=component-gpfs-updates-compute
   description=GPFS 3.5.0-13 updates, and portab. layer built into kernel
   version=3.5.0
   release=13
   serverroles=mgtnode,compute
```

```
   kitrepoid=rhels6.4
   kitcompdeps=component-gpfs-base
   kitpkgdeps=gpfs.base,gpfs.docs,gpfs.msg.en_US,gpfs.gpl,gpfs.gplbin
   postinstall=gpfs-addnode.sh

kitpackage:
   filename=gpfs.base-3.5.0-13.x86_64.update.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.update

kitpackage:
   filename=gpfs.docs-3.5.0-13.noarch.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.update

kitpackage:
   filename=gpfs.msg.en_US-3.5.0-13.noarch.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.update

kitpackage:
   filename=gpfs.gpl-3.5.0-13.noarch.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.update

kitpackage:
   filename=gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
   kitrepoid=rhels6.4
   isexternalpkg=no
   rpm_prebuiltdir=gpfs.update

[root@i04n50 gpfs.update]#
```

By studying the configuration file that is shown in Example 3-18 on page 94, you can see that the two kit components have dependencies on the GPFS base components. This means that you cannot install the update components if the base component is not installed.

The kit skeleton for the GPFS updates is shown in Example 3-19.

Example 3-19 Kit skeleton for the GPFS updates

```
[root@i04n50 gpfs.update]# ls -ltR
-rw-r--r-- 1 root root     1889 Oct 29 16:50 buildkit.conf
drwxr-xr-x 2 root root     4096 Oct 22 15:40 docs
drwxr-xr-x 3 root root     4096 Oct 22 15:40 other_files
drwxr-xr-x 2 root root     4096 Oct 22 15:40 plugins
drwxr-xr-x 3 root root     4096 Oct 30 09:41 scripts
drwxr-xr-x 4 root root     4096 Oct 25 21:29 source_packages

. . . (truncated )

./scripts:
```

```
-rwxr-xr-x 1 root root 1749 Oct 30 09:41 gpfs-addnode.sh
drwxr-xr-x 2 root root 4096 Oct 22 15:40 sample

. . . (truncated )

./source_packages:
drwxr-xr-x 2 root root 4096 Oct 25 21:29 gpfs.update
drwxr-xr-x 6 root root 4096 Oct 22 15:40 sample

./source_packages/gpfs.update:
-rw-r--r-- 1 root root 1866576 Oct 25 21:13 gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
-rw-r--r-- 1 root root   106817 Oct 25 21:13 gpfs.msg.en_US-3.5.0-13.noarch.rpm
-rw-r--r-- 1 root root   554372 Oct 25 21:13 gpfs.gpl-3.5.0-13.noarch.rpm
-rw-r--r-- 1 root root 13156422 Oct 25 21:13
gpfs.base-3.5.0-13.x86_64.update.rpm
-rw-r--r-- 1 root root   254205 Oct 25 21:13 gpfs.docs-3.5.0-13.noarch.rpm

. . . (truncated )

[root@i04n50 gpfs.update]#
```

The kit component that is installed on compute nodes includes a script (`gpfs-addnode.sh`) that adds the compute nodes as GPFS clients. Take care when you are copying and pasting *“”* (see Example 3-20), because we assume that the GPFS server nodes are installed and configured. For this reason, we first install and configure the GPFS server nodes, and then the client nodes while we are building our cluster.

Example 3-20 gpfs-addnode script

```
[root@i04n50 scripts]# cat gpfs-addnode.sh
#!/bin/bash

# Define the primary and secondary GPFS configuration servers
PRIMARY=i04n51-gpfs1
SECONDARY=i04n52-gpfs2
CONFIGnode=$PRIMARY

# Collect current GPFS configuration from the GPFS configuration nodes
################################################################################
CLUSTER=$(ssh -o "StrictHostKeyChecking=no" ${PRIMARY} /usr/lpp/mmfs/bin/mmlscluster)

if [ "$CLUSTER" == "" ];
then
    CLUSTER=$(ssh -o "StrictHostKeyChecking=no" ${SECONDARY} /usr/lpp/mmfs/bin/mmlscluster)
    CONFIGnode=$SECONDARY

    if [ "$CLUSTER" == "" ];
    then
        exit 111
    fi
fi


# Determine if current node is a member of the GPFS cluster
################################################################################
MYIPS=$(ip addr | grep "inet " |awk '{print $2}' | cut -f 1 -d / | grep -v 127.0.0.1)

for IP in $MYIPS
do
    GPFSHOSTNAME=$(echo "$CLUSTER"| grep $IP | awk '{print $2}')
    if [ "$GPFSHOSTNAME" ];
    then
        break
    fi
done


# Add or restore node configuration
################################################################################
if [ "$GPFSHOSTNAME" == "" ];
```

```
then
    # This node is not part of GPFS cluster; let's add it
    ssh -o "StrictHostKeyChecking=no" ${CONFIGnode} /usr/lpp/mmfs/bin/mmaddnode -N `hostname`
    sleep 5
    /usr/lpp/mmfs/bin/mmchlicense client --accept -N `hostname`
    sleep 5
    /usr/lpp/mmfs/bin/mmstartup
    sleep 5
    /usr/lpp/mmfs/bin/mmmount all
    echo 'PATH=$PATH:/usr/lpp/mmfs/bin' >> /etc/profile
else
    # This node is defined in GPFS cluster; let's restore the GPFS database into this node
     /usr/lpp/mmfs/bin/mmsdrrestore -p ${CONFIGnode} -R /usr/bin/scp
    sleep 5
    /usr/lpp/mmfs/bin/mmstartup
    sleep 5
    /usr/lpp/mmfs/bin/mmmount all
    echo 'PATH=$PATH:/usr/lpp/mmfs/bin' >> /etc/profile
fi

[root@i04n50 scripts]#
```

4. Bundle the kit and add it to pHPC.

After you add all of your packages, scripts, and application parameters to the skeleton, you must validate your configuration by building the kit package repositories, followed by building the archive.

Validating the configuration of the kit is the process of parsing all of the information that you added to the skeleton and checking it to ensure that every statement in the `buildkit.conf` file is valid (for example, the scripts exists into the skeleton and can be run), as shown in Example 3-21.

Example 3-21 Validates the configuration

```
root@i04n50 gpfs.base]# buildkit chkconfig
No errors were found in Kit Build File /gpfs.base/buildkit.conf.
[root@i04n50 gpfs.base]#
```

After the validation is completed successfully, we can build our kit, as shown in Example 3-22.

Example 3-22 Building the kit

```
[root@i04n50 gpfs.base]# buildkit buildrepo all
Spawning worker 0 with 4 pkgs
Workers Finished
Gathering worker results
Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete
[root@i04n50 gpfs.base]#
```

This process includes the following overall steps:

1. Create a directory that acts as a repository into the named Kit Package Repository: `< Kit directory location >/build/kit_repodir/<Kit-Pkg-Repo>`.
2. Build the Component Metapackages that are associated with the Kit Package Repository and add all related packages to the Kit Package Repository directory.
3. Build the repository metadata for the Kit Package Repository. The repository metadata is based on the OS native package format. For example, for RHEL, we build the YUM repository metadata by using the **createrepo** command.

After the repository is built, the skeleton receives several other directories that are generated by the build command (see Example 3-23).

Example 3-23 Directories that are generated by the build command

```
[root@i04n50 gpfs.base]# ls -ltR
drwxr-xr-x 4 root root     4096 Oct 29 14:38 build
drwxr-xr-x 8 root root     4096 Oct 29 14:38 rpmbuild
drwxr-xr-x 2 root root     4096 Oct 29 14:38 tmp
-rw-r--r-- 1 root root     1257 Oct 27 22:26 buildkit.conf
drwxr-xr-x 4 root root     4096 Oct 25 21:18 source_packages
drwxr-xr-x 3 root root     4096 Oct 22 15:40 scripts
drwxr-xr-x 2 root root     4096 Oct 22 15:40 docs
drwxr-xr-x 3 root root     4096 Oct 22 15:40 other_files
./build:
drwxr-xr-x 3 root root 4096 Oct 29 14:38 gpfs-3.5.0-3
drwxr-xr-x 3 root root 4096 Oct 29 14:38 kit_repodir
./build/gpfs-3.5.0-3:
lrwxrwxrwx 1 root root   43 Oct 29 14:38 repos -> /gpfs.base/build/kit_repodir
-rw-r--r-- 1 root root  934 Oct 29 14:38 kit.conf
./build/kit_repodir:
drwxr-xr-x 3 root root 4096 Oct 29 14:38 gpfs-3.5.0-3-rhels-6.4-x86_64
./build/kit_repodir/gpfs-3.5.0-3-rhels-6.4-x86_64:
drwxr-xr-x 2 root root     4096 Oct 29 14:38 repodata
-rw-r--r-- 1 root root     1459 Oct 29 14:38
component-gpfs-base-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root 12405953 Oct 25 21:05 gpfs.base-3.5.0-3.x86_64.rpm
-rw-r--r-- 1 root root   230114 Oct 25 21:05 gpfs.docs-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root   509477 Oct 25 21:05 gpfs.gpl-3.5.0-3.noarch.rpm
-rw-r--r-- 1 root root    99638 Oct 25 21:05 gpfs.msg.en_US-3.5.0-3.noar.rpm
./build/kit_repodir/gpfs-3.5.0-3-rhels-6.4-x86_64/repodata:
-rw-r--r-- 1 root root 2979 Oct 29 14:38 repomd.xml
-rw-r--r-- 1 root root 9690 Oct 29 14:38 7149( ... truncated
)528-primary.sqlite.bz2
-rw-r--r-- 1 root root 5096 Oct 29 14:38 5d94( ... truncated
)354-filelists.sqlite.bz2
-rw-r--r-- 1 root root  932 Oct 29 14:38 d87c( ... truncated )cdc-other.sqlite.bz2
-rw-r--r-- 1 root root 2995 Oct 29 14:38 228b( ... truncated )896-primary.xml.gz
-rw-r--r-- 1 root root  451 Oct 29 14:38 3b7e( ... truncated )f9c-other.xml.gz
-rw-r--r-- 1 root root 3970 Oct 29 14:38 eb19( ... truncated )203-filelists.xml.gz
[root@i04n50 gpfs.base]#
```

4. After successfully building your repositories into the skeleton, pack all files into a `.tar` file, which is added later to the pHPC as a kit, as shown in Example 3-24.

Example 3-24 Building the .tar file

```
[root@i04n50 gpfs.base]# buildkit buildtar
Kit tar file /gpfs.base1/gpfs-3.5.0-3.tar.bz2 successfully built.
[root@i04n50 gpfs.base]#
```

Note: If you must build repositories for OS distributions, versions, or architectures that do not match the current system, you might need to copy your kit template directory to an appropriate server to build that repository, and then copy the results back to your main build server.

Now that our kit is built, we can add it to the pHPC database by using the GUI, as shown in Figure 3-12.

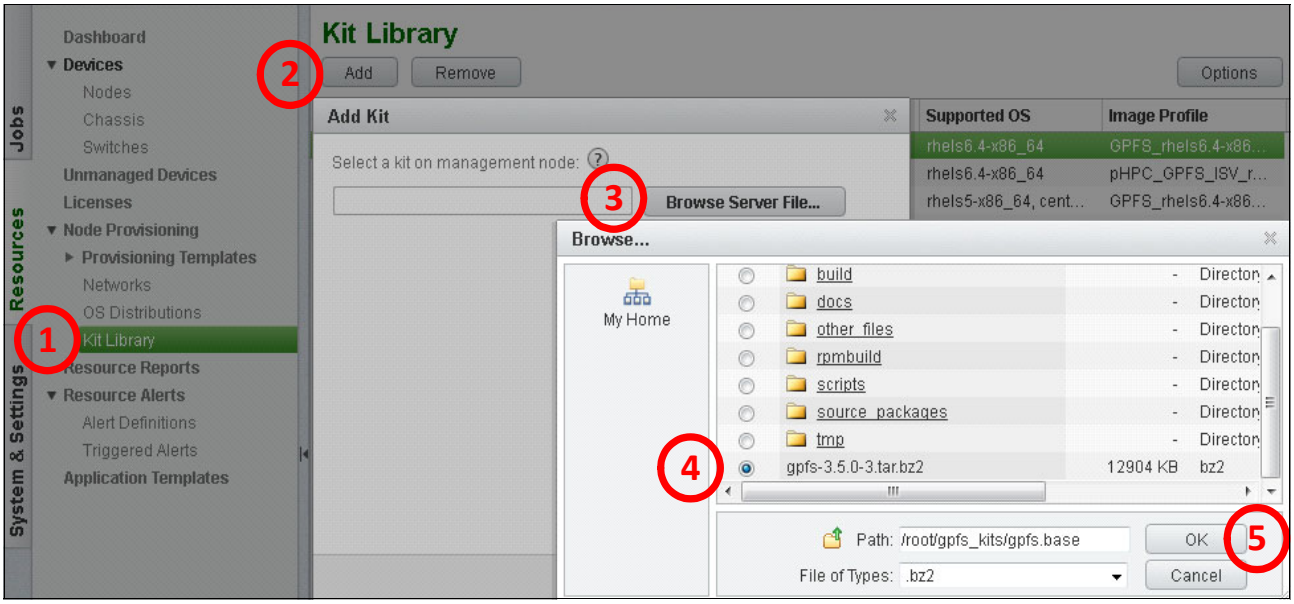

Figure 3-12 Kit Library GUI

Network profiles

A network profile defines the network interface configurations that are used by a compute node that is deployed through pHPC. to build a network profile to be applied to a compute node during the provisioning phase, we must define the building blocks of a network profile. The building blocks are the logical networks and interfaces.

You can define many types of logical networks according to your needs. A network is represented by a subnet and includes an IP address range and a gateway.

Normally, you define a logical network that is based on your cluster requirements. For example, in our cluster during the installation, we defined two logical networks for two purposes: one on which we provision the compute nodes, and the second is a BMC network that is used for remote hardware management. But as this cluster is not built only for compute nodes provisioning, we must build a network for the application's communication that is installed on the nodes. All of the communication between the nodes of the application flows through this network, as shown in Figure 3-13 on page 100.

Note: At this point, we do not describe any association of this network with any of the interfaces of the compute nodes. A logical network definition is a subnet, which includes an IP address range and a gateway.

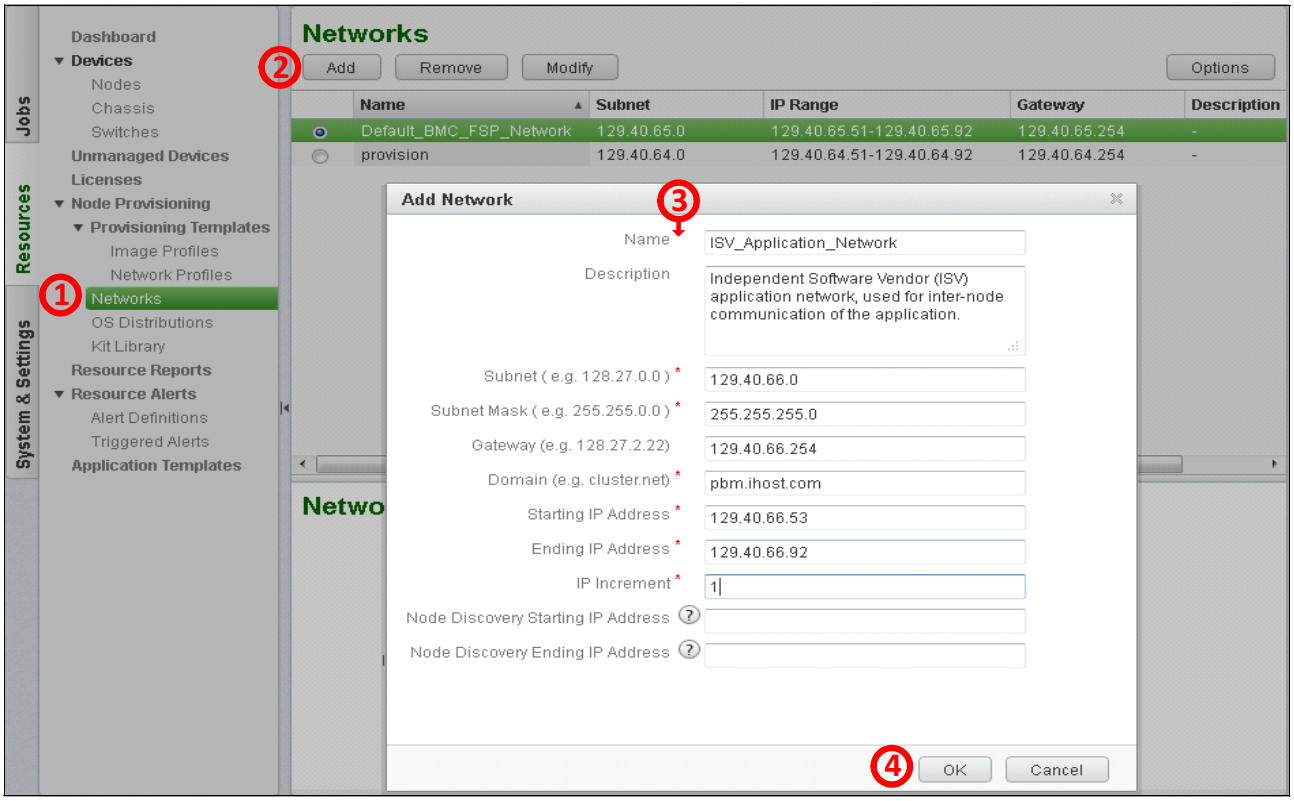

Figure 3-13 Network configuration profile

During the installation phase of pHPC, a default network profile is created, which includes the provisioning and BMC networks that were defined during the installation process, as shown in Figure 3-14.

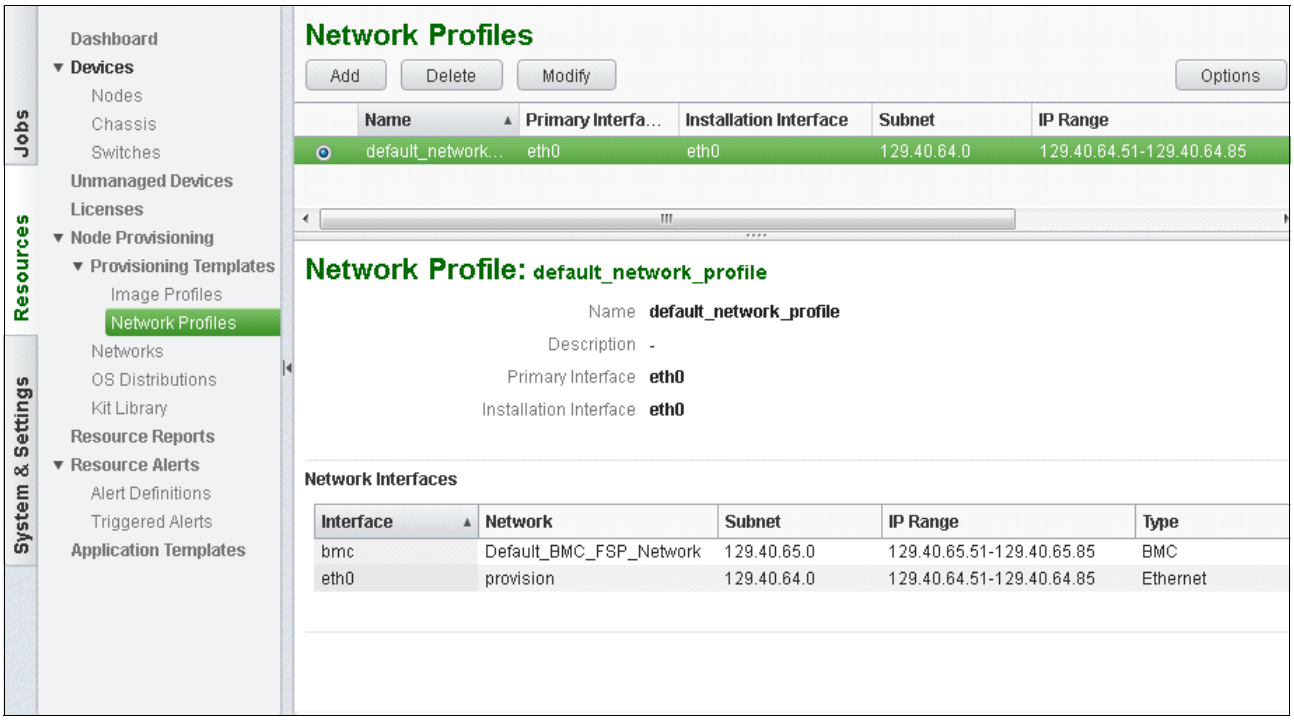

Figure 3-14 Network Profile GUI

This network profile is enough to provision your compute nodes. However, if you need more networks to define a new network profile (see Figure 3-14 on page 100), add the logical networks that you defined (for example, ISV-Application-Network) and the predefined logical networks (provision and Default_BMC_FSP_Network).

To add a network profile for provisioning the compute nodes and another network for the application, complete the steps that are shown in Figure 3-13 on page 100.

In some environments (which is not the case for our cluster), it might be necessary to build a bonding interface out of few available interfaces on the compute nodes. To do this, you must create a script that performs the bonding and place it into the `/install/postscripts` directory.

The best option while creating this script is to build a generic script, which accepts the `bonding-config.sh bond0 eth1 eht2` parameters. Figure 3-15 shows how to associate a logical network to a bonding interface.

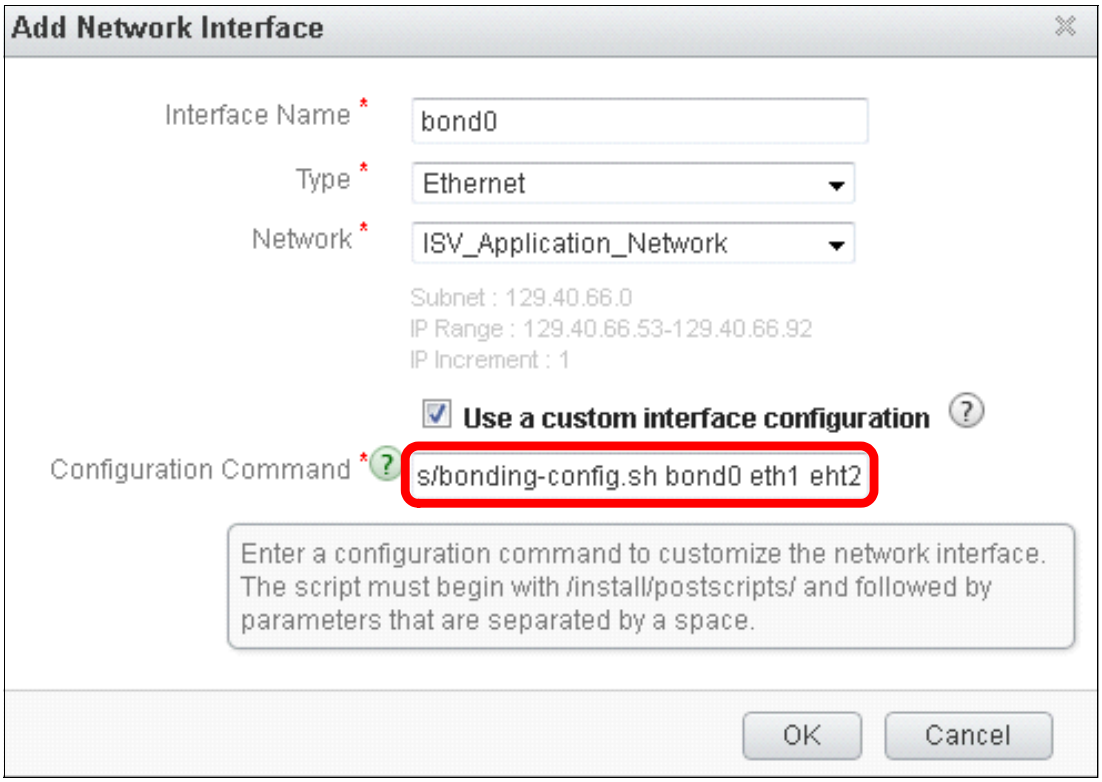

Figure 3-15 Adding a network interface GUI

Image profiles

Image profiles represent a logical definition of what is provisioned on the compute nodes, including the operating system image and other software (kits) with its configuration scripts, postinstallation and post-boot scripts, and eventual custom packages and kernel modules, as shown in Figure 3-16.

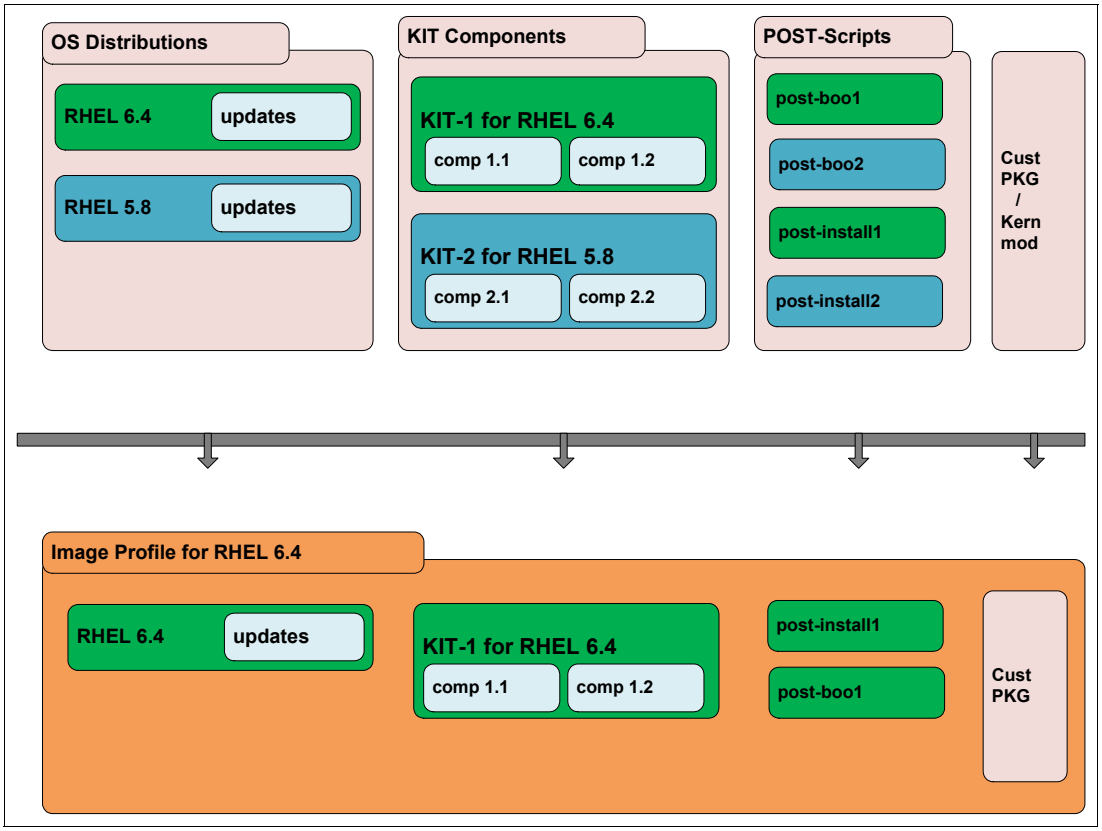

Figure 3-16 Image profiles

Image profiles are used to provision and eventually update the compute nodes with all the software linked (added) to the image profile.

Note: While you are working (adding or modifying) with an image profile, if you add a package that has dependencies, the related dependent packages are automatically installed.

Remember that an image profile is associated (linked) to a compute node, and any modification you make to an image profile makes the node out of sync. After you finish your modifications to your existing profile, you can synchronize all of the nodes that use that image profile.

If you do not know exactly which nodes are potentially affected by this image profile modification, click the menu **Devices** → **Nodes** (see Figure 3-17 on page 103), click **Options** and then click **Image Profile**. After this modification, you have a new colon in your **Devices** → **Nodes** window that is named Image Profile, which tells you the associated image profile for each node.

As shown in Figure 3-17, an image profile is always built on top of an OS distribution. Whenever you add an OS distribution to your pHPC database, two image profiles are automatically created, a stateful image profile and a stateless image profile. You cannot create an image profile from scratch, but you can create one by duplicating a pre-generated image (stateless of stateful) that was added by the OS distribution creation.

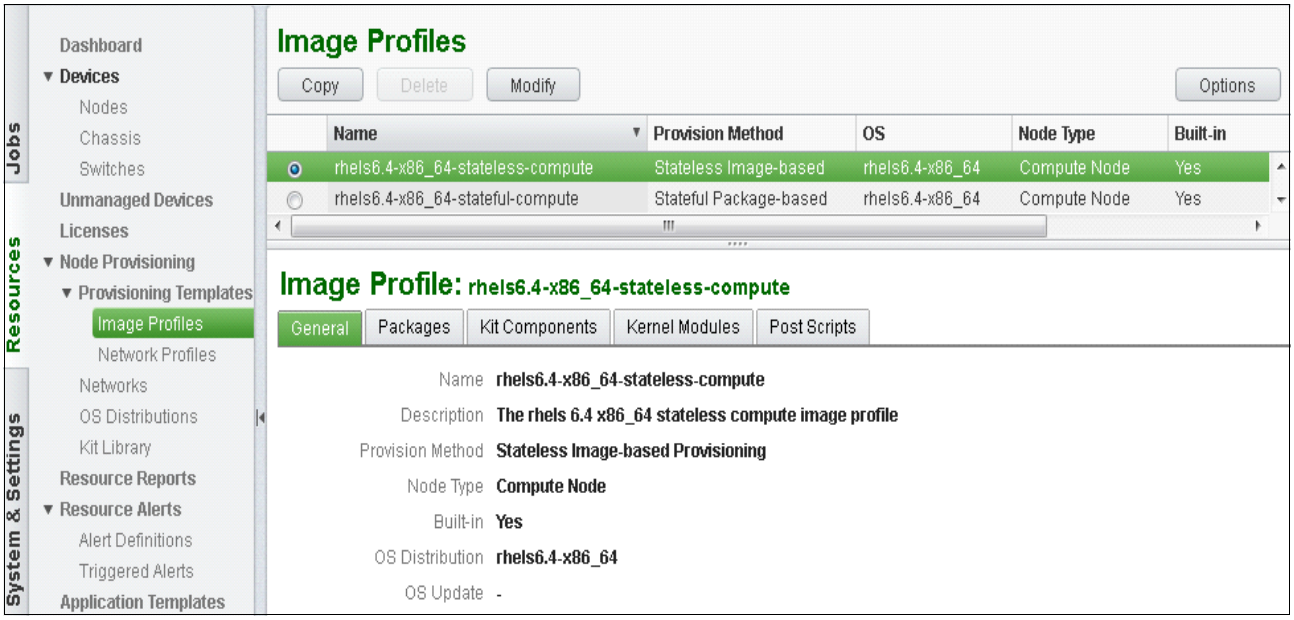

Figure 3-17 Image profiles GUI

Each OS distribution is copied to a default destination directory `/install/OS-name/OS-arch`, in which OS-name is the distribution name, and OS-arch is the OS architecture.

Stateful image profile is a package-based profile that helps to deploy the operating system and related software components onto persistent storage of compute nodes (local disk, SAN disk, or iSCSI device), and the changes that are made are persistent across node reboots.

Stateless image profile is an image-based profile that helps to deploy the operating system and related software components into memory, which makes the changes non-persistent across reboots. Installing a compute node by using a provisioning template that is based on a stateless image profile is faster because the compute node does not install all the packages at provisioning time. Instead, it uses a pre-generated stateless image (with all components) that is built and stored in the management node.

To create an image profile, you must make a copy of an existing image profile (that was built in by OS Distribution creation) and modify it according to your needs, as shown in Figure 3-18.

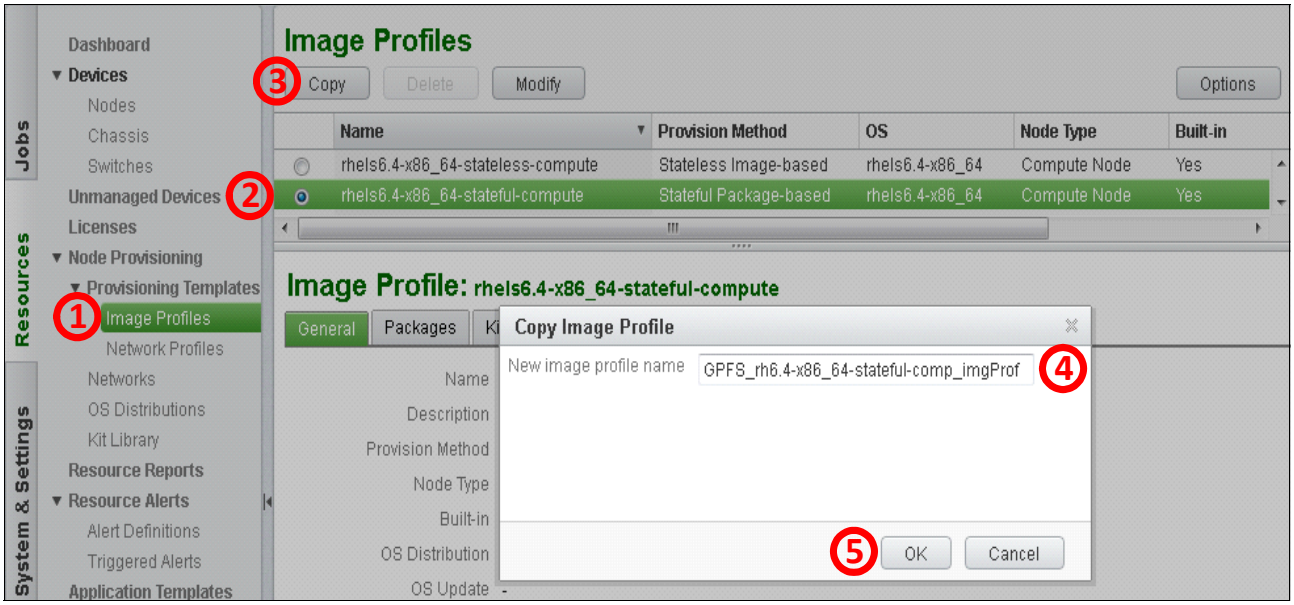

Figure 3-18 Image Profiles creation

Note: An image profile can be deleted only if it is not used by any nodes and provisioning templates. You cannot delete image profiles that were created by the installer.

In our cluster scenario, we intend to install the GPFS NSD servers by using pHPC because it has an advantage in that the SSH key exchange between the nodes is automatically done by pHPC.

For this reason, we are modifying the image profile we created and customizing it according to our needs, as shown in Figure 3-19. To modify your new image profile, select your image profile in the list and click **Modify**. A window opens that includes many tabs for changing different aspects of your image profile.

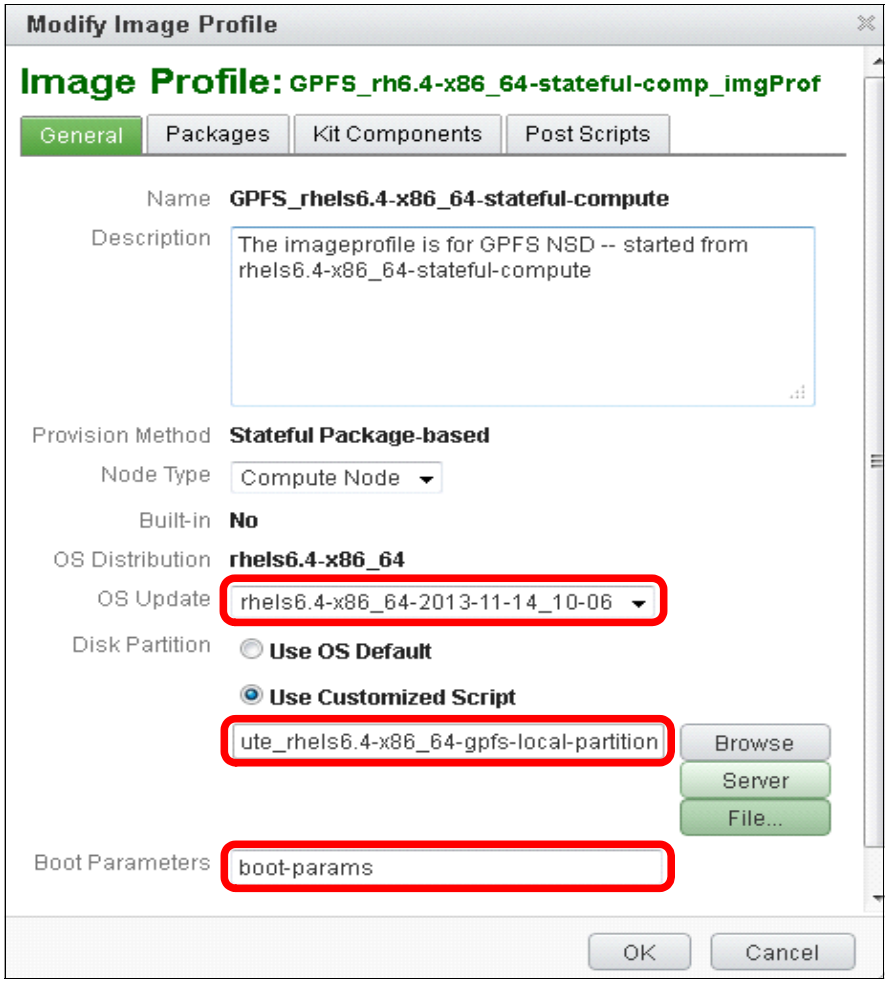

Figure 3-19 Modifying the image profile GUI

In the first tab, you can modify several key parameters of your profile. The first option is to choose whether you want to update the OS distribution on which this image profile was built when you perform the node provisioning. By default, it does not update the OS distribution.

If you want to instruct the installer on how to partition your local disks on the compute nodes, add a file into the `/install/partitionscripts` directory that contains your partitioning layout. For example, in our scenario, we need a certain layout of the internal disks of the GPFS NSD servers as we share only internal disks of these nodes to our compute nodes. Although this is not a recommended setup for a production environment, it is sufficient for our demonstration purposes.

Example 3-25 on page 106 shows the partitioning layout that we used for our setup.

Example 3-25 Partition layout

```
[root@i04n50 partitionscripts]# cat
GPFS_compute_rhels6.4-x86_64-gpfs-local-partition
#!/bin/sh

# Use Anaconda to create and format LVM partition

cat << EOF > /tmp/partitionfile

bootloader   --location=partition --driveorder=sda
clearpart --all --drives=sda --initlabel

#For UEFI machine, /boot/efi is necessary for using
part /boot/efi --size=100 --fstype vfat --ondisk=sda
part /boot      --fstype ext4 --size=200 --ondisk=sda

part /gpfs      --fstype ext4 --size=100000 --ondisk=sda
part pv.00       --size=100 --grow  --asprimary --ondisk=sda

volgroup rootvg --pesize=32768 pv.00
logvol /        --fstype ext4 --name=rootlv --vgname=rootvg --size=8192
logvol swap     --fstype swap --name=swaplv --vgname=rootvg --size=1024 --grow
--maxsize=4096

EOF
[root@i04n50 partitionscripts]#
```

Note: For our scenario, after the GPFS NSD nodes installation, we unmounted `/gpfs`, deleted its corresponding line from `/etc/fstab`, and then used the device `/dev/sda4` as a GPFS NSD device.

As shown in Figure 3-20 on page 107, the Packages tab is where you can choose the packages that are installed during the provisioning phase. By default, not all of the packages from the OS distribution on which this image profile is built are installed. Therefore, if you still need some other packages that are not installed by default, you can select them from the OS Packages section, as shown in Figure 3-20 on page 107. The list of the packages is normally long and you can browse through the packages by using the arrows at the top of the list. If you are looking for specific packages and you know their names, you can use the Filter option.

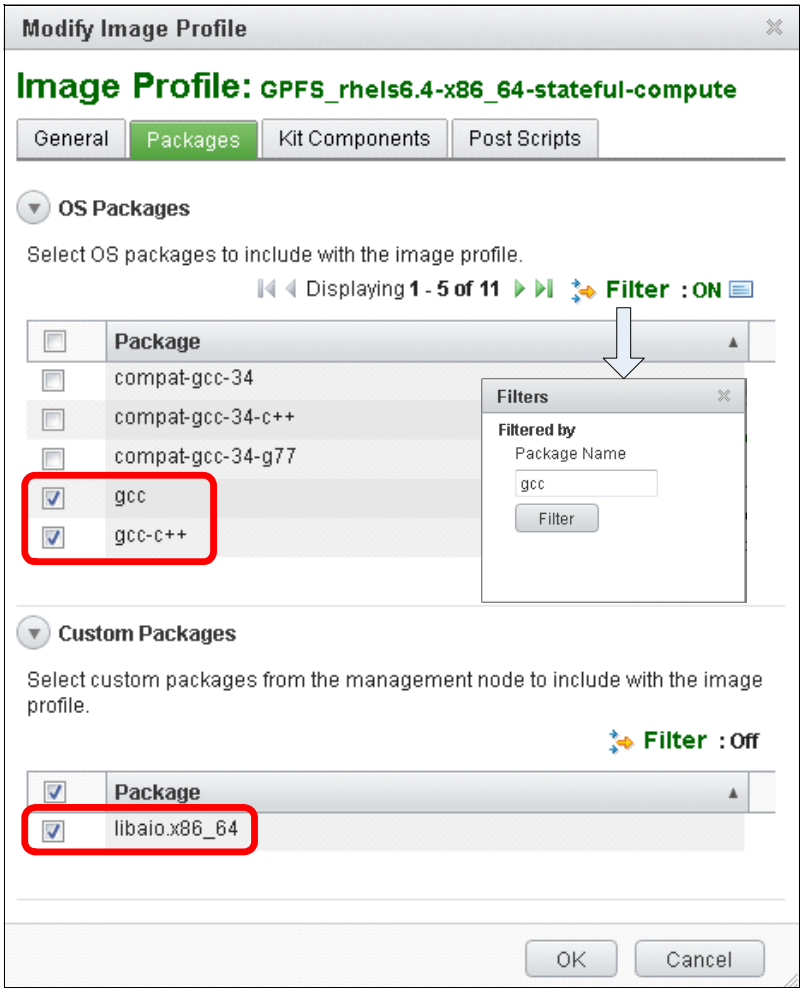

Figure 3-20 Modifying image profile GUI

If you need to install custom packages (which are not present in the OS distribution), place these packages into the `/install/contrib/OS-version/OS-architecture` directory and then refresh the GUI by using the following command:

```
[root@i04n50]# plcclient.sh -d "pcmimageprofileloader“
```

Note: In our case scenario, we used a custom package as an example because it was not needed by any application or any other component.

As shown in Figure 3-21, by using the Kit Components tab, you can choose which KIT or Kit components are installed in your nodes. For example, for our GPFS NSD servers, we chose to install only `gpfs-base`, its corresponding updates (`gpfs-updates`), and `phpc-base`.

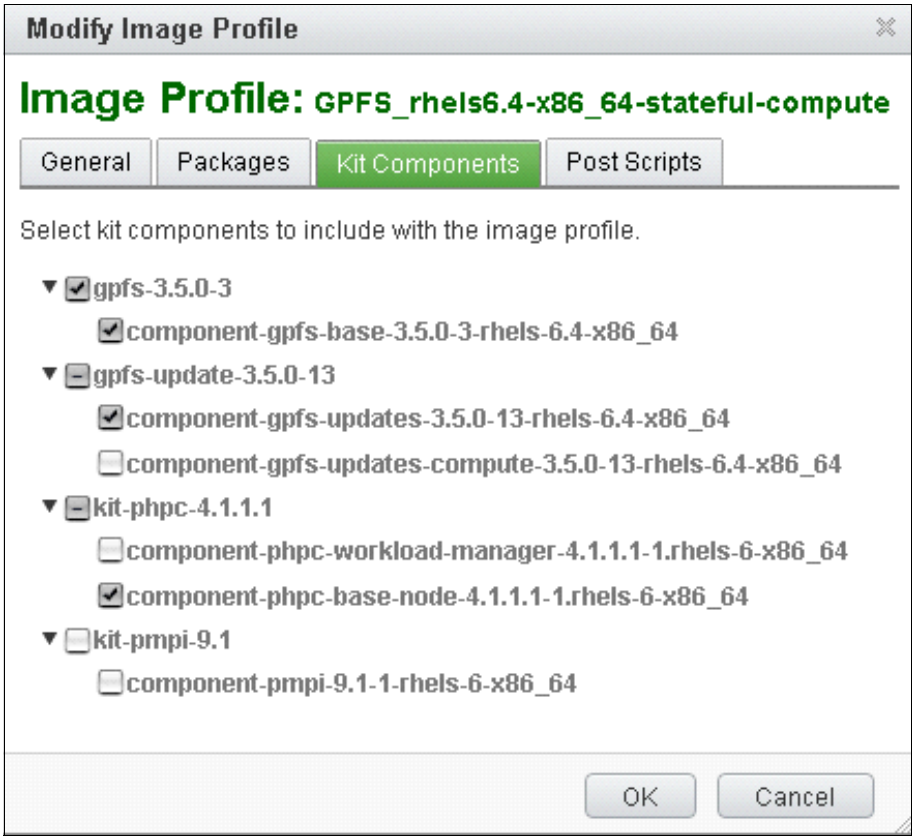

Figure 3-21 Modifying an image profile GUI

As shown in Figure 3-22 on page 109, by using the Post Scripts tab, you can specify which post scripts to run at boot or installation time. You can add the following types of custom scripts to an image profile:

- Post-install scripts are run on each node that is provisioned with this image profile only when the node is installed or reinstalled.
- Post-boot scripts are run on each node that is provisioned with this image profile each time the node is rebooted or during node update operations. If you modify the post-boot script, you must reboot the nodes or perform a synchronize operation.

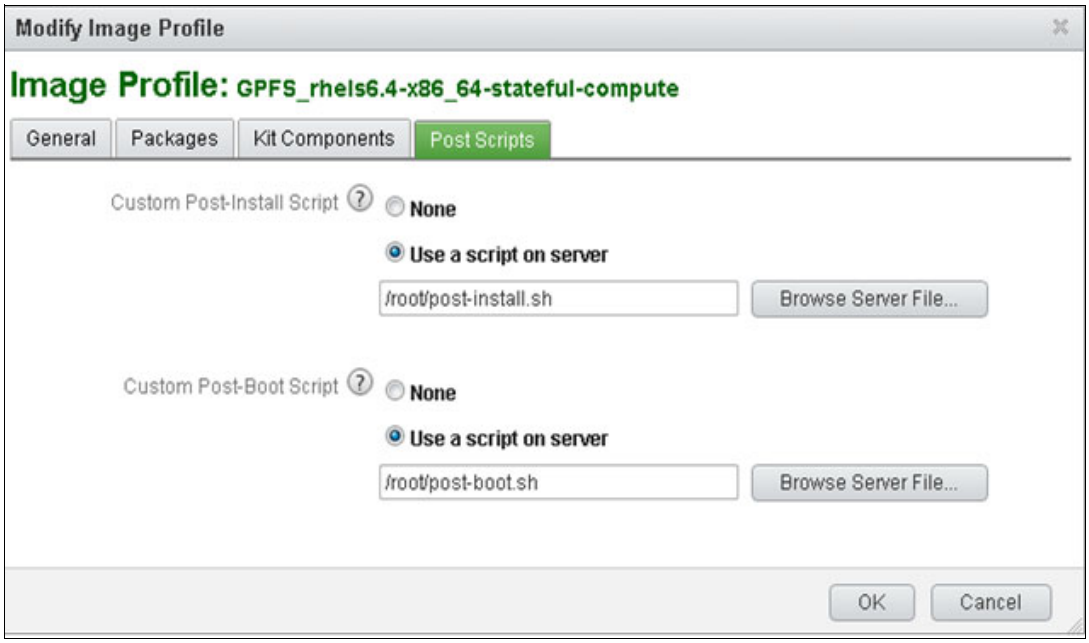

Figure 3-22 Adding post scripts

For the stateless image profiles, you find the Kernel Modules tab in which you can specify other modules or drivers to be added to the kernel image `initrd.img`, as shown in Figure 3-23.

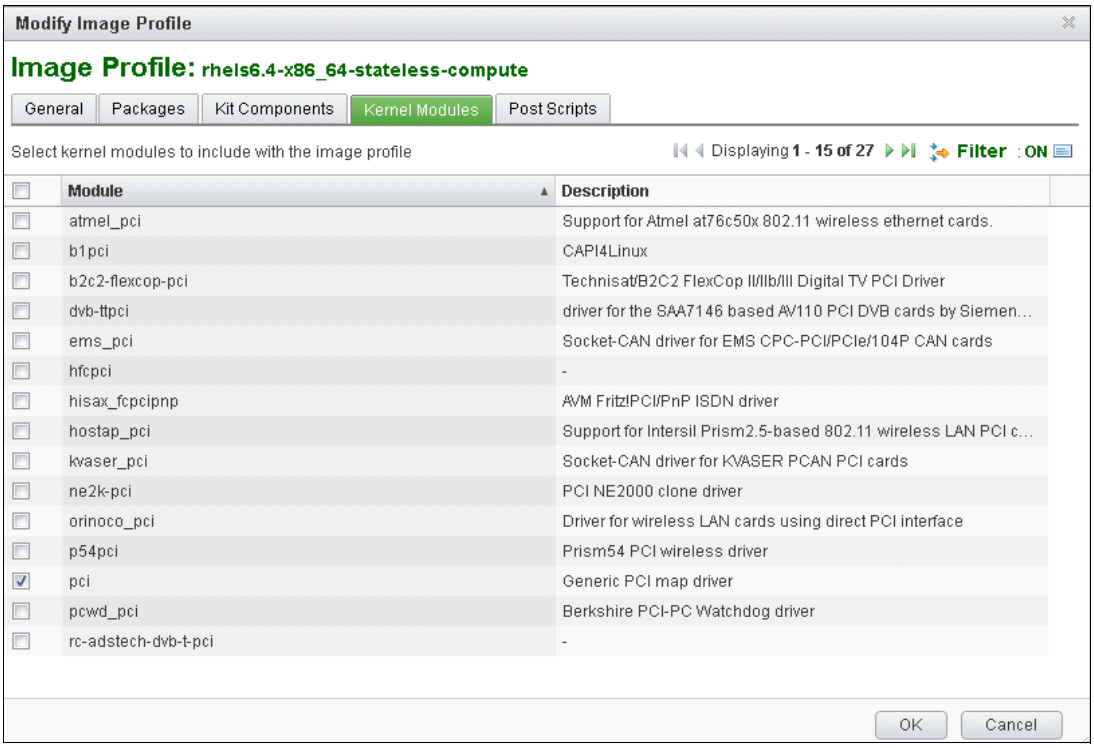

Figure 3-23 Kernel modules tab

The default `initrd.img` of stateless images contains a limited set of drivers and you might encounter errors when you provision stateless nodes because of the lack of drivers.

Note: Remember to use the Filter (green) option to filter for your specific needs. Also, the image profile can be deleted only if it is not used by any nodes and provisioning templates. You cannot delete image profiles that were created by the installer.

Hardware profiles

A hardware profile is a collection of parameters that are used by pHPC when remote management commands are run on different types of hardware. For example, to list your hardware profiles that are defined into your pHPC database, use the command as shown in Example 3-26.

Example 3-26 Listing hardware profiles from the Platform HPC database

```
[root@i04n50 ~]# tabdump nodehm
#node,power,mgt,cons,termserver,termport,conserver,serialport,serialspeed,serialfl
ow,getmac,cmdmapping,comments,disable
"__HardwareProfile_IPMI",,"ipmi",,,,,"0","19200","hard",,"/opt/pcm/etc/hwmgt/mappi
ngs/HWCmdMapping_ipmi.xml",,
"__HardwareProfile_IBM_Flex_System_x",,"ipmi",,,,,"0","115200","hard",,"/opt/pcm/e
tc/hwmgt/mappings/HWCmdMapping_flex_x.xml",,
"__HardwareProfile_IBM_System_x_M4",,"ipmi",,,,,"0","115200","hard",,"/opt/pcm/etc
/hwmgt/mappings/HWCmdMapping_rackmount_x.xml",,
"__HardwareProfile_IBM_iDataPlex_M4",,"ipmi",,,,,"0","115200","hard",,"/opt/pcm/et
c/hwmgt/mappings/HWCmdMapping_rackmount_x.xml",,
"__HardwareProfile_IBM_idataplex_m3",,"ipmi",,,,,"0","115200","hard",,"/opt/pcm/et
c/hwmgt/mappings/HWCmdMapping_idataplex_m3.xml",,
 [root@i04n50 ~]#
```

Note: The last line in Example 3-26 is the definition of a profile that was added manually. We defined a new profile because we are using different hardware than the default that was predefined in pHPC.

To add a profile, use the command that is shown in Example 3-27.

Example 3-27 Adding a profile

```
[root@i04n50 ~]# pcmaddhwprofile name=IBM_idataplex_m3 type=ipmi
cmdmapping=/opt/pcm/etc/hwmgt/mappings/HWCmdMapping_idataplex_m3.xml"
serialspeed="115200" bmcportmode=dedicated bmcuser=USERID  bmcpassword=PASSWORD
```

where:

- `cmdmapping` is the xCAT mapping file that maps Platform Cluster Manager command to xCAT commands and is set to the absolute path of the mapping file.
- `serialspeed` is the serial speed number and is set to 115200, instead of the default value of 19200.
- `bmcportmode` is the BMC interface mode and is set to dedicated or shared for the hardware management profile type `ipmi`.
- `bmcuser` and `bmcpassword` is set to a valid BMC user name and password that is defined for the cluster.

Note: Because there were problems connecting to the serial console of the servers, we changed the default serial speed parameter 9600 - 115200. To make this change, you must run the **`tabedit nodehm`** command and modify the speed parameter; the **`tabedit`** command opens a vi editor.

Provisioning template

Now that we defined all of the components that are needed to build a provisioning template, we define the provisioning template. A provisioning template is a logical concept that exists only in the GUI that aggregates the software, network, and hardware attributes of a node. A provisioning template is built based on common characteristics (profiles) of nodes to be provisioned, including an image profile, network profile, and a hardware profile.

A provisioning template can be used to provision one node or several nodes at a time. The provisioning template is used only during provisioning phase.

The provisioning template is not attached to the nodes it was used to provision. When nodes are added to the cluster, they are linked (attached) to the image profile, network profile, and a hardware profile, not to the provisioning template.

The installer creates two templates (one stateful, one stateless), as shown in Figure 3-24.

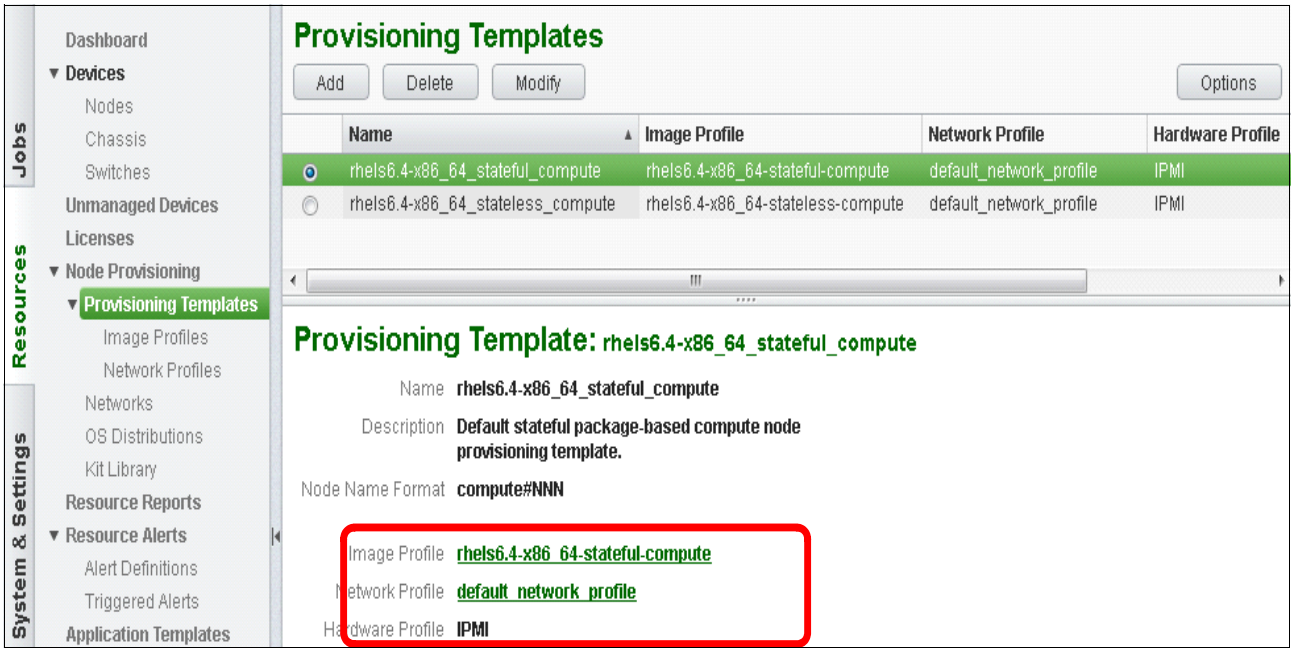

Figure 3-24 Provisioning templates GUI

The two default provisioning templates are built based on the following default image, network, and hardware profiles:

- For stateful compute nodes it uses:
  - Image profile = "`<os-arch>-stateful-compute`"
  - Network profile = "`default_network_profile`"
  - Hardware profile = `IPMI`

- For stateless compute nodes it uses:
  - Image profile = “<os-arch>-stateless-compute”
  - Network profile = “default_network_profile”
  - Hardware profile = IPMI

Now that all of the components are described, we can create a provisioning template (see Figure 3-25) that is used for provisioning our GPFS NSD server nodes.

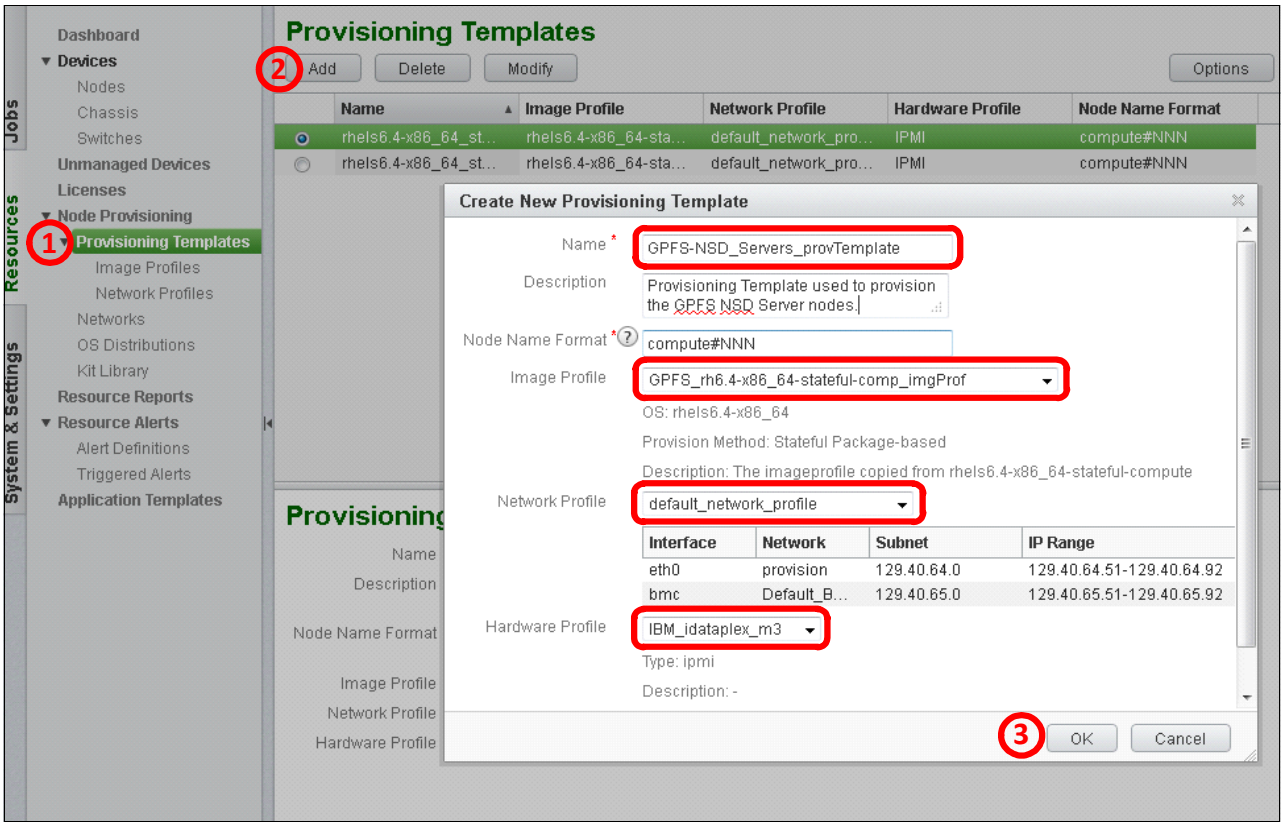

Figure 3-25 Creating provisioning templates

For our scenario to provision the GPFS NSD server nodes, you first must select the nodes into the GUI then click **Reinstall**, which starts a wizard that guides you through the provisioning process. Because of our preparation work in which the image, network, and hardware profiles were assigned to a provisioning template, we must use only this provisioning template to provision the node. You can see in Figure 3-25, after selecting the provisioning template, all of its components are displayed at the bottom of the wizard window.

Provisioning can garner the following results:

- Status: **[Defined], [Installing]**, and **[Booting Booted]** where:
  - Defined: The node that is added to cluster is deployed or reprovisioned when started.
  - Installing (Stateful): Installing OS for stateful compute node (no such status if not power-on via GUI, thus, not available for first provisioning).
  - Booting (Stateful only): In process to boot-up.
  - Booted: The node is booted up successfully.
  - Failed: Running the postbootscript failed.

Synchronization status indicates whether the node is in sync with the configuration of corresponding Image Profile. The available statuses are: “synced”, “out-of-sync”, or “-” (Unknown).

Limitation does not reflect the synchronization status for configuration files, which cannot be modified via GUI, for example, `/etc/fstab` and `/etc/passwd`.

The CFM function can be used from the GUI via the nodes’ Synchronize action or by using the **`updatenode`** command. Each Image profile has the following CFM-related files on the management node (MN), which determines what is synchronized:

```
/install/osimages/<Image_Profile>/synclist.cfm
/install/osimages/<Image_Profile>/pkglist.cfm
```

The `synclist.cfm` file contains a list of files to be synchronized from the management node to the client nodes by using the methods:

- Replacing the content of the destination file with the content of the source file. The source file is in the left of the arrow → (as shown in the following example) and represents a file on the management node. The destination is on the right side of the arrow → and represents the file to be modified on the destination compute node. For example, the following line is overwriting the content of `/etc/hosts` on the compute nodes with the content of `/etc/hosts` from the management node. This happens because `…/cfmdir/etc/hosts` file is a soft link to the following line:

  `/install/osimages/<Image_Profile>/cfmdir/etc/hosts → /etc/hosts`

- Merging the content of source with destination file:

  `/install/osimages/<Image_Profile>/cfmdir/etc/passwd.merge → /etc/hosts`

A complete example file for both methods is shown in Example 3-28.

Example 3-28 Replacing and merging files

```
/install/osimages/<Image_Profile>/cfmdir/etc/hosts -> /etc/hosts
MERGE:
/install/osimages/<Image_Profile>/cfmdir/etc/shadow.merge -> /etc/shadow
/install/osimages/<Image_Profile>/cfmdir/etc/group.merge -> /etc/group
/install/osimages/<Image_Profile>/cfmdir/etc/passwd.merge -> /etc/passwd
```

The `cfmdir` file acts as a repository that stores files to be pushed to compute nodes, as shown in Example 3-29. The files in the `/install/osimages/<Image_Profile>/cfmdir/*` directory can be flat files or soft links to the files on the management’s node file system (for example, hosts → `/etc/hosts`).

Example 3-29 cfmdir repository

```
[root@i04n50 cfmdir]# ls -l etc/*
lrwxrwxrwx 1 root root 16 Nov 12 11:56 group.merge -> /etc/group.merge
lrwxrwxrwx 1 root root 10 Nov 12 11:56 hosts -> /etc/hosts
lrwxrwxrwx 1 root root 17 Nov 12 11:56 passwd.merge -> /etc/passwd.merge
lrwxrwxrwx 1 root root 17 Nov 12 11:56 shadow.merge -> /etc/shadow.merge

[root@i04n50 etc]# cat /etc/group.merge
ngs2:x:31002:
phpcadmin:x:30495:
[root@i04n50 etc]#
```

The `pkglist.cfm` file contains a list of packages to be synchronized with the compute nodes. For example, if you define a package in this file and the package is part of the OS distribution that belongs to this Image Profile, the compute nodes are updated with this package when you run an **updatenode** command.

After GPFS NSD server nodes are provisioned by using the GPFS-NSD_Servers_provTemplate, the nodes are ready to be configured as NSD nodes. During the provisioning, all GPFS software and updates are installed.

In the next section, we describe the steps that are used to configure the NSD nodes for our scenario. These nodes are used only for demonstration purposes. If you are deploying for production, plan according to your cluster needs. Each NSD node has one internal disk, which we use as a shared disk by replicating the data across nodes. Remember that our Image Profile included a partitioning script that creates an `ext4` partition that is mounted under the `/gpfs` directory. We reuse this partition and allocate it to GPFS to be used as an NSD disk.

Note: Unmount the `/gpfs` directory and comment its corresponding line in `/etc/fstab`.

Complete the following steps to configure the NSD nodes:

1. Configure the profile of both nodes, as shown in Example 3-30.

Example 3-30 Profile configuration

```
[root@i04n50 ~]# echo 'PATH=$PATH:/usr/lpp/mmfs/bin' >> /etc/profile
[root@i04n50 ~]# . /etc/profile
```

2. Define the GPFS nodes as quorum nodes and file system managers by creating a file as shown in Example 3-31.

Example 3-31 Defining GPFS node and file system managers

```
[root@i04n50 ~]# cat /tmp/nodes
i04n51-gpfs1:manager-quorum
i04n52-gpfs2:manager-quorum
[root@i04n50 ~]#
```

3. Create the cluster as shown in Example 3-32.

Example 3-32 Creating the GPFS cluster

```
[root@i04n51-gpfs1 ~]# mmcrcluster -N /tmp/nodes -p i04n51-gpfs1 -s
i04n52-gpfs2 -r /usr/bin/ssh -R /usr/bin/scp -C pHPC -U pbm.ihost.com –A

Warning: Permanently added 'i04n51-gpfs1' (RSA) to the list of known hosts.
Mon Oct 28 20:54:44 EDT 2013: mmcrcluster: Processing node i04n51-gpfs1
Mon Oct 28 20:54:44 EDT 2013: mmcrcluster: Processing node i04n52-gpfs2
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@i04n51-gpfs1 ~]#
```

4. List the configuration of your cluster and licensing, as shown in Example 3-33.

Example 3-33 Listing the cluster configuration

```
[root@i04n51-gpfs1 ~]# mmlscluster

===============================================================================
| Warning:                                                                    |
|   This cluster contains nodes that do not have a proper GPFS license        |
|   designation.  This violates the terms of the GPFS licensing agreement.    |
|   Use the mmchlicense command and assign the appropriate GPFS licenses      |
|   to each of the nodes in the cluster.  For more information about GPFS     |
|   license designation, see the Concepts, Planning, and Installation Guide.  |
===============================================================================

GPFS cluster information
========================
  GPFS cluster name:         pHPC.i04n51-gpfs1
  GPFS cluster id:           3691382246057379667
  GPFS UID domain:           pbm.ihost.com
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp

GPFS cluster configuration servers:
-----------------------------------
  Primary server:    i04n51-gpfs1
  Secondary server:  i04n52-gpfs2

 Node  Daemon node name  IP address    Admin node name  Designation
-----------------------------------------------------------------------
   1   i04n51-gpfs1      129.40.64.51  i04n51-gpfs1     quorum-manager
   2   i04n52-gpfs2      129.40.64.52  i04n52-gpfs2     quorum-manager
```

5. Before you configure your cluster, you must accept the product license as shown in Example 3-34.

Example 3-34 Step to accept the license

```
[root@i04n51-gpfs1 ~]# mmchlicense server --accept -N i04n51-gpfs1,i04n52-gpfs2

The following nodes will be designated as possessing GPFS server licenses:
   i04n51-gpfs1
   i04n52-gpfs2
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@i04n51-gpfs1 ~]

[root@i04n51-gpfs1 ~]#  mmlslicense -L
 Node name                      Required license   Designated license
--------------------------------------------------------------------
i04n51-gpfs1                        server             server
i04n52-gpfs2                        server             server

 Summary information
---------------------
Number of nodes defined in the cluster:                       2
```

```
Number of nodes with server license designation:                2
Number of nodes with client license designation:                0
Number of nodes still requiring server license designation:     0
Number of nodes still requiring client license designation:     0
[root@i04n51-gpfs1 ~]#
```

6. Define the NSD shared disks, as shown in Example 3-35.

Example 3-35 Defining the NSD shared disks

```
[root@i04n50 ~]# cat /tmp/nsd
%nsd: device=/dev/sda3
   servers=i04n51-gpfs1
   usage=dataAndMetadata
   failureGroup=11
%nsd: device=/dev/sda3
   servers=i04n52-gpfs2
   usage=dataAndMetadata
   failureGroup=12
[root@i04n50 ~]#
```

7. Create the NSD, as shown in Example 3-36.

Example 3-36 Creating the NSD

```
[root@i04n51-gpfs1 ~]# mmcrnsd -F /tmp/nsd
mmcrnsd: Processing disk sda3
mmcrnsd: Processing disk sda3

mmcrnsd: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@i04n51-gpfs1 ~]#

[root@i04n51-gpfs1 ~]# cat /tmp/nsd
# DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool
# /dev/sda3:i04n51-gpfs1::dataAndMetadata:11::
gpfs1nsd:::dataAndMetadata:11::system
# /dev/sda3:i04n52-gpfs2::dataAndMetadata:12::
gpfs2nsd:::dataAndMetadata:12::system
[root@i04n51-gpfs1 ~]#

[root@i04n51-gpfs1 ~]# mmlsnsd

 File system   Disk name    NSD servers
---------------------------------------------------------------------------
 (free disk)   gpfs1nsd     i04n51-gpfs1
 (free disk)   gpfs2nsd     i04n52-gpfs2

[root@i04n51-gpfs1 ~]#
```

8. Before you can create a file system on the NSD disks that are defined in Example 3-36, the GPFS server cluster must be running on all nodes from which you are going to use an NSD disk, as shown in Example 3-37 on page 117.

Example 3-37 Starting up the GPFS cluster

```
[root@i04n51-gpfs1 ~]# mmstartup -a
Tue Oct 29 09:51:34 EDT 2013: mmstartup: Starting GPFS ...
[root@i04n51-gpfs1 ~]#

[root@i04n51-gpfs1 ~]# mmgetstate -a

 Node number  Node name        GPFS state
------------------------------------------
       1      i04n51-gpfs1     active
       2      i04n52-gpfs2     active
[root@i04n51-gpfs1 ~]#
```

9. Create the file system that is shared across client nodes, as shown in Example 3-38.

Example 3-38 Creating the GPFS file system

```
[root@i04n51-gpfs1 ~]# mmcrfs /gpfs1 /dev/gpfs1 -F /tmp/nsd -B 256k -n 100 -v no
-R 2 -M 2 -r 2 -m2

The following disks of gpfs1 will be formatted on node i04n51-gpfs1:
    gpfs1nsd: size 102400000 KB
    gpfs2nsd: size 102400000 KB
Formatting file system ...
Disks up to size 912 GB can be added to storage pool system.
Creating Inode File
Creating Allocation Maps
Creating Log Files
  80 % complete on Tue Oct 29 09:53:25 2013
 100 % complete on Tue Oct 29 09:53:26 2013
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
  52 % complete on Tue Oct 29 09:53:31 2013
 100 % complete on Tue Oct 29 09:53:36 2013
Completed creation of file system /dev/gpfs1.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@i04n51-gpfs1 ~]#


[root@i04n51-gpfs1 ~]# mmlsfs all

File system attributes for /dev/gpfs1:
======================================
flag                value                    description
------------------- ------------------------ -----------------------------------
 -f                 8192                     Minimum fragment size in bytes
 -i                 512                      Inode size in bytes
 -I                 16384                    Indirect block size in bytes
 -m                 2                        Default number of metadata replicas
 -M                 2                        Maximum number of metadata replicas
 -r                 2                        Default number of data replicas
 -R                 2                        Maximum number of data replicas
 -j                 cluster                  Block allocation type
 -D                 nfs4                     File locking semantics in effect
```

```
 -k                 all                      ACL semantics in effect
 -n                 100                      Estimated number of nodes that will
mount file system
 -B                 262144                   Block size
 -Q                 none                     Quotas enforced
                    none                     Default quotas enabled
 --filesetdf        No                       Fileset df enabled?
 -V                 13.23 (3.5.0.7)          File system version
 --create-time      Tue Oct 29 09:53:22 2013 File system creation time
 -u                 Yes                      Support for large LUNs?
 -z                 No                       Is DMAPI enabled?
 -L                 4194304                  Logfile size
 -E                 Yes                      Exact mtime mount option
 -S                 No                       Suppress atime mount option
 -K                 whenpossible             Strict replica allocation option
 --fastea           Yes                      Fast external attributes enabled?
 --inode-limit      200704                   Maximum number of inodes
 -P                 system                   Disk storage pools in file system
 -d                 gpfs1nsd;gpfs2nsd        Disks in file system
 --perfileset-quota no                       Per-fileset quota enforcement
 -A                 yes                      Automatic mount option
 -o                 none                     Additional mount options
 -T                 /gpfs1                   Default mount point
 --mount-priority   0                        Mount priority
[root@i04n51-gpfs1 ~]#
```

Note: From a performance perspective, it is recommended that you set the GPFS block size to match the application buffer size, the RAID stripe size, or a multiple of the RAID stripe size. If the GPFS block size does not match the RAID stripe size, performance can be severely degraded, especially for write operations. For more information about the **mmcrfs** command, see *IBM GPFS 3.5: Advanced Administration Guide*, SC23-5182-05.

10. Mount the file system on all NSD server nodes, as shown in Example 3-39.

Example 3-39 Mounting the file system

```
[root@i04n51-gpfs1 ~]# mmmount gpfs1 -a
Tue Oct 29 09:57:31 EDT 2013: mmmount: Mounting file systems ...
[root@i04n51-gpfs1 ~]#

[root@i04n51-gpfs1 ~]# df -h
Filesystem  SizeUsedAvailUse%Mounted on
/dev/mapper/rootvg-rootlv7.9G1.1G6.5G15%/
Tmpfs          12G012G0%/dev/shm
/dev/sda2   97M32M61M35%/boot
/dev/sda1   100M252K100M1%/boot/efi
/dev/gpfs1  196G1.1G195G1%/gpfs1
[root@i04n51-gpfs1 ~]#

[root@i04n51-gpfs1 ~]# mmlsmount all -L

File system gpfs1 is mounted on 2 nodes:
  129.40.64.51    i04n51-gpfs1
```

```
  129.40.64.52    i04n52-gpfs2
[root@i04n51-gpfs1 ~]#
```

The GPFS NSD server nodes now are fully configured and they can be used to add more client nodes to the GPFS cluster by using an automate process during compute nodes provisioning.

3.2.8 IBM Platform HPC high availability

In our cluster scenario, we plan for high availability of the management nodes and Platform LSF installation on the management nodes. The high availability can be achieved by adding a secondary management node, which can take over the services if the primary (initial) management node fails. In this case, we can say that the high availability mode is active-standby where the active management node provides the corresponding services and the standby management node takes over all the services when active node is down.

The primary management node is the active node after the initial high availability enablement. The configuration data of this node before high availability enablement becomes the configuration data of the high availability cluster. It also acts as the primary master node for workload management.

Secondary management node is the standby node after the initial high availability enablement and acts as a failover node that takes over the services and become active when the primary management node goes down (fails).

Shared resources

Because the standby node takes over if the active nodes goes down, it must access all of the configuration data of the downed node (OS distributions, kit components, profiles, provisioning templates, and so on). The standby node also must take over the IP address of the active node to provide the corresponding services to the compute nodes or the users.

The shared storage server is an external storage server that stores all of the configuration data of management nodes, the user HOME directories, and system working directories. Normally, this shared storage is an NFS server that can provide access for the management nodes and users to their resources.

Virtual IP is a floating IP that is always configured on the active management node to provide services to the users (for example, access to active Platform LSF queue) and compute nodes (for example, NTP and syslog).

High availability

High Availability of the management nodes is accomplished by using EGO service, which is also known as high availability (HA) manager. HA manager runs on the management nodes and monitors the health of local services. If some of the services become unavailable, they are restarted. HA manager monitors these services via HA service agents.

At the same time, the HA manager also is monitoring the management node peers through a heartbeat signal. If the active node is down or if the HA manager detects unrecoverable errors, it migrates all controlled services to the standby (failover) node.

When a high availability solution is built, the following constraints apply to the management nodes:

- The management nodes must have the same or similar hardware.
- The management nodes must have the same partitioning layout, the same network settings, and must use the same network interfaces to connect to the provision and public networks.
- The management nodes must be configured with the same time, time zone, and current date.
- The virtual IP address is in the IP range of the corresponding networks on the primary management node.

IBM Platform HPC supports the following methods for setting up a high available environment:

- High availability environment fresh set up
- High availability environment setup on an existing cluster

The high availability environment fresh setup is simple and can be performed by following the installation manual of IBM Platform HPC. In our example scenario, we did not configure the high availability from the scratch; therefore, we configure it at this stage. The procedure for building the high availability on an existing environment has some similarities with a fresh setup, but many things are different because the existing cluster must be reconfigured.

As always in an existing cluster, there are compute nodes running that depend on the cluster services that are running on the management node (for example, NFS server and NTP, syslog). The compute nodes are configured to ask for these services to the provisioning IP address of the management node. However, because this IP address is replaced by the virtual IP address, all of the compute nodes must be reconfigured or reprovisioned. Also, the IP address of the shared storage server is changed and the compute nodes must become aware.

The recommended method to use is to reprovision all the compute nodes.

Note: Setting up the Platform HPC environment temporarily affects the operation of the cluster. It must be scheduled to avoid affecting users of the environment.

To set up the high availability environment for an existing IBM Platform HPC cluster, you should use the steps that are described next.

Installing Platform HPC on the failover node

Before you set up the high availability environment, you must install IBM Platform HPC on the failover management node with the same version of IBM Platform HPC ISO or DVD, and use the same installation options.

It is assumed the following tasks are complete:

- IBM Platform HPC is installed on the primary management node and the high availability environment is not set up.
- The operating system is installed in the secondary (failover) management node with the same partitioning layout as the primary and IBM Platform HPC is not yet installed.

To ensure that the secondary management node is installed with the same options as the primary management node, you must collect the following information about primary management node:

- Partitions layout

  Run the **df -h** and **fdisk -l** commands on the primary and the secondary management nodes to check the partition layout and look for any inconsistency. If the failover node has a different partition layout, you should reinstall the operating system with the same partition layout.

  The `/install` mount point on the secondary management node must be identical to the primary management node. Run the **df -h /install** command on the primary management node (as shown in Example 3-40 on page 121) to see which partition is used for the `/install` directory. Use the same partition mount point for the depot (`/install`) directory when the failover management node is installed.

  In Example 3-40, the / is used for depot (`/install`) directory on the primary management node.

  Example 3-40 Checking partition information about the management nodes

```
[root@i04n50 ~]# df -h /install
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/rootvg-lv_install
                       40G  6.0G   32G  16% /install
[root@i04n50 ~]#

[root@i04n49 ~]# df -h /install
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/rootvg-lv_install
                       40G  176M   38G   1% /install
[root@i04n49 ~]#
```

Note: The */install* directory was its own partition before enabling HA. Space is required on `/install` to install pHPC. After HA is enabled, `/install` is renamed to `/install.PCMHA`. pHPC had replicated files under `/install` that are needed from `/install` (which is now `/install.pHPC`) to the shared storage and created the link `/install` → `/shared_phpc/install`.

- Time and time zone

  Reset the current date and time on the failover management node to ensure that the two management nodes have the same date and time, as shown in Example 3-41.

  Example 3-41 Reset the day and time on the failover management node

```
[root@i04n49 ~]# date -s '20131125 10:31:00'
```

  Collect the time zone that is defined on the primary management node and align both management nodes to the same time zone, as shown in Example 3-42.

  Example 3-42 Collecting the time zone

```
[root@i04n50 ~]# lsdef -t site -o clustersite -i timezone
Object name: clustersite
    timezone=America/New_York
[root@i04n50 ~]#
```

If the time zone is different on the secondary node, set up the time zone as shown in Example 3-43.

Example 3-43 Setting up similar time zones on the management nodes

```
[root@i04n50 ~]# echo 'ZONE="America/New_York"' > /etc/sysconfig/clock
[root@i04n50 ~]# ln –s /usr/share/zoneinfo/US/Eastern /etc/localtime
```

Note: On the SLES operating system, `TIMEZONE` is used instead of `ZONE`.

- Network settings

  Complete the following steps to check and collect the network setting information:

  a. Ensure that the failover and virtual IP addresses are not used by the compute nodes, as shown in Example 3-44.

  Example 3-44 Checking for duplicate IP addresses

  ```
  [root@i04n50 ~]# tabdump nics|grep 129.40.64.49
  [root@i04n50 ~]# tabdump nics|grep 129.40.64.100
  [root@i04n50 ~]# tabdump nics|grep 129.50.64.49
  [root@i04n50 ~]# tabdump nics|grep 129.50.64.100
  ```

  b. Ensure that the failover and virtual IP addresses are in the static IP address range, as shown in Example 3-45.

  Example 3-45 Checking the IP addresses and their static IP address ranges

  ```
  [root@i04n50 ~]# lsdef -t network -i staticrange,dynamicrange
  Object name: Default_BMC_FSP_Network
      dynamicrange=
      staticrange=129.40.65.51-129.40.65.92
  Object name: ISV_Application_Network
      dynamicrange=
      staticrange=129.40.66.51-129.40.66.92
  Object name: provision
      dynamicrange=129.40.64.230-129.40.64.254
      staticrange=129.40.64.51-129.40.64.92
  [root@i04n50 ~]#
  ```

  In Example 3-46, the public virtual IP address is not in the IP range. This can be changed by changing the definition of the network object.

  Example 3-46 Changing the network object definition

  ```
  [root@i04n50 ~]# chdef -t network -o provision
  staticrange=129.40.64.49-129.40.64.100
  [root@i04n50 ~]# chdef -t network -o public
  staticrange=129.50.64.49-129.50.64.100
  ```

  Note: Ensure that the IP address range does not conflict with dynamic IP address range.

  c. Ensure that the same network interfaces are used on the failover node to access the provision and public networks, as shown in Example 3-47 on page 123.

Example 3-47 Checking the network interfaces

```
[root@i04n50 ~]# lsdef i04n50 | grep nicips
nicips.eth0=129.40.64.50
nicips.eth1=129.50.64.50
[root@i04n50 ~]#

[root@i04n50 ~]# ping –c 2 –I eth0 129.40.64.49
[root@i04n50 ~]# ping –c 2 –I eth1 129.50.64.49
```

To get more installation options of the primary management node that is used as a reference when a secondary management node is installed, review the installation log file of IBM Platform HPC: `/opt/pcm/log/phpc-installer.log`.

After you collect all installation options on the primary management node and ensure that the failover node is consistent with the primary management node, run the IBM Platform HPC installer `phpc-installer` as user root. By using the installer, you can specify your installation options; however, ensure that you use the same installation options that were collected in the failover management node configuration.

For more information about installing IBM Platform HPC on the secondary management node, see 3.2.6, “Cluster deployment” on page 75.

Installing the high availability environment

If in your cluster you do not want to reprovision the compute nodes, complete the following steps before you set up high availability:

1. Shut down the entire Platform LSF cluster, as shown in Example 3-48.

Example 3-48 Shutting down Platform LSF

```
[root@i04n50 ~]# lsfshutdown –f
Shutting down all slave batch daemons ...

Shut down slave batch daemon on all the hosts? [y/n] y
Shut down slave batch daemon on <i04n50> ...... done
Shut down slave batch daemon on <i04n51-gpfs1> ...... Host control failed:
Connection refused by server
Shut down slave batch daemon on <i04n52-gpfs2> ...... Host control failed:
Connection refused by server
Shut down slave batch daemon on <i04n53> ...... done
Shut down slave batch daemon on <i04n54> ...... done
Shutting down all RESes ...
Do you really want to shut down RES on all hosts? [y/n] y
Shut down RES on <i04n51-gpfs1> ...... failed: A connect sys call failed:
Connection refused
Shut down RES on <i04n52-gpfs2> ...... failed: A connect sys call failed:
Connection refused
Shut down RES on <i04n50> ...... done
Shut down RES on <i04n53> ...... done
Shut down RES on <i04n54> ...... done

Shutting down all LIMs ...
Do you really want to shut down LIMs on all hosts? [y/n] y
Shut down LIM on <i04n50> ...... done
Shut down LIM on <i04n53> ...... done
Shut down LIM on <i04n54> ...... done
```

```
Trying unavailable hosts :

Shut down LIM on <i04n51-gpfs1> ...... ls_limcontrol: Communication time out
Shut down LIM on <i04n52-gpfs2> ...... ls_limcontrol: Communication time out
[root@i04n50 ~]#
```

2. Unmount and remove the `/home` and `/share` mount on the compute nodes, as shown in Example 3-49.

Example 3-49 Removing /home and /share

```
[root@i04n50 ~]# updatenode __Managed 'mountnfs del'
i04n52-gpfs2: Wed Dec 18 14:56:36 EST 2013 Running postscript: mountnfs del
i04n54: Wed Dec 18 14:56:36 EST 2013 Running postscript: mountnfs del
i04n53: Wed Dec 18 14:56:36 EST 2013 Running postscript: mountnfs del
i04n51-gpfs1: Wed Dec 18 14:56:36 EST 2013 Running postscript: mountnfs del
i04n52-gpfs2: Postscript: mountnfs del exited with code 0
i04n52-gpfs2: Running of postscripts has completed.
i04n54: Postscript: mountnfs del exited with code 0
i04n54: Running of postscripts has completed.
i04n51-gpfs1: Postscript: mountnfs del exited with code 0
i04n51-gpfs1: Running of postscripts has completed.
i04n53: Postscript: mountnfs del exited with code 0
i04n53: Running of postscripts has completed.
. . .
[root@i04n50 ~]#
```

3. Add the shared storage server as an unmanaged node to the IBM Platform HPC cluster, as shown in Example 3-50.

Example 3-50 Adding the shared storage server

```
[root@i04n50 ~]# nodeaddunmged hostname=nfsserver ip=129.40.64.210
Created unmanaged node.
[root@i04n50 ~]#
```

This operation prevents the IP of the shared storage server (NFS server) from being allocated to a compute nodes. It also ensures that the NFS server name can be resolved consistently across the cluster.

4. Add the failover node entry to the `/etc/hosts` file on the primary management node, as shown in Example 3-51.

Example 3-51 Adding the failover node entry to /etc/hosts

```
[root@i04n50 ~]# echo "129.40.64.49 i04n49 i04n49.pbm.ihost.com i04n49-eth0" >>
/etc/hosts
[root@i04n50 ~]# ping i04n49
```

5. Configure the password-less SSH connection between management nodes, as shown Example 3-52 on page 125.

Example 3-52 SSH configuration on the management nodes

```
[root@i04n50 ~]# cat /root/.ssh/id_rsa.pub > /root/.ssh/authorized_keys
[root@i04n50 ~]# ssh i04n49  “cp –rf /root/.ssh /root/.ssh.PCMHA”
root@i04n49's password:
[root@i04n50 ~]# scp –r  /root/.ssh/*  i04n49:/root/.ssh
root@i04n49's password:
authorized_keys                100%  393     0.4KB/s   00:00
config                         100%   25     0.0KB/s   00:00
id_rsa                         100% 1675     1.6KB/s   00:00
id_rsa.pub                     100%  393     0.4KB/s   00:00
known_hosts                    100% 2394     2.3KB/s   00:00
[root@i04n50 ~]# ssh i04n49 uptime
05:06:53 up 4 days,  1:03,  2 users,  load average: 0.12, 0.11, 0.09
```

6. Verify that the shared storage server is available, as shown in Example 3-53.

Example 3-53 Verifying the availability of the shared storage server

```
[root@i04n50 ~]# showmount -e 129.40.64.210
Export list for 129.40.64.210:
/gpfs/fs21/cluster4/phpc4111/shared 129.40.66.0/24,129.40.64.0/24
/gpfs/fs21/cluster4/phpc4111/home   129.40.66.0/24,129.40.64.0/24
```

Configuring high availability

At this stage, the following assumptions are made:

- IBM Platform HPC was installed successfully on both management nodes and the high availability environment is not enabled.
- The password-less SSH connection is setup between the primary management node and the failover node.
- Shared file systems for high availability were created on the shared storage server and have sufficient free capacity.

The high availability environment can be enabled by using the HA management tool (**pcmhatool**). This tool defines the HA environment by using the predefined HA definition file.

Define a HA definition file according to your high availability environment configuration. The HA definition file example `ha.info.example` is in the `/opt/pcm/share/examples/HA` directory. For our scenario, we created the HA definitions file `/root/ha.info`, as shown in Example 3-54.

Example 3-54 HA definitions file

```
virtualmn:
   nicips.eth1:0=129.50.64.100
   nicips.eth0:0=129.40.64.100
   sharefs_mntp.work=129.40.64.210:/gpfs/fs21/cluster4/phpc4111/shared
   sharefs_mntp.home=129.40.64.210:/gpfs/fs21/cluster4/phpc4111/home
```

Note: You must use the IP address of the shared storage server that is connected to the provision network for the shared file systems.

Enable HA by using the following command:

```
# pcmhatool config -i ha-definition-file-path  -s failover-node-name
```

Note: The HA management tool supports the Bash shell only.

After the enablement starts, the tool automatically parses your HA definition file and checks the hardware and software configurations. The tool displays the following information that is based on the results:

- OK: Indicates that no problems are found for the checked item.
- WARNING: Indicates that the configuration of an item does not match the requirements; HA enablement continues despite the warnings.
- FAILED: The HA enablement quits if the tool cannot recover from an error.

Note: It takes approximately 30 - 90 minutes to set up high availability because of the synchronization process between the primary management node and the shared storage server.

Example 3-55 shows setting up high availability between nodes `i04n50` and `i04n49`.

Example 3-55 Setting up high availability

```
[root@i04n50 HA]# pcmhatool config -i ha.info -s i04n49
Setting up high availability between the management node 'i04n50' and the standby
management node 'i04n49'. Do not perform any operations during this time.

   Parsing high availability data...                        [  OK  ]
   ================================================================
   Virtual node name: virtualmn
   Virtual IP address: eth0:0; 129.40.64.100
   Shared work directory on: 129.40.64.210:/gpfs/fs21/cluster4/phpc4111/shared
   Shared home directory on: 129.40.64.210:/gpfs/fs21/cluster4/phpc4111/home
   ================================================================

   During the high availability setup, data is copied to shared directories
   and IBM Platform HPC services are restarted several times.
Ensure that the external shared storage server and network connections are
available.

   Continue? (Y/N) [N]: Y


   Checking if passwordless SSH is available...             [  OK  ]
   Getting standby management node data...                  [  OK  ]
   Checking if the secondary management node is
   consistent the management node...                        [  OK  ]
   Checking if the virtual network interface <eth0:0>
   is available...                                          [  OK  ]
   Checking if the virtual IP address <129.40.64.100>
   is available...                                          [  OK  ]
   Checking if the shared directory home is available...    [  OK  ]
   Checking if the shared directory work is available...    [  OK  ]
   Stopping services...                                     [  OK  ]
   Synchronizing system work data. This can take
   several minutes...                                       [  OK  ]
   Synchronizing user home data. This can take
   several minutes...                                       [  OK  ]
```

```
   Starting the xCAT daemon...                              [  OK  ]
   Saving high availability data...                         [  OK  ]

   Starting high availability configuration on the management node...
   Setting up hosts/resolv files:                           [  OK  ]
   Setting up dhcpd:                                        [  OK  ]
   Setting up named:                                        [  OK  ]
   Setting up shared file system for HA:                    [  OK  ]
   Setting up ntpd:                                         [  OK  ]
   Setting up LSF configuration:                            [  OK  ]
   Setting up web portal:                                   [  OK  ]
   Setting up Platform HPC configuration:                   [  OK  ]
   Updating management node:                                [  OK  ]

   Configured high availability on the management node.
   Generating the high availability data file...            [  OK  ]

   Starting high availability configuration on the standby management node...
   High availability configured on the standby management node.
   Starting IBM Platform HPC services...                    [  OK  ]
   Starting services on the failover node...                [  OK  ]

High availability setup is complete. The setup log file pcmhatool.log is located
in the /opt/pcm/log directory.

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 Platform HPC 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.
[root@i04n50 HA]#
```

Note: If the management node crashes during the HA environment set up process, you should run the command again to clean up the incomplete environment and then restart the HA setup environment.

You can find the log file `pcmhatool.log` in `/opt/pcm/log` directory. This log file includes details and results about the HA environment setting up.

Verifying the high availability environment

The high availability status can be seen on the Web Portal or the CLI.

Showing HA status on WEN portal

Log in to the Platform HPC Web Portal with the management node virtual IP address. After you log in, the Resource Dashboard is displayed in the Web Portal. You can see two management node statuses in the Cluster Health panel. The current role (management node or failover node) also is shown behind the node name, as shown in Figure 3-26.

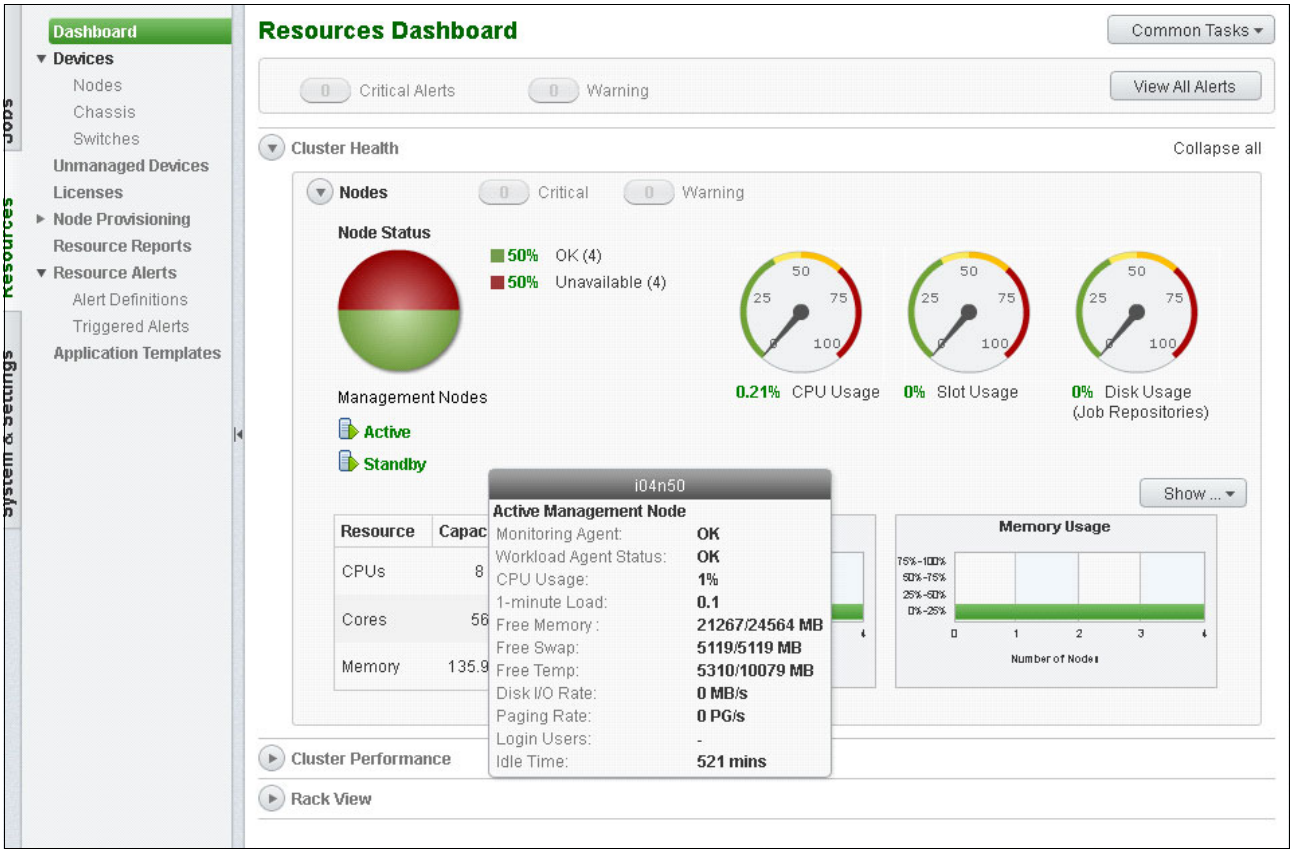

Figure 3-26 WEN Portal Resource Dashboard

The management node is the current active node where all services are running. The failover node is the node that is running in standby mode.

Showing HA status on the CLI

Displaying the HA status displays the current high availability status, including the state of the nodes, failover mode, and the status of running services, as shown in Example 3-56.

Example 3-56 Showing high availability status

```
[root@i04n50 ~]# pcmhatool status
Primary management node name: i04n50 [ok]
Secondary management node name: i04n49 [ok]
Failover mode: Automatic
Cluster management services running on: i04n50
Workload management services running on: i04n50
[root@i04n50 ~]#
```

To display the HA status for the HA manager and service agents, run the **service phpc status** command. All services must be in the STARTED state, as shown in Example 3-57 on page 129.

Example 3-57 Displaying the HA manager and service agent statuses

```
[root@i04n50 ~]# service phpc status

Show status of the LSF subsystem
lim (pid 3410) is running...
res (pid 3412) is running...
sbatchd (pid 3415) is running... [  OK  ]

SERVICE  STATE    ALLOC CONSUMER RGROUP RESOURCE SLOTS SEQ_NO INST_STATE ACTI
JOBDT    STARTED  9     /Manage* Manag* i04n50   1     1      RUN         115
PLC2     STARTED  6     /Manage* Manag* i04n50   1     1      RUN         116
WEBGUI   STARTED  4     /Manage* Manag* i04n50   1     1      RUN         117
PURGER   STARTED  8     /Manage* Manag* i04n50   1     1      RUN         118
PTC      STARTED  7     /Manage* Manag* i04n50   1     1      RUN         119
PLC      STARTED  5     /Manage* Manag* i04n50   1     1      RUN         120
XCAT     STARTED  3     /Manage* Manag* i04n50   1     1      RUN         112
PCMHA    STARTED  1     /Manage* Manag* i04n50   1     1      RUN         113
PCMDB    STARTED  2     /Manage* Manag* i04n50   1     1      RUN         114
[root@i04n50 ~]#
```

Run the **pcmhatool info** command to displays high availability settings, including the virtual IP address, management node name, and a list of shared directories, as shown in Example 3-58.

Example 3-58 Displaying high availability settings

```
[root@i04n50 ~]# pcmhatool info
   Configuring status: OK
   ================================================================
   High availability group members: i04n50, i04n49
   Virtual node name: virtualmn
   Virtual IP address: eth0:0; 129.40.64.100
   Shared work directory on: 129.40.64.210:/gpfs/fs21/cluster4/phpc4111/shared
   Shared home directory on: 129.40.64.210:/gpfs/fs21/cluster4/phpc4111/home
[root@i04n50 ~]#
```

Manual failover test

Example 3-59 shows how to perform a manual failover test.

Example 3-59 Performing a manual failover test

```
[root@i04n50 ~]# pcmhatool failto -t i04n49
   Migrating IBM Platform HPC services from 'i04n50' to 'i04n49'.
   During this process, do not perform any operations.

   During the failover process, services are restarted and any service operation
attempt can fail. If an operation fails, retry the operation after the failover
process is complete.

   Continue? (Y/N) [N]: y

   Starting manual failover...                              [  OK  ]
   Stopping services running on i04n50...                   [  OK  ]
   Cluster management services are now running on i04n49.
[root@i04n50 ~]#
```

```
[root@i04n50 ~]# service phpc status

Show status of the LSF subsystem
lim (pid 3410) is running...
res (pid 3412) is running...
sbatchd (pid 3415) is running...
                                                          [  OK  ]
SERVICE  STATE    ALLOC CONSUMER RGROUP RESOURCE SLOTS SEQ_NO INST_STATE ACTI
JOBDT    STARTED  9     /Manage* Manag* i04n49   1     1      RUN         126
PLC2     STARTED  6     /Manage* Manag* i04n49   1     1      RUN         125
WEBGUI   STARTED  4     /Manage* Manag* i04n49   1     1      RUN         124
PURGER   STARTED  8     /Manage* Manag* i04n49   1     1      RUN         123
PTC      STARTED  7     /Manage* Manag* i04n49   1     1      RUN         122
PLC      STARTED  5     /Manage* Manag* i04n49   1     1      RUN         121
XCAT     STARTED  3     /Manage* Manag* i04n49   1     1      RUN         129
PCMHA    STARTED  1     /Manage* Manag* i04n49   1     1      RUN         128
PCMDB    STARTED  2     /Manage* Manag* i04n49   1     1      RUN         127
[root@i04n50 ~]#
bsub
```

With the successful completion of `pcmhatool failto`, the node `i04n49` is now the active management node, and `i04n50` is the standby management node. This status can be confirmed by using the **`pcmhatool status`** and **`service phpc status`** tools.

3.2.9 Running applications in Platform HPC

Implementing a cluster with IBM Platform HPC 4.1.1.1 provides users with an integrated environment to run jobs that are scheduled by IBM Platform LSF Express Edition 9.1.1.0 on the cluster nodes. Users can submit and monitor the jobs in the pHPC web portal or by using the CLI of Platform LSF.

Upgrading Platform LSF

When it is necessary to upgrade Platform LSF Express Edition to Platform LSF Standard Edition, or to apply fixes to Platform LSF to run applications, the integrated environment of pHPC is preserved for any Platform LSF 9.1.1.x updates. Because Platform LSF is installed on the shared directory that is mounted across the cluster nodes, changes to the Platform LSF installation must be made on the workload management node only.

To upgrade the entitlement of Platform LSF from Express Edition to Standard Edition, as root, replace the LSF_Express_HPC entitlement by LSF_Standard entitlement in the entitlement file `/opt/pcm/entitlement/phpc.entitlement` and restart Platform LSF by running the **`lsfrestart`** command, as shown in Example 3-60 and Example 3-61 on page 131.

Example 3-60 LSF_EXPRESS_HPC entitlement

```
[root@i04n50 ~]# lsid
IBM Platform LSF Express 9.1.1.0 for IBM Platform HPC, Feb 27 2013
Copyright International Business Machines Corp, 1992-2013.
US Government Users Restricted Rights - Use, duplication or disclosure restricted
by GSA ADP Schedule Contract with IBM Corp.
My cluster name is phpc_cluster
My master name is i04n50
```

Example 3-61 LSF_Standard entitlement

```
root@i04n50 ~]# lsid
IBM Platform LSF Standard 9.1.1.0, Feb 27 2013
Copyright International Business Machines Corp, 1992-2013.
US Government Users Restricted Rights - Use, duplication or disclosure restricted
by GSA ADP Schedule Contract with IBM Corp.

My cluster name is phpc_cluster
My master name is i04n50
```

Platform LSF fixes are available for download from the IBM Fix Central website and are applied to the workload management node by following the instructions that are included within the fix `.tar` file. The Fix Central is available at:

https://www-933.ibm.com/support/fixcentral/

4

IBM Big Data implementation on an IBM High Performance Computing cluster

Big Data technologies describe a new generation of technologies and architectures that are designed to economically extract value from large volumes of various data by enabling high velocity capture, discovery, or analysis.

In today's environment, Big Data environments are complex and adding challenges to the already cumbersome company technology structures. IBM high performance computing (HPC) and IBM Platform Symphony provide reference architectures for companies that are seeking a cross reference of data for specific analytics.

The output of Big Data analysis can generate key insights to improve client experience, focal data for marketing, operations, and finances, among others.

This chapter includes the following topics:

- IBM high performance computing for Big Data analytics reference architectures
- High performance low latency Big Data solutions stack by using PCM-SE for a Platform Symphony MapReduce cluster

4.1 IBM high performance computing for Big Data analytics reference architectures

In this section, a reference architecture in the field or enterprise architecture seeks to provide a template solution for a Big Data architecture for use in multiple domains. In this section, we describe the structures, elements, and relations to provide a concrete architecture in the family of enterprise architecture.

With the combination of complex systems and high performance data processing comes the HPC Big Data reference architecture, which helps analyze data and shows IT architects how to implement this architecture in their analytics workloads.

HPC Big Data contains multiple layers of existing solutions that, can provide high performance for accelerated data processing when integrated. In this scenario, we divide the Big Data Implementation into the following layers:

- Visual analytics layer
- Data reduction layer
- Data model layer
- Data and modeling integration layer

The architecture within these four layers can be seen as a group of non-structured data, which was filtered across multiple systems to generate a comprehensive data analytic or transaction data-generated output.

Figure 4-1 shows the different aspects that compose the IBM HPC Big Data reference architecture.

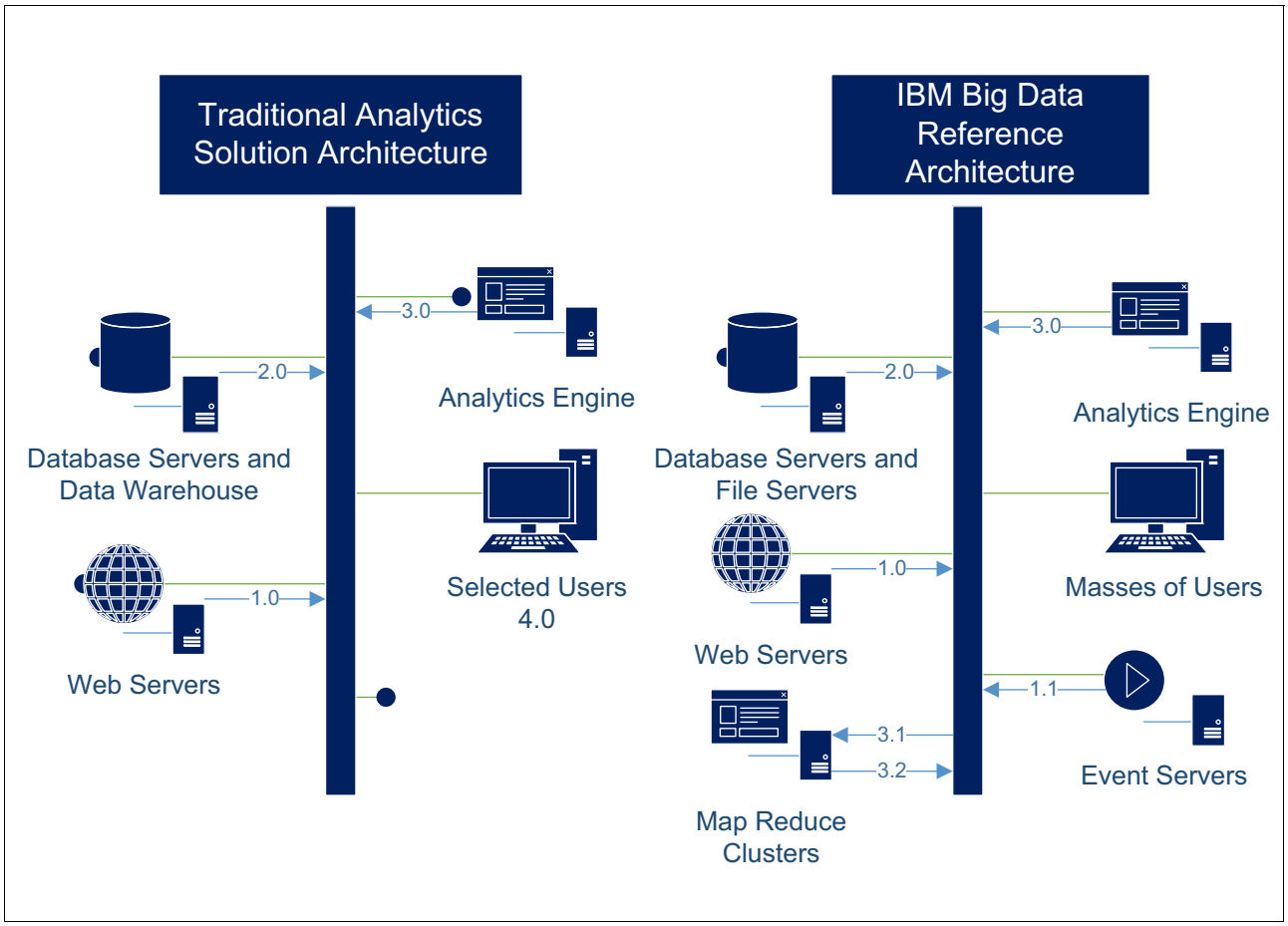

Figure 4-1 Reference architecture

Within the elements that are involved in the reference architecture, there are key components with which the data sets can flow. In this reference architecture, the configuration is designed for data sets so large that they become awkward to work with by using database management tools alone.

Some issues companies encounter today are data retention, data storage, share analytics, visualization, and search. When a Big Data architecture is built, the following aspects should not be ignored when a Big Data environment is implemented:

- Speed

  Typically, you consider speed as the time an information data set travels from point A to point B. However, in today's world, the word *speed* also is used to describe the speed at which data is flowing. As the world becomes smarter, sets of data come from various sources of information that share data to different sources for concatenation and storage usage. At this pace, there are more artifacts that generate data, which makes it almost impossible to analyze all of the data with an expectation to find insights in this generated sets of information. When you deal with Big Data, you are required to run analytics against a large volume of data diversity while this data is still in motion, and not when it lands at its final destination (point B).

- Data diversity

  Data diversity represents the different types of data that is generated from RFID sensors to social media user-generated content to smart devices. All of the data that is generated by all of these devices has structure or no structure data that makes it even more complex to analyze and generate specific output. With data generation coming from multiple sources, websites, log files, indexing, documents, email, sensors, and so on, Big Data provides companies with an opportunity to analyze all sorts of data, both relational and non-relational.

- Volume

  Volume is the most important aspect in a Big Data environment. Volume makes reference to the amount of data that was addressed and with the number of data handling incrementing at a fascinating pace. The world is expected to handle almost 30 zettabytes (ZB) by 2020. By using Big Data, organizations face the analysis of massive data volumes. Therefore, the ability to analyze data is one of the biggest challenges companies face today.

Big Data analytics is an advanced application that is used to process and run analytics in large data sets. Advance analytics are not composed by one single application, but multiple tools and collection techniques that can include data mining, predictive analytics, big SQL, data visualization, business intelligence, artificial intelligence, processing language, databases, and various methods to support analytics, such as Platform Symphony MapReduce, in-memory database, and columnar data stores.

The massive Big Data analytics that require performance and scalability are the most common problems that traditional platforms encounter when they are responding to large-scale data sets. In this aspect of analyzing data, we encounter two main consolidated methods: store-analyze or analyze-store.

In our architecture, Hadoop (with the use of IBM GPFS and IBM Platform Symphony) are some of the key elements that are used to run the Platform Symphony MapReduce in an analyze-store architecture. Hadoop enables a computing solution, such as high-performance data clustering to scale. You can add new nodes as needed and add them without the need to change data formats, how the data is loaded, jobs are written, or what applications are on top. The ability of Hadoop is to absorb any type of data that is structured or non-structured for large numbers of data sources.

Platform Symphony MapReduce divides the input data set into independent tasks. These tasks are then divided into map tasks in a parallel manner.

As shown in Figure 4-2 on page 137, we can visualize the framework sorting the outputs of a map in which the input and output of the jobs are stored in a file system. The framework takes care of the task scheduling along with the monitoring and running of the tasks.

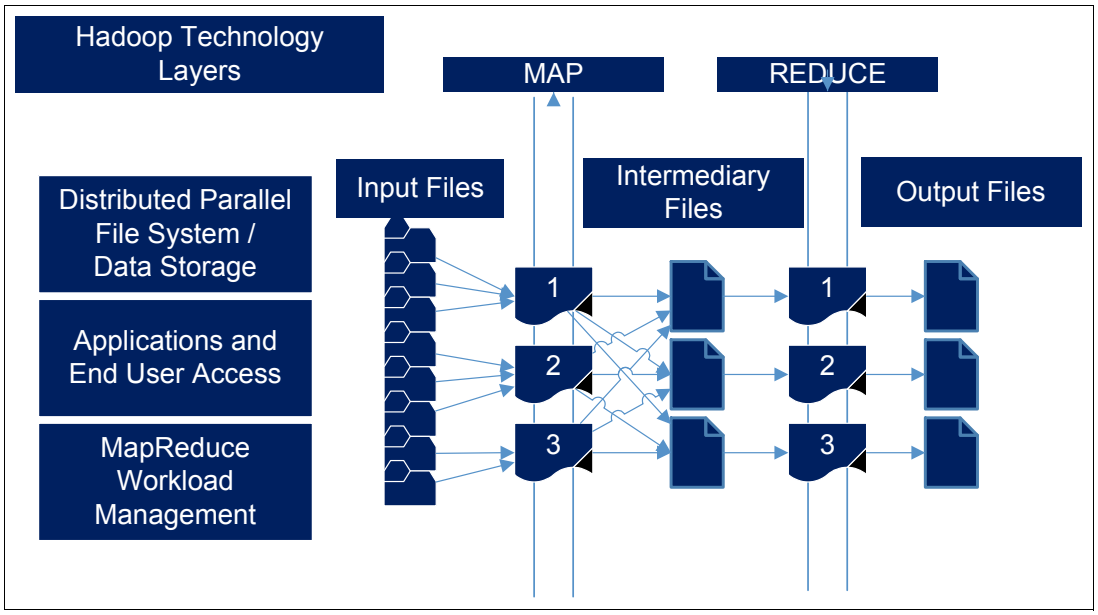

Figure 4-2 Platform Symphony MapReduce

It is important to highlight the add-ons Hadoop uses, such as Pig and Hive, as shown in the overall reference architecture (see Figure 4-1 on page 135).

For Big Data analytics, there are several approaches that we use to define the final architecture. Some of the several aspects to consider are direct analytics on big parallel processing, analytics over Hadoop, or indirect analytics over Hadoop. The key function to select is the Platform Symphony MapReduce programming model, in our case, we take the approach of three dimensional considering volume, data diversity, and speed, and consider that the technology can support batch and real-time processing.

The use of GPFS-FPO on top of a high performance cluster provides the ability to use adaptive Platform Symphony MapReduce with Platform Symphony to extend Hadoop, which creates individual maps that are aware of other maps. This configuration creates an environment efficiency for making better decisions, as shown in Figure 4-3.

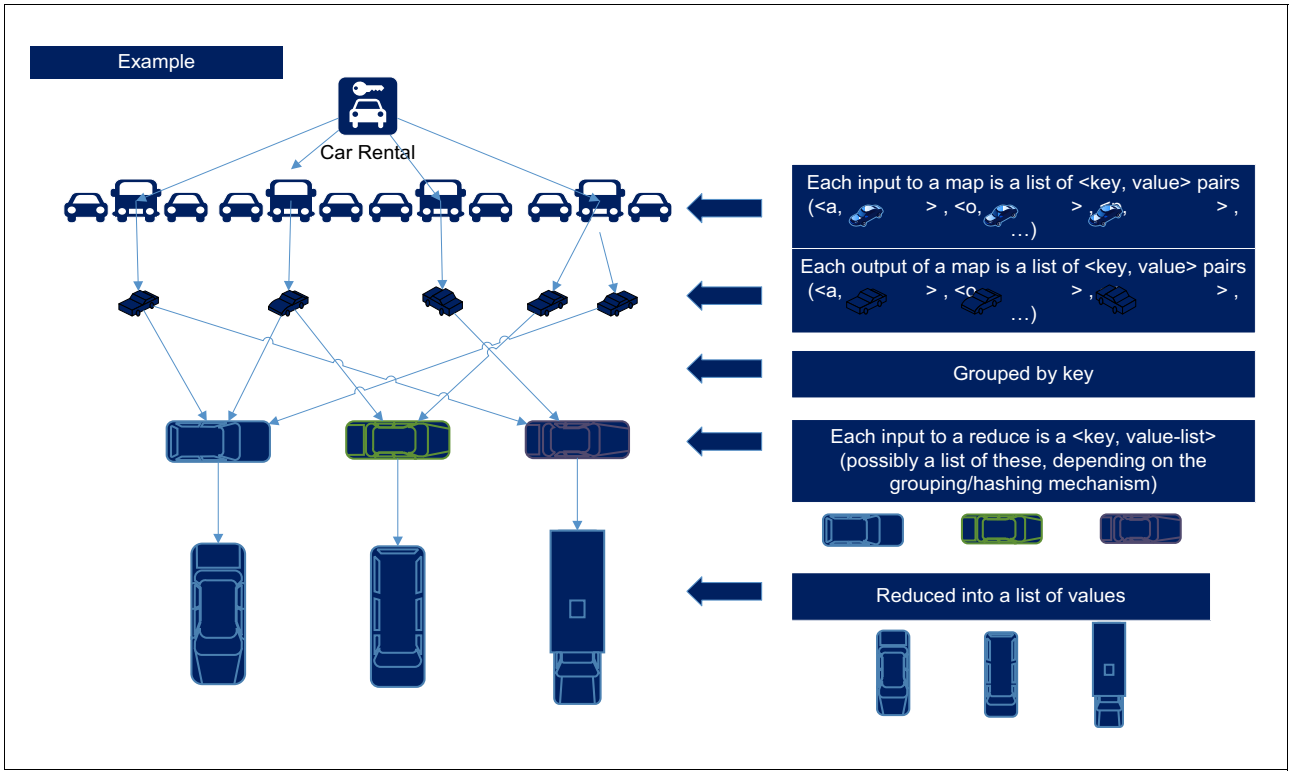

Figure 4-3 Platform Symphony MapReduce programming model example

The overall comparison of traditional analytics versus an IBM HPC Big Data environment (see Figure 4-4) helps clients to support massive amounts of data processing and create a new data combination and synergy to analyze and compare in real time the data that is processed.

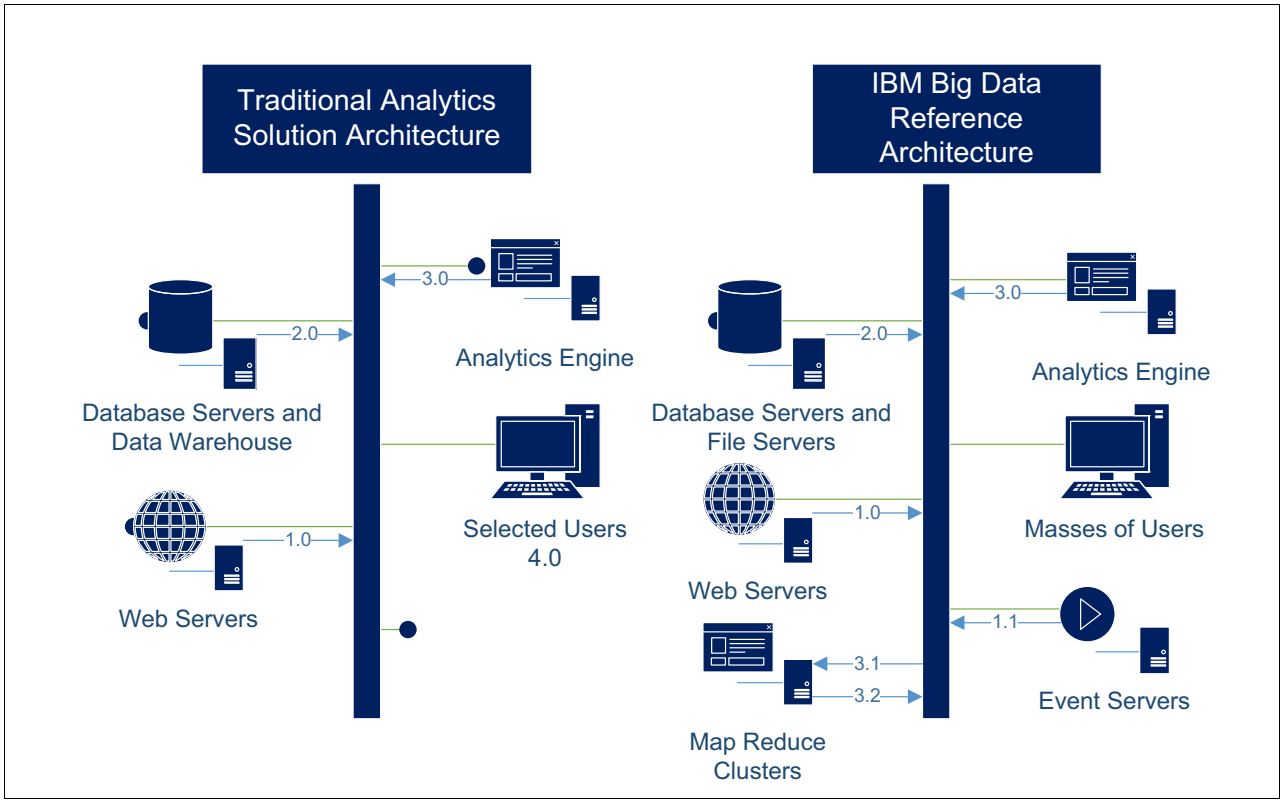

Figure 4-4 Traditional analytics architecture versus IBM Big Data architecture

4.2 High performance low latency Big Data solutions stack by using PCM-SE for a Platform Symphony MapReduce cluster

In this section, a hardware reference architecture in the field of infrastructure architecture seeks to provide a demonstration solution for a Big Data architecture for multiple domains of use. This section consists on structures, elements, and relations to provide a concrete architecture in the family of infrastructure architecture, as shown in Figure 4-5.

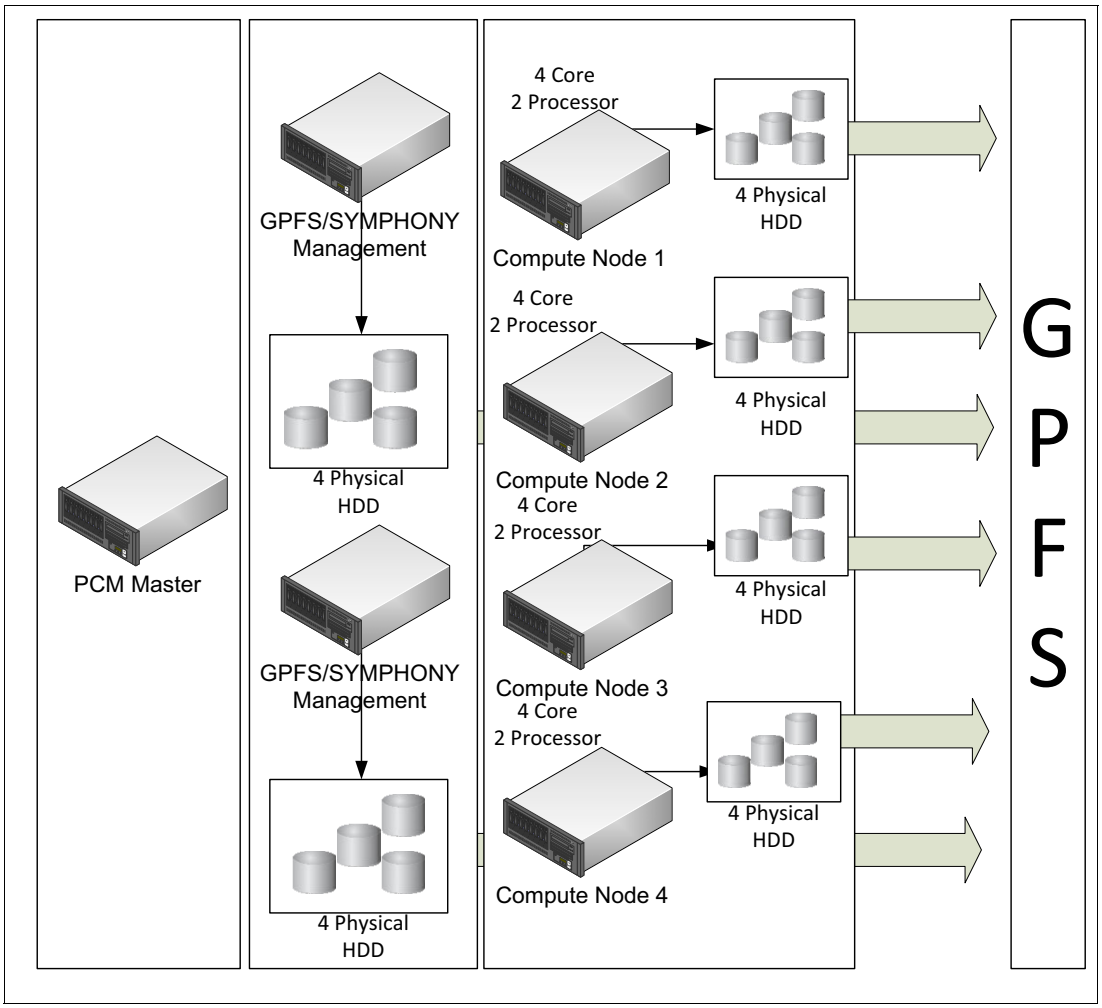

Figure 4-5 IBM Big Data solution stack

4.2.1 Installing IBM Platform Cluster Manager Standard Edition

In this installation process, we assume that you have the operating system installed. Before you begin your installation, ensure that the following preinstallation tasks are completed:

1. Review the following Platform Cluster Manager requirements:
   - Check for the minimum hardware requirements
   - Check for the minimum software requirement

2. Configure and test switches. Ensure that the necessary switches are configured to work with Platform Cluster Manager.
3. Plan your network configuration, including the following components:
   - Provision network information
   - Public network information
   - BMC network information
4. Obtain a copy of IBM Platform Cluster Manager Standard Edition.

   If you do not have a copy of IBM Platform Cluster Manager Standard Edition, you can download it from IBM Passport Advantage®.

Note: For more information about installing PCM-SE product, see the `pcmse_install.pdf` file that is in the `/docs` directory of the product media.

Complete the following steps to complete the IBM Platform Cluster Manager Standard Edition (PCM-SE) product installation:

1. Mount the Platform Cluster Manager installation media.

   If you install Platform Cluster Manager from the ISO file, mount the ISO into a directory, such as `/mnt`, as shown in Example 4-1.

   Example 4-1 Mount command for the product ISO file

```
# mount -o loop pcmse-4.1.1.x64.rhel.iso /mnt
```

   If you install Platform Cluster Manager from DVD media, mount to a directory, such as `/mnt`.
2. After mounting the PCM-SE product media, run the following command to start the installation process (as shown in Example 4-2):

   `./pcm-installer`

Example 4-2 The pcm-installer command output following the PCM-SE installation process

```
[root@mgt01 ~]# /mnt/pcm-installer
   Preparing to install 'pcm-installer'...[  OK  ]
   Finding the product entitlement file...[  OK  ]

   ================================================================
   Welcome to the IBM Platform Cluster Manager 4.1 Installation
   ================================================================

   The complete IBM Platform Cluster Manager 4.1 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.
```

3. Accept the license agreement and continue, as shown in Example 4-3.

Example 4-3 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.
```

The management node pre-checking process automatically starts, as shown in Example 4-4.

Example 4-4 Management node pre-checking process

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

Checking hardware architecture...[  OK  ]
Checking OS compatibility...[  OK  ]
Checking free memory...[  OK  ]
Checking if SELinux is disabled...[  OK  ]
Checking if Auto Update is disabled...[  OK  ]
Checking if NetworkManager is disabled...[  OK  ]
Checking if PostgreSQL is disabled...[  OK  ]
Checking for DNS service...[  OK  ]
Checking for DHCP service...[  OK  ]
Checking management node name...[  OK  ]
Checking static NIC...[  OK  ]
Probing DNS settings...[  OK  ]
Probing language and locale settings...[  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...[  OK  ]
```

Check all product installation prerequisites before you move on to the next step. In this part of the installation process, you are required to enter configuration parameters to complete the process.

The following installation mode options available:

- Quick installation
- Custom installation

For more information about the installation option comparison table, see the PCM-SE installation guide, which is available at this website:

http://publibfp.dhe.ibm.com/epubs/pdf/c2761070.pdf

Select the custom installation mode, as shown in Example 4-5.

Example 4-5 Custom installation option that is selected

```
   ================================================================
   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
```

4. Select a mount point for the depot (`/install`) directory, as shown in Example 4-6. The depot (`/install`) directory stores installation files for Platform Cluster Manager. The Platform Cluster Manager management node checks for the required disk space.

Example 4-6 Product media mount point and location selection

```
Found valid mount point for depot (/install) directory.

   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 ISO image or mount point:

   /home/RHEL6.4-20130130.0-Server-x86_64-DVD1.iso
```

5. Select the location from where you want to install the operating system. The operating system version that you select must be the same as the operating system version on the management node.

6. Select a network interface for the provisioning network, as shown in Example 4-7.

Example 4-7 Network setup sequence for the custom installation method

```
Select a network interface for the provisioning network from the following
options:
   1) Interface: eth0, IP: 10.10.1.8, Netmask: 255.255.255.0
   2) Interface: eth0.800, IP: 9.8.227.30, Netmask: 255.255.255.0
   Enter your selection [1]:  1

Enter IP address range used for provisioning compute nodes[10.10.1.3-10.10.1.200]:
10.10.1.21-10.10.1.99

Do you want to provision compute nodes with node discovery? (Y/N) [Y]: 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. [10.10.1.201-10.10.1.254]: 10.10.1.101-10.10.1.2 199

The management node is connected to the public network by:

   1) Interface: eth0.800, IP: 9.8.227.30, Netmask: 255.255.255.0
   2) It is not connected to the public network

   Enter your selection [1]:  1

Enable Platform Cluster Manager specific rules for the management node firewall to
the public interface? (Y/N) [Y]: n Y

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

7. Enter the IP address range that is used for provisioning compute nodes.
8. Choose whether to provision compute nodes automatically with the node discovery method.
9. Enter a node discovery IP address range to be used for provisioning compute nodes by node discovery.

   The node discovery IP address range is a temporary IP address range that is used to automatically provision nodes by using the auto node discovery method. This range cannot overlap the range that is specified for the provisioning compute nodes.
10. Select how the management node is connected to the public network. If the management node is not connected to the public network, select the **It is not connected to the public network** option. If your management node is connected to a public network, you can enable the following settings:
    - Enable Platform Cluster Manager specific rules for the management node firewall that is connected to the public interface.
    - Enable NAT forwarding on the management node for all compute nodes.

    For more information about selecting a BNC network option, see the PCM-SE installation guide that is available at this website:

    http://publibfp.dhe.ibm.com/epubs/pdf/c2761070.pdf
11. Enter a domain name for the provisioning network.
12. Enter the IP addresses of your name servers, which are separated by commas.

13. Set the NTP server.

14. Export the home directory on the management node and use it for all compute nodes.

    Change the root password for compute nodes and the default password for the Platform Cluster Manager database.

    A summary of the selected installation settings is displayed as shown in Example 4-8. To change any of these settings, enter 99 to reselect the settings.

Example 4-8 Shows the Platform Cluster Manager installation summary

```
================================================================
    Platform Cluster Manager Installation Summary
================================================================
You have selected the following installation settings:
Provision network domain:                localnet
Provision network interface:             eth0, 10.10.1.0/255.255.255.0
Public network interface:                eth0.800, 9.8.227.0/255.255.255.0
Depot (/install) directory mount point:  /install
OS media:
/home/RHEL6.4-20130130.0-Server-x86_64-DVD1.iso
Network Interface:                       eth0
eth0 IP address range for compute nodes: 10.10.1.21-10.10.1.99
eth0 IP address range for node discovery:10.10.1.101-10.10.1.199
Enable firewall:                         Yes
Enable NAT forwarding:                   Yes
NTP server:                              mgt01.localnet
Database administrator password:         ********
Compute node root password:              *********
Export home directory:                   No
================================================================
   Note: To copy the OS from the OS DVD, you must insert the 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
```

15. Press 1 to begin the installation, as shown in Example 4-8. Example 4-9 shows the installation progress output.

Example 4-9 Installation progress output

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

   Copying Platform Cluster Manager core packages... [-] [-] [\] [\] [|] [|] [|]
[/] [/] [-][  OK  ]

   Copying Platform Cluster Manager add-on kits... [-][  OK  ]

   Adding OS from media '/home/RHEL6.4-20130130.0-Server-x86_64-DVD1.iso'...

   * Verifying that the OS distribution, architecture, and version are
supported...[  OK  ]

   * Detected OS: [rhel 6 x86_64][  OK  ]
```

```
* Copying OS media. This can take a few minutes...
```

When the installation process is complete, you can access PCM-SE through the web-based user interface, as shown in Figure 4-6.

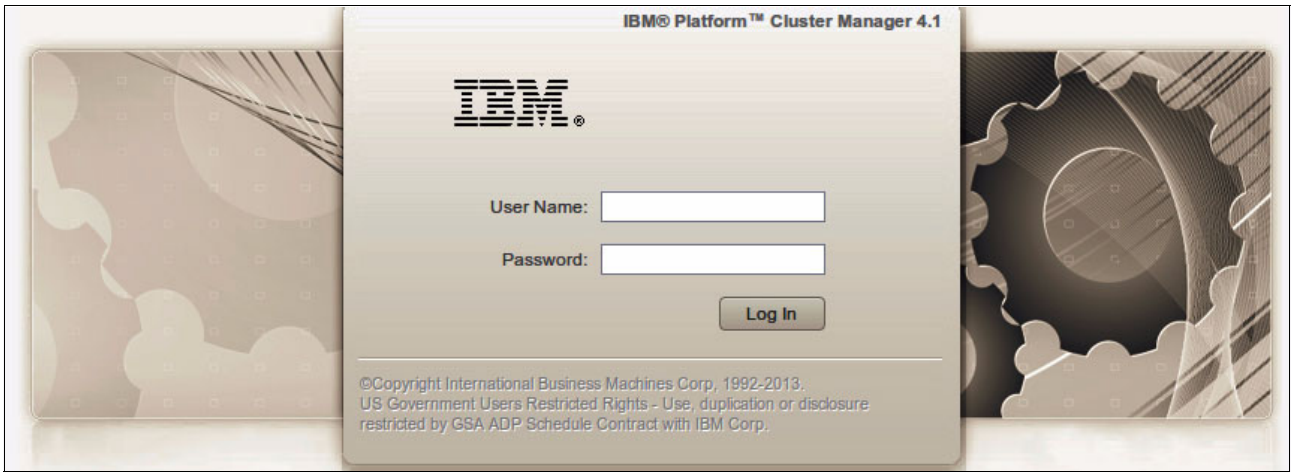

Figure 4-6 IBM Cluster Manager login window

The HTTP address to access the administration interface is shown in Example 4-10.

Example 4-10 Administration interface address

```
http://<<server hostname>>:8080/platform/framework/login/toLogin.action
```

4.2.2 Installing the IBM General Parallel File System

In this section, we describe how to install the IBM General Parallel File System (GPFS) as part of the cluster configuration.

Introducing the General Parallel File System

The GPFS is a cluster file system, which means that it provides concurrent access to a single file system or set of file systems from multiple nodes. These nodes all can be SAN-attached or a mix of SAN- and network-attached. This enables high performance access to this common set of data to support a scale-out solution or provide a high availability platform.

GPFS has many features beyond common data access, including data replication, policy-based storage management, and multi-site operations. You can create a GPFS cluster of AIX nodes, Linux nodes, Windows server nodes, or a mix of all three. GPFS can run on virtualized instances that provide common data access in environments, use logical partitioning, or other hypervisors. Multiple GPFS clusters can share data within a location or across wide area network (WAN) connections.

The strengths of GPFS

GPFS provides a global namespace, shared file system access among GPFS clusters, simultaneous file access from multiple nodes, high recoverability, and data availability through replication, the ability to make changes while a file system is mounted, and simplified administration, even in large environments.

Preparing the environment on Linux nodes

Before proceeding with the installation, prepare your environment by completing the following steps:

1. Add the GPFS bin directory to your shell PATH

   Ensure that the PATH environment variable for the root user on each node includes `/usr/lpp/mmfs/bin`. (This is not required for the operation of GPFS, but it can simplify administration). Example 4-11 shows how to include the GPFS path on the PATH variable:

   Example 4-11 Including the GPFS path

   ```
   [root@compute000 ~]# echo "PATH=\$PATH:/usr/lpp/mmfs/bin" >>/etc/bashrc
   ```

   GPFS commands operate on all nodes that are required to perform tasks. When you are administering a cluster, it can be useful to have a more general form of running commands on all of the nodes. One suggested way to do this is to use a utility, such as **mmdsh** (from GPFS) or **xdsh** (from Platform Cluster Manager) that can run commands on all nodes in the cluster. For example, you can use **xdsh** to check the kernel version of each node in your cluster, as shown in Example 4-12.

   ```
   # xdsh <<noderange>> uname -opr
   ```

   Example 4-12 An xdsh command to check the kernel version of each node

   ```
   [root@mgt01 ~]# xdsh compute uname -opr
   compute000: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   compute001: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   compute002: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   compute003: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   compute004: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   compute005: 2.6.32-358.el6.x86_64 x86_64 GNU/Linux
   ```

2. Verify that prerequisite software is installed.

   Before GPFS is installed, it is necessary to verify that you have the correct levels of the prerequisite software installed on each node in the cluster. To verify whether the latest level of prerequisite supported software is installed before you proceed with your GPFS installation, see the GPFS FAQ, which is available at this website:

   http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.doc%2Fgpfs_faqs%2Fgpfsclustersfaq.html

3. Accept the electronic license agreement on Linux nodes.

   The GPFS software license agreement is shipped with the GPFS software and is viewable electronically.

   When you extract the GPFS software, you are asked whether you accept the license. The electronic license agreement must be accepted before software installation can continue, as shown in Example 4-13 on page 148. Read the software agreement carefully before you accept the license.

Example 4-13 GPFS installation and license acceptance

```
[root@mgt01 gpfsinstall]# ./gpfs_install-3.5.0-0_x86_64

Extracting License Acceptance Process Tool to /usr/lpp/mmfs/3.5 ...
tail -n +429 ./gpfs_install-3.5.0-0_x86_64 | /bin/tar -C /usr/lpp/mmfs/3.5 -xvz
--exclude=*rpm --exclude=*tgz 2> /dev/null 1> /dev/null

Installing JRE ...
tail -n +429 ./gpfs_install-3.5.0-0_x86_64 | /bin/tar -C /usr/lpp/mmfs/3.5
--wildcards -xvz  ./ibm-java*tgz 2> /dev/null 1> /dev/null

Invoking License Acceptance Process Tool ...
/usr/lpp/mmfs/3.5/ibm-java-x86_64-60/jre/bin/java -cp
/usr/lpp/mmfs/3.5/LAP_HOME/LAPApp.jar com.ibm.lex.lapapp.LAP -l
/usr/lpp/mmfs/3.5/LA_HOME -m /usr/lpp/mmfs/3.5 -s /usr/lpp/mmfs/3.5

License Agreement Terms accepted.

Extracting Product RPMs to /usr/lpp/mmfs/3.5 ...
tail -n +429 ./gpfs_install-3.5.0-0_x86_64 | /bin/tar -C /usr/lpp/mmfs/3.5
--wildcards -xvz  ./gpfs.base-3.5.0-3.x86_64.rpm ./gpfs.base_3.5.0-3_amd64.deb
./gpfs.docs-3.5.0-3.noarch.rpm ./gpfs.docs_3.5.0-3_all.deb
./gpfs.gpl-3.5.0-3.noarch.rpm ./gpfs.gpl_3.5.0-3_all.deb
./gpfs.msg.en-us_3.5.0-3_all.deb ./gpfs.msg.en_US-3.5.0-3.noarch.rpm 2> /dev/null
1> /dev/null

   - gpfs.base-3.5.0-3.x86_64.rpm
   - gpfs.base_3.5.0-3_amd64.deb
   - gpfs.docs-3.5.0-3.noarch.rpm
   - gpfs.docs_3.5.0-3_all.deb
   - gpfs.gpl-3.5.0-3.noarch.rpm
   - gpfs.gpl_3.5.0-3_all.deb
   - gpfs.msg.en-us_3.5.0-3_all.deb
   - gpfs.msg.en_US-3.5.0-3.noarch.rpm

Removing License Acceptance Process Tool from /usr/lpp/mmfs/3.5 ...

rm -rf  /usr/lpp/mmfs/3.5/LAP_HOME /usr/lpp/mmfs/3.5/LA_HOME

Removing JRE from /usr/lpp/mmfs/3.5 ...

rm -rf /usr/lpp/mmfs/3.5/ibm-java*tgz
```

4. Install the GPFS software packages.

   The GPFS software is installed by using the **rpm** command (for SLES and RHEL Linux) or the **dpkg** command (for Debian Linux).

   Install the GPFS packages by running the commands, as shown in Example 4-14 on page 149 (RedHat EL 6.4).

Example 4-14 Installing GPFS

```
[root@mgt01 3.5]# pwd
/usr/lpp/mmfs/3.5
[root@mgt01 3.5]# ls
gpfs.base_3.5.0-3_amd64.deb   gpfs.docs_3.5.0-3_all.deb
gpfs.gpl_3.5.0-3_all.deb     gpfs.msg.en-us_3.5.0-3_all.deb     license
gpfs.base-3.5.0-3.x86_64.rpm  gpfs.docs-3.5.0-3.noarch.rpm
gpfs.gpl-3.5.0-3.noarch.rpm  gpfs.msg.en_US-3.5.0-3.noarch.rpm  status.dat
[root@mgt01 3.5]# rpm -ivh *.rpm
Preparing...                ########################################### [100%]
   1:gpfs.base              ########################################### [ 25%]
   2:gpfs.gpl               ########################################### [ 50%]
   3:gpfs.msg.en_US         ########################################### [ 75%]
   4:gpfs.docs              ########################################### [100%]
```

5. After the GPFS base version is installed, install the updates as shown in Example 4-15.

Example 4-15 GPFS updates installation

```
[root@mgt01 FIX]# ls
GPFS-3.5.0.13-x86_64-Linux.tar.gz
[root@mgt01 FIX]# tar -xvf GPFS-3.5.0.13-x86_64-Linux.tar.gz
changelog
gpfs.base_3.5.0-13_amd64_update.deb
gpfs.base-3.5.0-13.x86_64.update.rpm
gpfs.docs_3.5.0-13_all.deb
gpfs.docs-3.5.0-13.noarch.rpm
gpfs.gpl_3.5.0-13_all.deb
gpfs.gpl-3.5.0-13.noarch.rpm
gpfs.msg.en-us_3.5.0-13_all.deb
gpfs.msg.en_US-3.5.0-13.noarch.rpm
README
[root@mgt01 FIX]# ls
changelog                           gpfs.base-3.5.0-13.x86_64.update.rpm
gpfs.gpl_3.5.0-13_all.deb        gpfs.msg.en_US-3.5.0-13.noarch.rpm
GPFS-3.5.0.13-x86_64-Linux.tar.gz    gpfs.docs_3.5.0-13_all.deb
gpfs.gpl-3.5.0-13.noarch.rpm     README
gpfs.base_3.5.0-13_amd64_update.deb  gpfs.docs-3.5.0-13.noarch.rpm
gpfs.msg.en-us_3.5.0-13_all.deb
[root@mgt01 FIX]# rpm -Uvh *.rpm
Preparing...                ########################################### [100%]
   1:gpfs.base              ########################################### [ 25%]
   2:gpfs.gpl               ########################################### [ 50%]
   3:gpfs.msg.en_US         ########################################### [ 75%]
   4:gpfs.docs              ########################################### [100%]
```

6. Verify the GPFS installation on SLES and RHEL Linux.

   To check that the software was successfully installed, run the **rpm** command, as shown in Example 4-16.

   Example 4-16 Checking the installation of GPFS

   ```
   rpm -qa | grep gpfs
   ```

4.2.3 GPFS open source portability layer

On Linux platforms, GPFS uses a loadable kernel module that enables the GPFS daemon to interact with the Linux kernel. Source code is provided for the portability layer so that the GPFS portability can be built and installed on various Linux kernel versions and configurations. When GPFS is installed on Linux, you must build a portability module that is based on your particular hardware platform and Linux distribution to enable communication between the Linux kernel and GPFS. For more information, see this website:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.v3r5.gpfs300.doc%2Fbl1ins_bldgpl.htm

Using automatic configuration tool to build the GPFS portability layer

To simplify the build process, GPFS provides an automatic configuration tool. The following procedure shows the prerequisites and commands that are use to build the GPFS portability layer by using the automatic configuration option (make Autoconfig):

- Package requirements:

```
kernel-devel
kernel-headers
xorg-x11-xauth
gcc-c++ imake
libXp
compat-libstdc++-33
compat-libstdc++-296
libstdc++
```

- Command sequence:

```
cd /usr/lpp/mmfs/src
make Autoconfig
make World
make InstallImages
```

Each kernel module is specific to a Linux version and platform. If you have multiple nodes that are running the same operating system level on the same platform, you can build the kernel module on one node. You then create an RPM that contains the binary module for ease of distribution.

If you choose to generate an RPM package for portability layer binaries, run the following command:

```
make rpm
```

Example 4-17 on page 151 shows this configuration as we performed it during the residency implementation.

Example 4-17 Command to install the prerequisite packages

```
[root@mgt01 src]# yum install kernel-devel kernel-headers xorg-x11-xauth gcc-c++
imake libXp compat-libstdc++-33 compat-libstdc++-296 libstdc++
Resolving Dependencies
--> Running transaction check...
.
.
.
Installed:
  compat-libstdc++-296.i686 0:2.96-144.el6         compat-libstdc++-33.x86_64
0:3.2.3-69.el6         gcc-c++.x86_64 0:4.4.7-3.el6          imake.x86_64
0:1.0.2-11.el6

  kernel-devel.x86_64 0:2.6.32-358.el6             kernel-headers.x86_64
0:2.6.32-358.el6            libXp.x86_64 0:1.0.0-15.1.el6

Dependency Installed:

  cloog-ppl.x86_64 0:0.15.7-1.2.el6      cpp.x86_64 0:4.4.7-3.el6   gcc.x86_64
0:4.4.7-3.el6              glibc.i686 0:2.12-1.107.el6  glibc-devel.x86_64
0:2.12-1.107.el6

  glibc-headers.x86_64 0:2.12-1.107.el6  libgcc.i686 0:4.4.7-3.el6
libstdc++-devel.x86_64 0:4.4.7-3.el6  mpfr.x86_64 0:2.4.1-6.el6
nss-softokn-freebl.i686 0:3.12.9-11.el6

  ppl.x86_64 0:0.10.2-11.el6
Complete!
```

Installing the GPFS portability layer

In this process, we configure the GPFS portability layer for the environment that is used and build the RPM file to be used on the nodes installation, as shown in Example 4-18.

Example 4-18 Configuring the GPFS portability layer and the RPM for the nodes installation

```
[root@mgt01 src]# make Autoconfig
cd /usr/lpp/mmfs/src/config; ./configure --genenvonly; if [ $? -eq 0 ]; then
/usr/bin/cpp -P def.mk.proto > ./def.mk; exit $? || exit 1; else exit $?; fi

[root@mgt01 src]# make World
Verifying that tools to build the portability layer exist....
cpp present
gcc present
g++ present
ld present
cd /usr/lpp/mmfs/src/config; /usr/bin/cpp -P def.mk.proto > ./def.mk; exit $? ||
exit 1
.
.
.
make[1]: Entering directory `/usr/lpp/mmfs/src/gpl-linux'
/usr/bin/install -c -m 0500 lxtrace /usr/lpp/mmfs/src/bin/lxtrace-`cat
//usr/lpp/mmfs/src/gpl-linux/gpl_kernel.tmp.ver`
/usr/bin/install -c -m 0500 kdump   /usr/lpp/mmfs/src/bin/kdump-`cat
//usr/lpp/mmfs/src/gpl-linux/gpl_kernel.tmp.ver`
```

```
make[1]: Leaving directory `/usr/lpp/mmfs/src/gpl-linux'

[root@mgt01 src]# make InstallImages
(cd gpl-linux; /usr/bin/make InstallImages; \
   exit $?) || exit 1
make[1]: Entering directory `/usr/lpp/mmfs/src/gpl-linux'
Pre-kbuild step 1...
make[2]: Entering directory `/usr/src/kernels/2.6.32-358.el6.x86_64'
  INSTALL /usr/lpp/mmfs/src/gpl-linux/kdump-kern-dummy.ko
  INSTALL /usr/lpp/mmfs/src/gpl-linux/kdump-kern-dwarfs.ko
  INSTALL /usr/lpp/mmfs/src/gpl-linux/mmfs26.ko
  INSTALL /usr/lpp/mmfs/src/gpl-linux/mmfslinux.ko
  INSTALL /usr/lpp/mmfs/src/gpl-linux/tracedev.ko
  DEPMOD  2.6.32-358.el6.x86_64
make[2]: Leaving directory `/usr/src/kernels/2.6.32-358.el6.x86_64'
make[1]: Leaving directory `/usr/lpp/mmfs/src/gpl-linux'

[root@mgt01 src]# make rpm
rm -rf /tmp/rpm
rpmbuild --define "MODKERNEL `cat
//usr/lpp/mmfs/src/gpl-linux/gpl_kernel.tmp.ver`" --define "GPLDIR
/usr/lpp/mmfs/src/gpl-linux" -bb /usr/lpp/mmfs/src/config/gpfs.gplbin.spec
--buildroot=/tmp/rpm
Executing(%install): /bin/sh -e /var/tmp/rpm-tmp.D9HwQd
+ umask 022
+ cd /root/rpmbuild/BUILD
+ '[' x/tmp/rpm = x ']'
+ '[' '' '!=' Linux ']'
+ export GPFS_LPP=/tmp/rpm/usr/lpp/mmfs
+ GPFS_LPP=/tmp/rpm/usr/lpp/mmfs
+ export GPFS_KO=/tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ GPFS_KO=/tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ export GPFS_BIN=/tmp/rpm/usr/lpp/mmfs/bin
+ GPFS_BIN=/tmp/rpm/usr/lpp/mmfs/bin
+ export LPPBIN=/usr/lpp/mmfs/src/bin
+ LPPBIN=/usr/lpp/mmfs/src/bin
+ install -d -m 755 /tmp/rpm/usr/lpp/mmfs
+ install -d -m 755 /tmp/rpm/usr/lpp/mmfs/bin
+ install -d -m 755 /tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ install -m 500 /usr/lpp/mmfs/src/bin/kdump-2.6.32-358.el6.x86_64
/tmp/rpm/usr/lpp/mmfs/bin/kdump-2.6.32-358.el6.x86_64
+ install -m 500 /usr/lpp/mmfs/src/bin/lxtrace-2.6.32-358.el6.x86_64
/tmp/rpm/usr/lpp/mmfs/bin/lxtrace-2.6.32-358.el6.x86_64
+ install -m 500 /usr/lpp/mmfs/src/gpl-linux/mmfs26.ko
/tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ install -m 500 /usr/lpp/mmfs/src/gpl-linux/mmfslinux.ko
/tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ install -m 500 /usr/lpp/mmfs/src/gpl-linux/tracedev.ko
/tmp/rpm/lib/modules/2.6.32-358.el6.x86_64/extra
+ exit 0
Processing files: gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64
Checking for unpackaged file(s): /usr/lib/rpm/check-files /tmp/rpm
Wrote:
/root/rpmbuild/RPMS/x86_64/gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.bCCFCI
```

```
+ umask 022
+ cd /root/rpmbuild/BUILD
+ /bin/rm -rf /tmp/rpm
+ exit 0
```

Note: When the command finishes, it displays the location of the generated portability layer RPM, as shown in the following example:

```
Wrote: /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
```

The generated RPM can be deployed only to machines with the identical architecture, distribution level, Linux kernel, and GPFS maintenance level.

4.2.4 Configuring GPFS on the cluster initial nodes

In this section, we describe the process to configure GPFS for the cluster initial nodes that exist on the configuration. GPFS must have the SSH key that is exchanged between servers. This step is not necessary on our cluster because the Platform Cluster Manager installation makes the SSH key exchange between the compute nodes. Therefore, this requirement is fulfilled when the nodes are installed.

Complete the following steps to configure GPFS on the cluster initial nodes:

1. Copy all GPFS RPMs, base level, downloaded updates, and the generated portability layer to the initial nodes, as shown in Example 4-19.

Example 4-19 All RPMs are copied on the initial nodes

```
[root@compute000 GPFS_RPMS]# ls
gpfs.base-3.5.0-13.x86_64.update.rpm  gpfs.docs-3.5.0-13.noarch.rpm
gpfs.gpl-3.5.0-13.noarch.rpm
gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
gpfs.msg.en_US-3.5.0-3.noarch.rpm
gpfs.base-3.5.0-3.x86_64.rpm          gpfs.docs-3.5.0-3.noarch.rpm
gpfs.gpl-3.5.0-3.noarch.rpm   gpfs.msg.en_US-3.5.0-13.noarch.rpm
```

2. Install the base GPFS software, as shown in Example 4-20.

Example 4-20 Installing the base GPFS software

```
[root@compute000 GPFS_RPMS]# rpm -ivh gpfs.*3.5.0-3*
Preparing...                ########################################### [100%]
   1:gpfs.base              ########################################### [ 25%]
   2:gpfs.gpl               ########################################### [ 50%]
   3:gpfs.msg.en_US         ########################################### [ 75%]
   4:gpfs.docs              ########################################### [100%]
[root@mgt01 GPFS_RPMS]#
```

3. Install the GPFS updates and the generated portability layer RPMs, as shown in Example 4-21 on page 154.

Example 4-21 Installing updates and portability layer

```
[root@mgt01 GPFS_RPMS]# rpm -Uvh gpfs.*3.5.0-13*
Preparing...                ########################################### [100%]
   1:gpfs.base              ########################################### [ 20%]
   2:gpfs.gpl               ########################################### [ 40%]
   3:gpfs.gplbin-2.6.32-358.########################################### [ 60%]
   4:gpfs.msg.en_US         ########################################### [ 80%]
   5:gpfs.docs              ########################################### [100%]
```

4. Repeat steps 1 - 3 on all nodes.
5. Use the **mmcrcluster** command to create the GPFS cluster, as shown in Example 4-22.

Example 4-22 Creating the GPFS cluster by using the mmcrcluster command

```
[root@compute000 ~]# mmcrcluster -N
compute000:manager-quorum,compute001:manager-quorum,compute002:quorum,compute003:q
uorum,compute004:quorum,compute005 -p compute000 -s compute001 -C bigdata -r
/usr/bin/ssh -R /usr/bin/scp
Warning: Permanently added 'compute000,10.10.1.21' (RSA) to the list of known
hosts.
Thu Oct 24 16:46:16 BRST 2013: mmcrcluster: Processing node compute000
Thu Oct 24 16:46:16 BRST 2013: mmcrcluster: Processing node compute001
Thu Oct 24 16:46:19 BRST 2013: mmcrcluster: Processing node compute002
Thu Oct 24 16:46:21 BRST 2013: mmcrcluster: Processing node compute003
Thu Oct 24 16:46:24 BRST 2013: mmcrcluster: Processing node compute004
Thu Oct 24 16:46:26 BRST 2013: mmcrcluster: Processing node compute005
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

6. Accept the license for all nodes, as shown in Example 4-23.

Example 4-23 Accepting the license

```
[root@compute000 ~]# mmchlicense server --accept -N
compute000,compute001,compute002,compute003,compute004,compute005
The following nodes will be designated as possessing GPFS server licenses:
        compute000
        compute001
        compute002
        compute003
        compute004
        compute005
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

7. Run the **mmlscluster** command for the GPFS cluster information, as shown in Example 4-24 on page 155.

Example 4-24 Listing the GPFS cluster

```
[root@compute000 ~]# mmlscluster
GPFS cluster information
========================
  GPFS cluster name:         bigdata.compute001
  GPFS cluster id:           10791631865508358903
  GPFS UID domain:           bigdata.compute001
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp

GPFS cluster configuration servers:
-----------------------------------
  Primary server:    compute001
  Secondary server:  compute002

 Node  Daemon node name  IP address  Admin node name  Designation
-----------------------------------------------------------------
   1   compute000        10.10.1.21  compute000       quorum-manager
   2   compute001        10.10.1.23  compute001       quorum-manager
   3   compute002        10.10.1.24  compute002       quorum
   4   compute003        10.10.1.25  compute003       quorum
   5   compute004        10.10.1.22  compute004       quorum
   6   compute005        10.10.1.26  compute005
```

Tip: To check that GPFS is running on all nodes, run the following GPFS start command:

`mmstartup -a`

Creating partitions

For the next steps, we create disks partitions to accommodate the data and metadata into the GPFS file system. In our case, we have three metadata replicas, so we create three partitions, one in each server. You can use a separate disk for this process; therefore, you do not need to do this step.

Note: The metadata and data have different placement policies, so we separate data and metadata.

If you are using the same disk to create the metadata space, run the **`fdisk`** command to create the partitions to use data and metadata, as shown in Example 4-25.

Example 4-25 Creating partitions by using fdisk

```
[root@compute000 ~]# fdisk /dev/dm-0

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
```

```
First cylinder (1-36407, default 1):
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-36407, default 36407): +80G

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (10445-36407, default 10445):
Using default value 10445
Last cylinder, +cylinders or +size{K,M,G} (10445-36407, default 36407):
Using default value 36407

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 22: Invalid argument.
The kernel still uses the old table. The new table will be used at
the next reboot or after you run partprobe(8) or kpartx(8)
Syncing disks.

[root@compute000 ~]# partprobe
Warning: WARNING: the kernel failed to re-read the partition table on /dev/sda
(Device or resource busy).  As a result, it may not reflect all of your changes
until after reboot.
```

Now we show an excerpt of the stanza file that we used for the disk distribution (see Example 4-26). Each dm device represents a single spindle with the exemption of dm-4 and dm-5, which are partitions that were created in the previous step for nodes compute000, compute001, and compute002 in our environment (these nodes have the same entries on the stanza file). In these three servers, we have five NSDs, one for the system pool where we have the metadata partitions and the other four are for the data pool where we have write affinity for the data. On all other compute nodes, we have only four NSDs where we configure only the data pool (these nodes have the same entries as compute005).

Example 4-26 Excerpt of the stanza file that is used in the environment

```
%pool:
pool=system
blockSize=1024K
layoutMap=cluster
allowWriteAffinity=no

%pool:
pool=datapool
blockSize=1024K
layoutMap=cluster
allowWriteAffinity=yes
writeAffinityDepth=1
blockGroupFactor=128

%nsd:  device=/dev/dm-4 servers=compute000
```

```
  usage=metadataOnly failureGroup=1 pool=system

%nsd:  device=/dev/dm-5 servers=compute000
  usage=dataOnly failureGroup=1,0,0 pool=datapool

%nsd:  device=/dev/dm-1 servers=compute000
  usage=dataOnly failureGroup=1,0,0 pool=datapool

%nsd:  device=/dev/dm-2 servers=compute000
  usage=dataOnly failureGroup=1,0,0 pool=datapool

%nsd:  device=/dev/dm-3 servers=compute000
  usage=dataOnly failureGroup=1,0,0 pool=datapool

.
.
.

%nsd:   device=/dev/dm-0 servers=compute005
  usage=dataOnly failureGroup=6,0,0 pool=datapool

%nsd:   device=/dev/dm-1 servers=compute005
  usage=dataOnly failureGroup=6,0,0 pool=datapool

%nsd:   device=/dev/dm-2 servers=compute005
  usage=dataOnly failureGroup=6,0,0 pool=datapool

%nsd:   device=/dev/dm-3 servers=compute005
  usage=dataOnly failureGroup=6,0,0 pool=datapool
```

With the stanza file that is shown in Example 4-26, we can now create the NSD, as shown in Example 4-27.

Example 4-27 Command to create the nsd with stanza file

```
[root@compute000 ~]# mmcrnsd -F /tmp/stanza.txt
mmcrnsd: Processing disk dm-4
mmcrnsd: Processing disk dm-5
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Processing disk dm-4
mmcrnsd: Processing disk dm-5
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Processing disk dm-4
mmcrnsd: Processing disk dm-5
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Processing disk dm-0
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
```

```
mmcrnsd: Processing disk dm-0
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Processing disk dm-0
mmcrnsd: Processing disk dm-1
mmcrnsd: Processing disk dm-2
mmcrnsd: Processing disk dm-3
mmcrnsd: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Attention: After the command completion, the `stanza.txt` file is automatically updated by the system. The name of the GPFS disks is added to each NDS stanza.

Now that we created all of the NSDs, it is time to create the file system. Because we have local disks and they are not redundant (no RAID array), we use GPFS replicas on this environment, as shown in Example 4-28.

Example 4-28 Creating the file system with three replicas for data and metadata

```
[root@compute000 ~]# mmcrfs bigdatafs -F /tmp/stanza.txt -A yes -B 1024K -j
cluster -m 3 -M 3 -r 3 -R 3 -T /mapred/

The following disks of bigdatafs will be formatted on node compute000:
    gpfs1nsd: size 83891398 KB
    gpfs2nsd: size 208547797 KB
    gpfs3nsd: size 285699072 KB
    gpfs4nsd: size 292444160 KB
    gpfs5nsd: size 292444160 KB
    gpfs6nsd: size 83891398 KB
    gpfs7nsd: size 208547797 KB
    gpfs8nsd: size 292444160 KB
    gpfs9nsd: size 292444160 KB
    gpfs10nsd: size 292444160 KB
    gpfs11nsd: size 83891398 KB
    gpfs12nsd: size 58950517 KB
    gpfs13nsd: size 142849536 KB
    gpfs14nsd: size 142849536 KB
    gpfs15nsd: size 142849536 KB
    gpfs16nsd: size 142849536 KB
    gpfs17nsd: size 142849536 KB
    gpfs18nsd: size 142849536 KB
    gpfs19nsd: size 142849536 KB
    gpfs20nsd: size 142849536 KB
    gpfs21nsd: size 142849536 KB
    gpfs22nsd: size 142849536 KB
    gpfs23nsd: size 142849536 KB
    gpfs24nsd: size 292444160 KB
    gpfs25nsd: size 292444160 KB
    gpfs26nsd: size 142849536 KB
    gpfs27nsd: size 142849536 KB
Formatting file system ...
Disks up to size 1.2 TB can be added to storage pool system.
Disks up to size 2.4 TB can be added to storage pool datapool.
```

```
Creating Inode File
  67 % complete on Mon Oct 28 10:18:53 2013
 100 % complete on Mon Oct 28 10:18:56 2013
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
Formatting Allocation Map for storage pool datapool
Completed creation of file system /dev/bigdatafs.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Tip: This is an asynchronous process that takes time to synchronize and show 100% usage.

Mount the file system. Use **mmsdh** to see the mount on all of nodes, as shown in Example 4-29.

Example 4-29 Mounts right after mounting

```
[root@compute000 ~]# mmmount /mapred -a
Mon Oct 28 10:39:02 BRST 2013: mmmount: Mounting file systems ...
[root@compute000 ~]#  mmdsh df -k |grep mapred
compute000:  /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute001:  /dev/bigdatafs       4665888768 4665888768         0 100% /mapred
compute003:  /dev/bigdatafs       4665888768 4665888768         0 100% /mapred
compute002:  /dev/bigdatafs       4665888768 4665888768         0 100% /mapred
compute004:  /dev/bigdatafs       4665888768 4665888768         0 100% /mapred
compute005:  /dev/bigdatafs       4665888768 4665888768         0 100% /mapred
```

After a few minutes, the file system is free and ready for use on all nodes, as shown in Example 4-30.

Example 4-30 File system free on all nodes

```
[root@mgt01 ~]# xdsh compute df -k |grep big
compute000: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute002: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute001: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute003: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute005: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
compute004: /dev/bigdatafs       4665888768     73728 4665815040   1% /mapred
```

Now that the file system is mounted, we must enforce the use of the location affinity for read and a file set to use the GPFS space for Platform Symphony MapReduce local directory. We then create a placement policy that makes the local directories on the newly created file set to use only one copy of data and then apply the policy on the file system. Example 4-31 on page 160 shows the commands that we used.

Example 4-31 Tune GPFS policies

```
[root@compute000 ~]# mkdir /mapred/mapred

[root@compute000 ~]# mmcrfileset bigdatafs mapred_local
Fileset mapred_local created with id 1 root inode 127077.

[root@compute000 ~]# mmlinkfileset bigdatafs mapred_local -J /mapred/mapred/local
Fileset mapred_local linked at /mapred/mapred/local

[root@compute000 ~]# /usr/lpp/mmfs/bin/mmchconfig readReplicaPolicy=local
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.

[root@compute000 ~]# /usr/lpp/mmfs/bin/mmchconfig restripeOnDiskFailure=yes -i
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.

[root@compute000 ~]# vi /tmp/placement.policy

[root@compute000 ~]# cat /tmp/placement.policy
RULE 'defplacement' SET POOL 'datapool'
RULE 'R1' SET POOL 'datapool' REPLICATE (1,3) FOR FILESET (mapred_local)

[root@compute000 ~]# mmchpolicy bigdatafs /tmp/placement.policy
```

4.2.5 Hadoop installation process for the initial nodes

The Hadoop software installation can be automated by using the `xdsh` tool to make the installation easier.

Installing IBM Java

Install the Java prerequisite `compat-libstdc++-33.x86_64`, as shown in Example 4-32.

Example 4-32 Installing compat-libstdc++-33

```
[root@compute000 tmp]# yum install compat-libstdc++-33.x86_64
Loaded plugins: product-id, security, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package compat-libstdc++-33.x86_64 0:3.2.3-69.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
================================================
 Package                         Arch                 Version
Repository                        Size
```

```
================================================================================
================================================
Installing:
 compat-libstdc++-33              x86_64              3.2.3-69.el6
rhels6.4-path0                   183 k

Transaction Summary
================================================================================
================================================
Install       1 Package(s)

Total download size: 183 k
Installed size: 806 k
Is this ok [y/N]: y
Downloading Packages:
compat-libstdc++-33-3.2.3-69.el6.x86_64.rpm
| 183 kB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
 Installing : compat-libstdc++-33-3.2.3-69.el6.x86_64
1/1
 Verifying : compat-libstdc++-33-3.2.3-69.el6.x86_64
1/1

Installed:
  compat-libstdc++-33.x86_64 0:3.2.3-69.el6

Complete!
```

Download IBM Java and put the RPM file in the `/tmp` directory. You are now ready to install IBM Java 1.6. Example 4-33 shows the installation in our environment.

Example 4-33 Installing IBM Java

```
root@compute000 tmp]# rpm -ivh ibm-java-x86_64-sdk-6.0-8.0.x86_64.rpm
Preparing...                ########################################### [100%]
   1:ibm-java-x86_64-sdk    ########################################### [100%]
```

Downloading and installing Hadoop

IBM Platform Symphony v6.1.1 accepts Hadoop v1.1.1; therefore, we download and install this version to use it with IBM Platform Symphony MapReduce, as shown in Example 4-34.

Example 4-34 Downloading and installing Hadoop v1.1.1

```
[root@compute000 tmp]# wget
http://archive.apache.org/dist/hadoop/core/hadoop-1.1.1/hadoop-1.1.1-1.x86_64.rpm
--2013-10-28 14:51:26--
http://archive.apache.org/dist/hadoop/core/hadoop-1.1.1/hadoop-1.1.1-1.x86_64.rpm
Resolving archive.apache.org (archive.apache.org)... 140.211.11.131,
192.87.106.229, 2001:610:1:80bc:192:87:106:229
```

```
Connecting to archive.apache.org (archive.apache.org)|140.211.11.131|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 35122218 (33M)
Saving to: `hadoop-1.1.1-1.x86_64.rpm'

100%[======================================>] 35,122,218   294K/s   in 2m 55s

2013-10-28 14:54:21 (196 KB/s) - `hadoop-1.1.1-1.x86_64.rpm' saved
[35122218/35122218]
[root@compute000 tmp]# rpm -ivh hadoop-1.1.1-1.x86_64.rpm
Preparing...                ########################################### [100%]
   1:hadoop                 ########################################### [100%]
```

4.2.6 Installing IBM Platform Symphony

Installing IBM Platform Symphony requires to know whether you are setting a failover server. In our scenario, we install one master server and a failover server.

Installing the master node

To install the IBM Platform Symphony master node, we export some variables that the program uses during the installation. Example 4-35 shows all of the needed variables.

Example 4-35 Exporting variables to install IBM Platform Symphony

```
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
export LD_LIBRARY_PATH=/usr/lib64
export HADOOP_VERSION=1_1_1
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
```

The cluster uses an admin user to perform the installation; therefore, we create this user and install the IBM Platform Symphony master node (in our case, compute000), as shown in Example 4-36.

Example 4-36 Creating the cluster admin user and installing the software

```
[root@compute000 Inst_Symphony]# groupadd egoadmin
[[root@compute000 Inst_Symphony]# useradd -g egoadmin egoadmin
[root@compute000 Inst_Symphony]# passwd egoadmin
Changing password for user egoadmin.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

[root@compute000 Inst_Symphony]# export CLUSTERADMIN=egoadmin


[root@compute000 Inst_Symphony]# ./symSetup6.1.1_lnx26-lib23-x64.bin


Extracting files... done.
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
.
.
.
The installation will be processed using the following settings:
Workload Execution Mode (WEM): Simplified
Cluster Administrator: egoadmin
Cluster Name: cluster1
Installation Directory: /opt/ibm/platformsymphony
Connection Base Port: 7869

   1:ego-lnx26-lib23-x64    ########################################## [100%]


Platform EGO 1.2.8 is installed successfully.
Install the SOAM package to complete the installation process. Source the
environment and run the <egoconfig> command to complete the setup after installing
the SOAM package.
Preparing...                ########################################## [100%]

   1:soam-lnx26-lib23-x64   ########################################## [100%]


IBM Platform Symphony 6.1.1 is installed at /opt/ibm/platformsymphony.
```

Now that the software is installed, you can log in by using the cluster admin account, set up the profile environment, create the cluster, and then use the IBM Platform Symphony Advanced Edition entitlement file so you can use the Platform Symphony MapReduce feature, as shown in Example 4-37.

Example 4-37 Installing and configuring IBM Platform Symphony

```
[root@compute000 Inst_Symphony]# su - egoadmin

Setup the environment variables.

[egoadmin@compute000 ~]$ . /opt/ibm/platformsymphony/profile.platform

[egoadmin@compute000 ~]$ egoconfig join compute000
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 <compute000> is the master
host.
Run <egoconfig setentitlement "entitlementfile"> before using the cluster.

Setting the entitlement for Symphony Advanced Edition:

[egoadmin@compute000 ~]$ egoconfig setentitlement
/tmp/Inst_Symphony/platform_sym_adv_entitlement.dat
```

```
Successfully set entitlement.

Change this path to point to your entitlement file.


On the shared filesystem create a shared directory for Symphony and give egoadmin
group and user privileges:

[root@compute000 ~]#  mkdir -p /mapred/symphony/kernel/conf
[root@compute000 ~]# chown -R egoadmin:egoadmin /mapred
```

To have a high available Platform Symphony cluster, we configure the master node to point the cluster configuration to the GPFS file system. We run the command as shown in Example 4-38.

Example 4-38 Setting the cluster to use configuration files on GPFS

```
[egoadmin@compute000 ~]$ egoconfig mghost /mapred/symphony
This host will use configuration files on a shared directory. Make sure that
/mapred/symphony 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
The shared configuration directory is /mapred/symphony/kernel/conf. You must reset
your environment before you can run any more EGO commands. Source the environment
/opt/ibm/platformsymphony/cshrc.platform or
/opt/ibm/platformsymphony/profile.platform again.
```

As shown in Example 4-39, rerun the Platform Symphony environment profile so that your cluster points to the correct paths.

Example 4-39 Running the Platform Symphony profile

```
[egoadmin@compute000 ~]$ . /opt/ibm/platformsymphony/profile.platform
```

Tip: It is a good idea to add the environment profile to the `egoadmin.bashrc` so you do not have to run it whenever you must perform any configuration tasks on the Platform Symphony cluster.

If you want to set the cluster to start at boot and grant root privileges for the egoadmin account, run as root the `egosetrc.sh` and `egosetsudoers.sh` scripts, as shown in Example 4-40.

Example 4-40 Running root scripts to enable sudo for an autostart

```
[egoadmin@compute000 ~]$ su -
Password:
[root@compute000 ~]# . /opt/ibm/platformsymphony/profile.platform
[root@compute000 ~]# egosetrc.sh
egosetrc succeeds
[root@compute000 ~]# egosetsudoers.sh
egosetsudoers succeeds
```

Because we are using a simplified WEM, if you need cluster scalability (more than 1000 CPU and users), increase the number of open files that are permitted for each user, as shown in Example 4-41.

Example 4-41 Changing nofile on the cluster

```
[root@compute000 ~]# vi /etc/security/limits.conf
[root@compute000 ~]# grep nofile /etc/security/limits.conf
#        - nofile - max number of open files
*       hard    nofile  6400
```

Now start your Platform Symphony cluster by running the **egosh ego start** command from the command line, as shown in Example 4-42.

Example 4-42 Starting the cluster on the management node

```
[egoadmin@compute000 ~]$ egosh ego start
Start up LIM on <compute000> ...... done
```

Installing the management failover node

To install the management failover node, we export some variables (see Example 4-43) that the program uses during the installation.

Example 4-43 Exporting variables to install Platform Symphony

```
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
export LD_LIBRARY_PATH=/usr/lib64
export HADOOP_VERSION=1_1_1
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
```

The cluster uses an admin user to perform the installation, as shown in Example 4-44. We then create this user and install it on the Platform Symphony master node (in our case, compute000).

Example 4-44 Adding user and installing the software

```
[root@compute001 Inst_Symphony]# groupadd egoadmin
[root@compute001 Inst_Symphony]# useradd -g egoadmin egoadmin
[root@compute001 Inst_Symphony]# passwd egoadmin
Changing password for user egoadmin.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

[root@compute001 Inst_Symphony]# export CLUSTERADMIN=egoadmin


[root@compute001 Inst_Symphony]# ./symSetup6.1.1_lnx26-lib23-x64.bin


Extracting files... done.
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
.
.
.
The installation will be processed using the following settings:
Workload Execution Mode (WEM): Simplified
Cluster Administrator: egoadmin
Cluster Name: cluster1
Installation Directory: /opt/ibm/platformsymphony
Connection Base Port: 7869

   1:ego-lnx26-lib23-x64    ########################################## [100%]


Platform EGO 1.2.8 is installed successfully.
Install the SOAM package to complete the installation process. Source the
environment and run the <egoconfig> command to complete the setup after installing
the SOAM package.
Preparing...                ########################################## [100%]

   1:soam-lnx26-lib23-x64   ########################################## [100%]


IBM Platform Symphony 6.1.1 is installed at /opt/ibm/platformsymphony.
```

Now that the software is installed, you can log in by using the cluster admin account, set up the profile environment, create the cluster, and use the Platform Symphony Advanced Edition entitlement file so that you can use the Platform Symphony MapReduce feature, as shown in Example 4-45.

Example 4-45 Installing and configuring IBM Platform Symphony

```
[root@compute001 Inst_Symphony]# su - egoadmin

Setup the environment variables.

[egoadmin@compute001 ~]$ . /opt/ibm/platformsymphony/profile.platform

[egoadmin@compute001 ~]$ egoconfig join compute000
```

To have a high available Platform Symphony cluster, we configure the failover node to point the cluster configuration to the GPFS file system. Therefore, we run the command as shown in Example 4-46 on page 167.

Example 4-46 Setting the cluster to use configuration files on GPFS

```
[egoadmin@compute001 ~]$ egoconfig mghost /mapred/symphony
This host will use configuration files on a shared directory. Make sure that
/mapred/symphony 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
The shared configuration directory is /mapred/symphony/kernel/conf. You must reset
your environment before you can run any more EGO commands. Source the environment
/opt/ibm/platformsymphony/cshrc.platform or
/opt/ibm/platformsymphony/profile.platform again.
```

As shown in Example 4-47, you must rerun the IBM Platform Symphony environment profile for your cluster to point to the correct paths.

Example 4-47 Running the IBM Platform Symphony profile

```
[egoadmin@compute001 ~]$ . /opt/ibm/platformsymphony/profile.platform
```

Tip: Remember to add the environment profile to the `egoadmin.bashrc` so you do not have to run it whenever you must perform any configuration on the IBM Platform Symphony cluster.

If you want to set the cluster to start at boot and to grant root privileges to the egoadmin account, run as root the `egosetrc.sh` and `egosetsudoers.sh` scripts, as shown in Example 4-48.

Example 4-48 Running root scripts to enable sudo for an autostart

```
[egoadmin@compute001 ~]$ su -
Password:
[root@compute001 ~]# . /opt/ibm/platformsymphony/profile.platform
[root@compute001 ~]# egosetrc.sh
egosetrc succeeds
[root@compute001 ~]# egosetsudoers.sh
egosetsudoers succeeds
```

Because we are using a simplified WEM, if you need cluster scalability (more than 1000 CPU and users), increase the number of open files that are permitted for each user, as shown in Example 4-49.

Example 4-49 Changing the nofile on the cluster

```
[root@compute001 ~]# vi /etc/security/limits.conf
[root@compute001 ~]# grep nofile /etc/security/limits.conf
#        - nofile - max number of open files
*       hard    nofile  6400
```

Now start your IBM Platform Symphony cluster by running the **`egosh ego start`** command from the command line, as shown in Example 4-50.

Example 4-50 Starting the cluster on the management node

```
[egoadmin@compute001 ~]$ egosh ego start
Start up LIM on <compute001> ...... done
```

Installing failover node

By using the graphical interface, complete the following steps to add compute001 node as the failover node:

1. Log in to the IBM Platform Symphony web-based GUI by using Admin as the User Name and Password, as shown in Figure 4-7. Point the web browser to:

   `http://<hostname/IP>:8080/platform`

Figure 4-7 Log in to the IBM Platform Symphony GUI

2. Click **Settings** → **Cluster** → **Master and Failovers,** as shown in Figure 4-8.

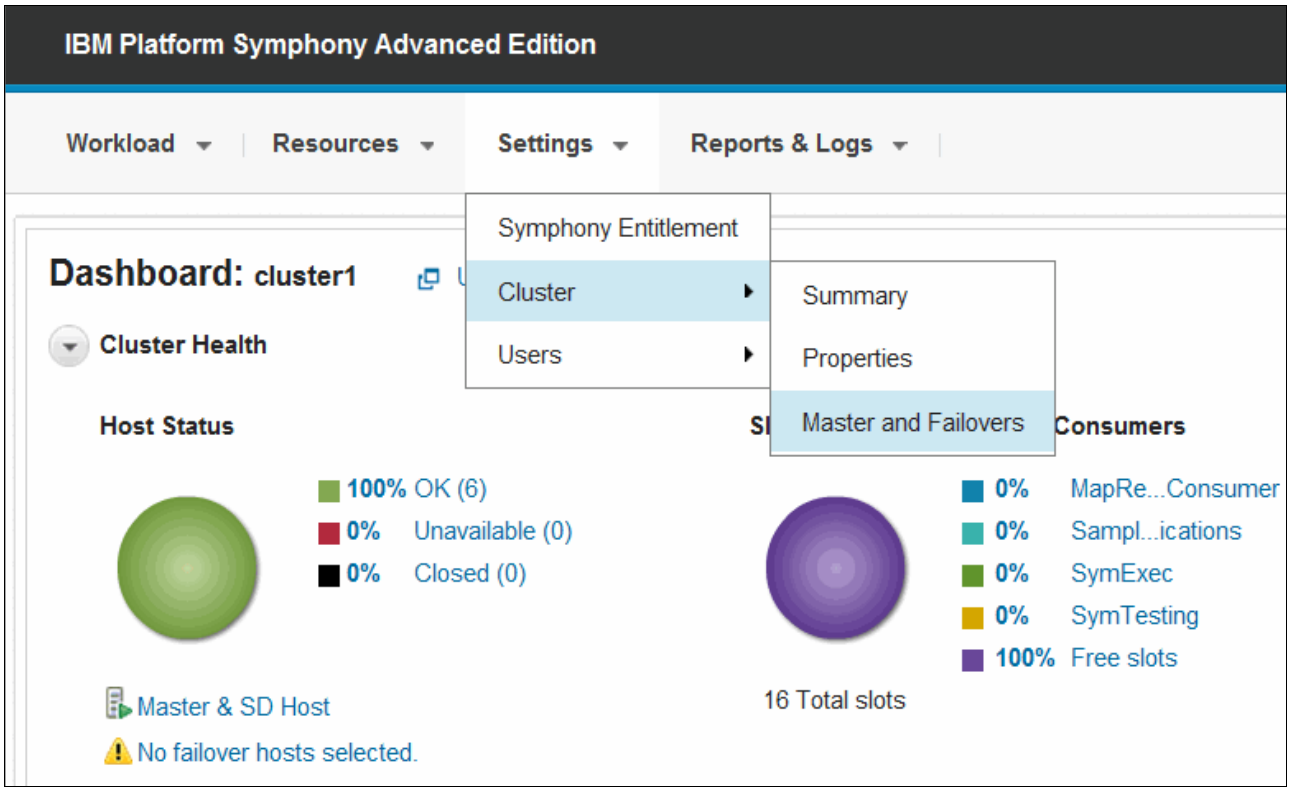

Figure 4-8 Selecting menu for the configure master and failover nodes

3. Select your failover node and then click **Add**, as shown in Figure 4-9.

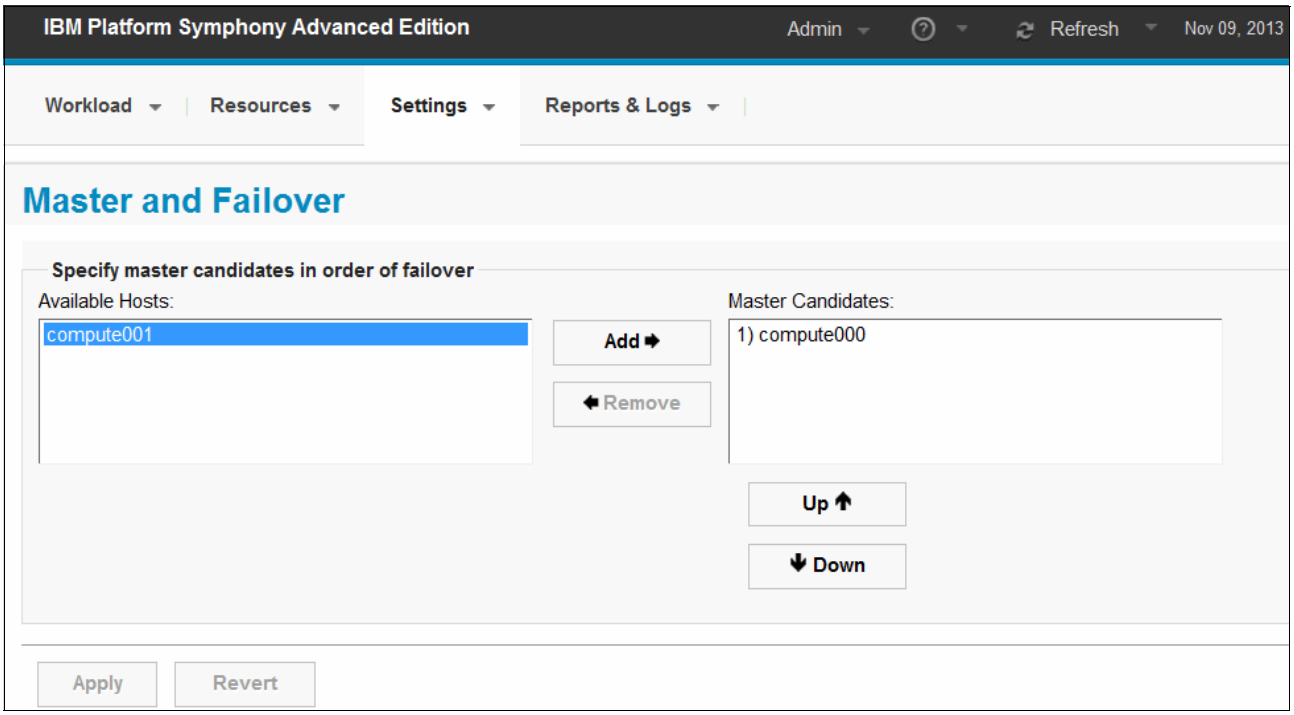

Figure 4-9 Selecting the failover node

4. Click **Apply** (as shown in Figure 4-10) and your failover node is configured.

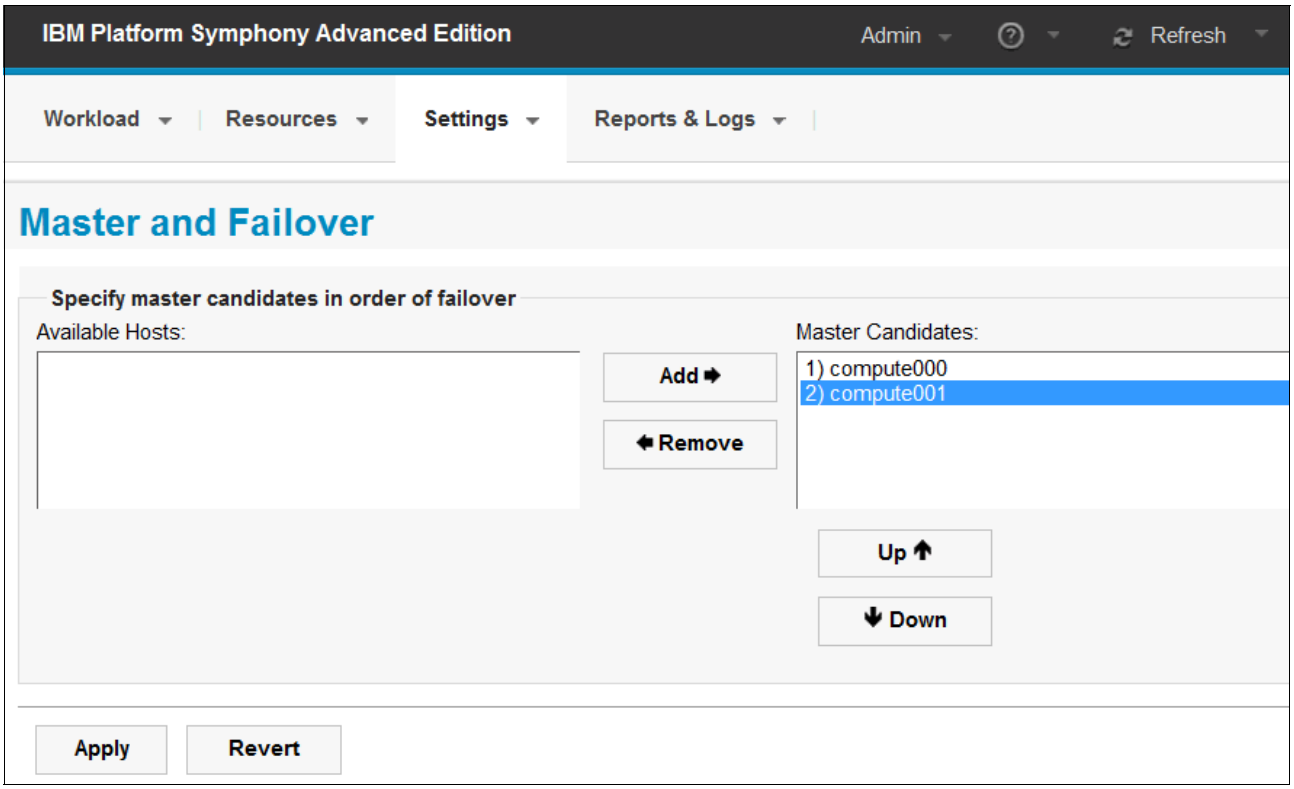

Figure 4-10 Applying the configuration

Installing the compute node

There are two ways to install the computes nodes: manually and automatically. In this section, we describe how to perform the installation manually. If you want to use the automated process to install the compute nodes, see 4.2.7, “Building an automatic kit template” on page 172 ”.

Tip: If you have many compute nodes, use the Platform Cluster Manager automated kit installation method.

To install the compute node manually, we still must export the variables that the program uses during the installation. Example 4-51 shows the needed variables.

Example 4-51 Exporting variables to install Platform Symphony

```
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
export LD_LIBRARY_PATH=/usr/lib64
export HADOOP_VERSION=1_1_1
export JAVA_HOME=/opt/ibm/java-x86_64-60
export HADOOP_HOME=/usr/share/hadoop
```

The cluster uses an admin user to perform the installation, as shown in Example 4-52 on page 171. Therefore, we create this user and then install it on the Platform Symphony master node (in our case, compute000).

Example 4-52 Adding the user and installing the software

```
[root@compute002 Inst_Symphony]# groupadd egoadmin
[root@compute002 Inst_Symphony]# useradd -g egoadmin egoadmin
[root@compute002 Inst_Symphony]# passwd egoadmin
Changing password for user egoadmin.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

[root@compute002 Inst_Symphony]# export CLUSTERADMIN=egoadmin


[root@compute002 Inst_Symphony]# ./symSetup6.1.1_lnx26-lib23-x64.bin


Extracting files... done.
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
.
.
.
The installation will be processed using the following settings:
Workload Execution Mode (WEM): Simplified
Cluster Administrator: egoadmin
Cluster Name: cluster1
Installation Directory: /opt/ibm/platformsymphony
Connection Base Port: 7869

   1:ego-lnx26-lib23-x64    ########################################## [100%]


Platform EGO 1.2.8 is installed successfully.
Install the SOAM package to complete the installation process. Source the
environment and run the <egoconfig> command to complete the setup after installing
the SOAM package.
Preparing...                ########################################## [100%]

   1:soam-lnx26-lib23-x64   ########################################## [100%]


IBM Platform Symphony 6.1.1 is installed at /opt/ibm/platformsymphony.
```

Now that the software is installed, log in by using the cluster admin account, set up the profile environment, create the cluster, and use the Platform Symphony Advanced Edition entitlement file so that you can use Platform Symphony MapReduce feature, as shown in Example 4-53 on page 172.

Example 4-53 Installing and configuring Platform Symphony

```
[root@compute002 Inst_Symphony]# su - egoadmin

Setup the environment variables.

[egoadmin@compute002 ~]$ . /opt/ibm/platformsymphony/profile.platform

[egoadmin@compute002 ~]$ egoconfig join compute000
```

To set the cluster to start at boot and to grant root privileges for the egoadmin account, run as root the `egosetrc.sh` and `egosetsudoers.sh` scripts, as shown in Example 4-54.

Example 4-54 Running root scripts to enable sudo for an autostart

```
[egoadmin@compute001 ~]$ su -
Password:
[root@compute001 ~]# . /opt/ibm/platformsymphony/profile.platform
[root@compute001 ~]# egosetrc.sh
egosetrc succeeds
[root@compute001 ~]# egosetsudoers.sh
egosetsudoers succeeds
```

Because we are using a simplified WEM, if you need cluster scalability (more than 1000 CPU and users), increase the number of open files that are permitted for each user, as shown in Example 4-55.

Example 4-55 Changing the nofile on the cluster

```
[root@compute002 ~]# vi /etc/security/limits.conf
[root@compute002 ~]# grep nofile /etc/security/limits.conf
#        - nofile - max number of open files
*       hard    nofile  6400
```

Start your Platform Symphony cluster by running the **`egosh ego start`** command from the command line, as shown in Example 4-56.

Example 4-56 Starting the cluster on the management node

```
[egoadmin@compute002 ~]$ egosh ego start
Start up LIM on <compute002> ...... done
```

4.2.7 Building an automatic kit template

In this section, we describe how to build a kit template to successfully deploy the software stack for the new nodes to be added to the cluster after the initial cluster configuration. The kit template is used by the PCM-SE image profile for the automatic deployment and configuration of GPFS, Hadoop, and Platform Symphony when new nodes are added to your cluster.

Complete the following steps to build a new kit from scratch:

1. From the management node of your PCM-SE cluster, go to the `/install/kits` directory and create a kit, as shown in Example 4-57 on page 173.

Example 4-57 Creating a sample kit

```
[root@mgt01 kits]#cd /install/kits
[root@mgt01 kits]# buildkit create kit-symphony-gpfs-hadoop
Kit template for kit-symphony-gpfs-hadoop created in
/install/kits/kit-symphony-gpfs-hadoop directory
[root@mgt01 kits]# cd kit-symphony-gpfs-hadoop/
[root@mgt01 kit-gpfs-base]# ls
buildkit.conf  docs  other_files  plugins  scripts  source_packages
```

2. Locate and edit the `buildkit.conf` file, as shown in Example 4-58.

Example 4-58 Example buildkit.conf file that is used for creating the automatic kit

```
kit:
   basename=kit-symphony-gpfs-hadoop
   description=description for kit-symphony-gpfs-hadoop
   version=1.0
   ostype=Linux
   kitdeployparams=sample/kitdeployparams.lst
   kitlicense=EPL

kitrepo:
   kitrepoid=rhels6.4
   osbasename=rhels
   osmajorversion=6
   osminorversion=4
   osarch=x86_64


kitcomponent:
    basename=kit-gpfs
    description=description for component kit-symphony-gpfs-hadoop_compute
    version=3.5.0
    release=11
    serverroles=compute
    kitrepoid=rhels6.4


kitcomponent:
    basename=kit-hadoop
    description=description for component kit-symphony-gpfs-hadoop_compute
    version=1.1
    release=1
    serverroles=compute
    kitrepoid=rhels6.4


kitcomponent:
    basename=kit-symphony
    description=description for component kit-symphony-gpfs-hadoop_compute
    version=6.1
    release=1
    serverroles=compute
    kitrepoid=rhels6.4
    kitcompdeps=kit-hadoop
    postbootscripts=sample/postboot.sh
```

```
kitpackage:
    filename=gpfs.base-3.5.0-3.x86_64.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.docs-3.5.0-3.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.gpl-3.5.0-3.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.msg.en_US-3.5.0-3.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.base-3.5.0-13.x86_64.update.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.docs-3.5.0-13.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.gpl-3.5.0-13.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=gpfs.msg.en_US-3.5.0-13.noarch.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=hadoop-1.1.1-1.x86_64.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=ibm-java-x86_64-sdk-6.0-8.0.x86_64.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes
```

```
kitpackage:
    filename=egocomp-lnx26-lib23-x64-1.2.8.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes

kitpackage:
    filename=soam-lnx26-lib23-x64-6.1.1.rpm
    kitrepoid=rhels6.4
    isexternalpkg=yes
```

3. Locate the `postboot.sh` script file that is in the `scripts/sample/` directory and add the text that is shown in Example 4-59.

Example 4-59 The postboot.sh sample file for installing and integrating the stack

```
##/bin/sh


#Gathering cluster information for the script
PCMMASTERIP=`grep ^NFSSERVER= /var/log/xcat/xcat.log|cut -d= -f2`
PCMMASTERHOST=`grep "${PCMMASTERIP} " /etc/hosts |awk '{print $2}'`
echo PCM Master host is $PCMMASTERHOST
MYNODEHOST=`hostname`
echo Hostname is $MYNODEHOST
#Waiting for /etc/hosts syncronization

GPFSMASTERHOST=`grep -v localhost /etc/hosts |awk '{print $2} '|grep -v
${MYNODEHOST} |grep -v ${PCMMASTERHOST} |head -1 |xargs -I {} ssh {} mmlscluster
|grep Primary |cut -d: -f2 |sed 's/\ //g'`

echo GPFS Master host is $GPFSMASTERHOST

SYMPMASTERHOST=`grep -v localhost /etc/hosts |awk '{print $2} '|grep -v
${MYNODEHOST} |grep -v ${PCMMASTERHOST}|head -1 |xargs -I {} ssh {} 'su - egoadmin
-c "egosh ego info"' |grep "EGO master host name"|cut -d: -f2 |sed 's/\ //g'`

echo "running gpfs install postboot script"

# First we check for GPFS installation
if [ -f /usr/lpp/mmfs/bin/mmfs ]
 then
        echo GPFS Already Installed
 else
        #If not installed then we install GPFS it in the correct order, first the
base, then the update and portability layer
        yum install -y gpfs.base-3.5.0-3 gpfs.docs-3.5.0-3 gpfs.gpl-3.5.0-3
gpfs.msg.en_US-3.5.0-3
        yum install -y gpfs.base-3.5.0-13 gpfs.docs-3.5.0-13
gpfs.gplbin-2.6.32-358.el6.x86_64.x86_64 gpfs.gpl-3.5.0-13 gpfs.msg.en_US-3.5.0-13

        #This command will add the new node to the existing GPFS cluster
        ssh ${GPFSMASTERHOST} mmaddnode -N ${MYNODEHOST}
        ssh ${GPFSMASTERHOST} mmchlicense server --accept -N ${MYNODEHOST}
```

```
        #Now we create the stanza file for the disks
        #This expression will calculate the next failuregroup for GPFS
        let FAILUREGROUP=`ssh ${GPFSMASTERHOST} mmlsdisk bigdatafs |awk '{print
$4}' |tail -1|cut -d, -f1`+1

        #We input the head of the stanza file
        echo "%pool:
pool=system
blockSize=1024K
layoutMap=cluster
allowWriteAffinity=no

%pool:
pool=datapool
blockSize=1024K
layoutMap=cluster
allowWriteAffinity=yes
writeAffinityDepth=1
blockGroupFactor=128" >/tmp/${MYNODEHOST}.stanza

        #In this environment we will have dm devices for disks so we will iterate
on all dm disks
        # other options to use on this next iteration would be like: "cat
/proc/partitions|awk '{print $4}'|grep ^sd |grep -v sda|xargs echo" for all sd
disks except sda

        for GPFSDISKS in `cat /proc/partitions|awk '{print $4}'|grep ^dm|xargs
echo`
        do
                echo "" >>/tmp/${MYNODEHOST}.stanza
                echo "%nsd:  device=/dev/${GPFSDISKS} servers=${MYNODEHOST}
usage=dataOnly failureGroup=${FAILUREGROUP},0,0 pool=datapool"
>>/tmp/${MYNODEHOST}.stanza
        done



        #This step will copy the file we just created to the GPFS Management Host
so we can create the nsd in the cluster and add disks to the filesystem
        scp /tmp/${MYNODEHOST}.stanza ${GPFSMASTERHOST}:/tmp/
        ssh ${GPFSMASTERHOST} mmcrnsd -v no -F /tmp/${MYNODEHOST}.stanza
        /usr/lpp/mmfs/bin/mmstartup
        mount /mapred
        ssh ${GPFSMASTERHOST} mmadddisk bigdatafs -F /tmp/${MYNODEHOST}.stanza -v
no
        grep '/usr/lpp/mmfs/bin' /etc/bashrc >>/dev/null
        if [ $? -eq 1 ]
                then
                echo export PATH=\$PATH:/usr/lpp/mmfs/bin >>/etc/bashrc
        fi


fi
```

```
echo "running hadoop postboot script"
yum -y install hadoop
yum -y install ibm-java-x86_64-sdk

############
##This Section will build mapred-site.xml file to comply with GPFS-FPO
echo Building mapred-site.xml

ssh ${SYMPMASTERHOST} "cat /etc/hadoop/mapred-site.xml" | sed
"s/\/local\/${SYMPMASTERHOST}/\/local\/${MYNODEHOST}/g"
>/etc/hadoop/mapred-site.xml

############
##This Section will build mapred-site.xml file to comply with GPFS-FPO
echo Building core-site.xml

ssh ${SYMPMASTERHOST} "cat /etc/hadoop/core-site.xml" | sed
"s/\/local\/${SYMPMASTERHOST}/\/local\/${MYNODEHOST}/g" >/etc/hadoop/core-site.xml

############
##This section will create links from GPFS-FPO and environmental variables needed
for Hadoop
export HADOOP_HOME=/usr/share/hadoop
ln -sf /usr/lpp/mmfs/fpo/hadoop-1.1.1/*.jar $HADOOP_HOME/lib
ln -sf /usr/lpp/mmfs/fpo/hadoop-1.1.1/libgpfshadoop.64.so /usr/lib64/
grep 'JAVA_HOME=/opt/ibm/java-x86_64-60' /etc/hadoop/hadoop-env.sh >>/dev/null
if [ $? -eq 1 ]
then
echo export JAVA_HOME=/opt/ibm/java-x86_64-60/ >>/etc/hadoop/hadoop-env.sh
fi
mkdir $HADOOP_HOME/hadoop-datastore
grep 'JAVA_HOME=/opt/ibm/java-x86_64-60' /etc/bashrc >>/dev/null
if [ $? -eq 1 ]
then
echo export JAVA_HOME=/opt/ibm/java-x86_64-60 >>/etc/bashrc
fi
grep 'HADOOP_HOME=/usr/share/hadoop' /etc/bashrc >>/dev/null
if [ $? -eq 1 ]
then
echo export HADOOP_HOME=/usr/share/hadoop >>/etc/bashrc
fi
grep 'LD_LIBRARY_PATH=/usr/lib64' /etc/bashrc >>/dev/null
if [ $? -eq 1 ]
then
echo export LD_LIBRARY_PATH=/usr/lib64 >>/etc/bashrc
fi
grep 'HADOOP_VERSION=1_1_1' /etc/bashrc >>/dev/null
if [ $? -eq 1 ]
then
echo export HADOOP_VERSION=1_1_1 >>/etc/bashrc
fi

############
##This section will add the new slave to the hadoop cluster
ssh ${SYMPMASTERHOST} grep ${MYNODEHOST} /etc/hadoop/slaves >>/dev/null
```

```
if [ $? -eq 1 ]
then
ssh ${SYMPMASTERHOST} "echo ${MYNODEHOST} >>/etc/hadoop/slaves"
fi



echo "running Symphony postboot script"

#This step will create the cluster admin user
groupadd egoadmin
useradd -g egoadmin egoadmin
echo ibm01ibm | passwd --stdin egoadmin

#Now we define the cluster user admin and install variables
export CLUSTERADMIN=egoadmin
export JAVA_HOME=/opt/ibm/java-x86_64-60/
export HADOOP_HOME=/usr/share/hadoop
export LD_LIBRARY_PATH=/usr/lib64
export HADOOP_VERSION=1_1_1


#Now we will install Symphony on the node and configure it.
yum install -y egocomp-lnx26-lib23-x64
yum install -y soam-lnx26-lib23-x64


#We will now join to the existing cluster, start the server and configure simphony
for the Platform Symphony MapReduce with GPFS
su - egoadmin -c ". /opt/ibm/platformsymphony/profile.platform ; egoconfig join
${SYMPMASTERHOST} -f"
. /opt/ibm/platformsymphony/profile.platform
egosetrc.sh
egosetsudoers.sh
echo ". /opt/ibm/platformsymphony/profile.platform" >>/home/egoadmin/.bashrc
su - egoadmin -c ". /opt/ibm/platformsymphony/profile.platform ; egosh ego start"
ln -fs /usr/lpp/mmfs/fpo/hadoop-1.1.1/*.jar $PMR_SERVERDIR/../lib
ln -fs /usr/lpp/mmfs/fpo/hadoop-1.1.1/hadoop-1.1.1-gpfs.jar
/opt/ibm/platformsymphony/soam/mapreduce/6.1.1/linux2.6-glibc2.3-x86_64/lib/hadoop
-1.1.1/
ln -fs /usr/lpp/mmfs/fpo/hadoop-1.1.1/libgpfshadoop.64.so
/opt/ibm/platformsymphony/soam/mapreduce/6.1.1/linux2.6-glibc2.3-x86_64/lib/libgpf
shadoop.so
ln -fs /usr/lpp/mmfs/lib/libgpfs.so /usr/lib/libgpfs.so
ln -fs /etc/hadoop/core-site.xml
/opt/ibm/platformsymphony/soam/mapreduce/conf/core-site.xml
ssh ${SYMPMASTERHOST} "cat
/opt/ibm/platformsymphony/soam/mapreduce/conf/pmr-site.xml" |sed
"s/\/local\/${SYMPMASTERHOST}/\/local\/${MYNODEHOST}/g"
>/opt/ibm/platformsymphony/soam/mapreduce/conf/pmr-site.xml

#Now we exchange user ssh key to
mkdir -p /home/egoadmin/.ssh
```

```
scp ${SYMPMASTERHOST}:/home/egoadmin/.ssh/id_dsa /home/egoadmin/.ssh/id_dsa
scp ${SYMPMASTERHOST}:/home/egoadmin/.ssh/authorized_keys
/home/egoadmin/.ssh/authorized_keys
chown -R egoadmin:egoadmin /home/egoadmin/.ssh
chmod 700 /home/egoadmin/.ssh
chmod 600 /home/egoadmin/.ssh/id_dsa
chmod 600 /home/egoadmin/.ssh/authorized_keys
```

4. Build the kit, as shown in Example 4-60.

Example 4-60 Building the kit repository

```
[root@mgt01 kit-symphony-gpfs-hadoop]# buildkit buildrepo rhels6.4
Spawning worker 0 with 3 pkgs
Workers Finished
Gathering worker results

Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete
```

5. Create the `.tar` file, as shown in Example 4-61.

Example 4-61 Creating the kit tar file

```
[root@mgt01 kit-symphony-gpfs-hadoop]# buildkit buildtar
Kit tar file
/install/kits/kit-symphony-gpfs-hadoop/build/kit-symphony-gpfs-hadoop-1.0-Linux
.NEED_PRODUCT_PKGS.tar.bz2 successfully built
```

6. Add all of packaged rpm files on the buildkit on one directory, then add the packages to the `.tar` file, as shown in Example 4-62.

Example 4-62 Adding the packages to the tar file

```
[root@mgt01 kit-symphony-gpfs-hadoop]# ls /tmp/computerpm
egocomp-lnx26-lib23-x64-1.2.8.rpm     gpfs.docs-3.5.0-13.noarch.rpm
gpfs.gpl-3.5.0-3.noarch.rpm
gpfs.msg.en_US-3.5.0-3.noarch.rpm       soam-lnx26-lib23-x64-6.1.1.rpm
gpfs.base-3.5.0-13.x86_64.update.rpm  gpfs.docs-3.5.0-3.noarch.rpm
gpfs.gplbin-2.6.32-358.el6.x86_64-3.5.0-13.x86_64.rpm
hadoop-1.1.1-1.x86_64.rpm
gpfs.base-3.5.0-3.x86_64.rpm          gpfs.gpl-3.5.0-13.noarch.rpm
gpfs.msg.en_US-3.5.0-13.noarch.rpm
ibm-java-x86_64-sdk-6.0-8.0.x86_64.rpm
[root@mgt01 kit-symphony-gpfs-hadoop]# buildkit addpkgs -p /tmp/computerpm
/install/kits/kit-symphony-gpfs-hadoop/build/kit-symphony-gpfs-hadoop-1.0-Linux
.NEED_PRODUCT_PKGS.tar.bz2
Spawning worker 0 with 16 pkgs
Workers Finished
Gathering worker results

Saving Primary metadata
Saving file lists metadata
Saving other metadata
```

```
Generating sqlite DBs
Sqlite DBs complete
Kit tar file
/install/kits/kit-symphony-gpfs-hadoop/kit-symphony-gpfs-hadoop-1.0-Linux.tar.b
z2 successfully built
```

Note: Before RPM packages are added to a kit template, ensure that the packages permission are at least set to 644 (RW-R--R--); otherwise, you experience permission issues during the nodes deployment.

4.2.8 Adding the kit to an image profile

Log in to PCM as shown in Example 4-10 on page 146. For more information about how to add the kit to an image profile, see 4.2.8, “Adding the kit to an image profile” on page 180.

4.2.9 Testing a Platform Symphony MapReduce job

When the cluster implementation is complete, it is time to test a Platform Symphony MapReduce job. In this section, we describe how to submit a sample Platform Symphony MapReduce test job into a Platform Symphony environment. Complete the following steps:

1. Open the Platform Symphony web-based interface by using the following address:

   `http://<<hostname/ip_addr>>:8080/platform/`

2. Log in by using the user/password default information (admin/admin) as shown in Figure 4-11.

Figure 4-11 IBM Platform Symphony login window

3. After you are logged in, click **Workload** → **MapReduce** → **Jobs**, as shown in Figure 4-12.

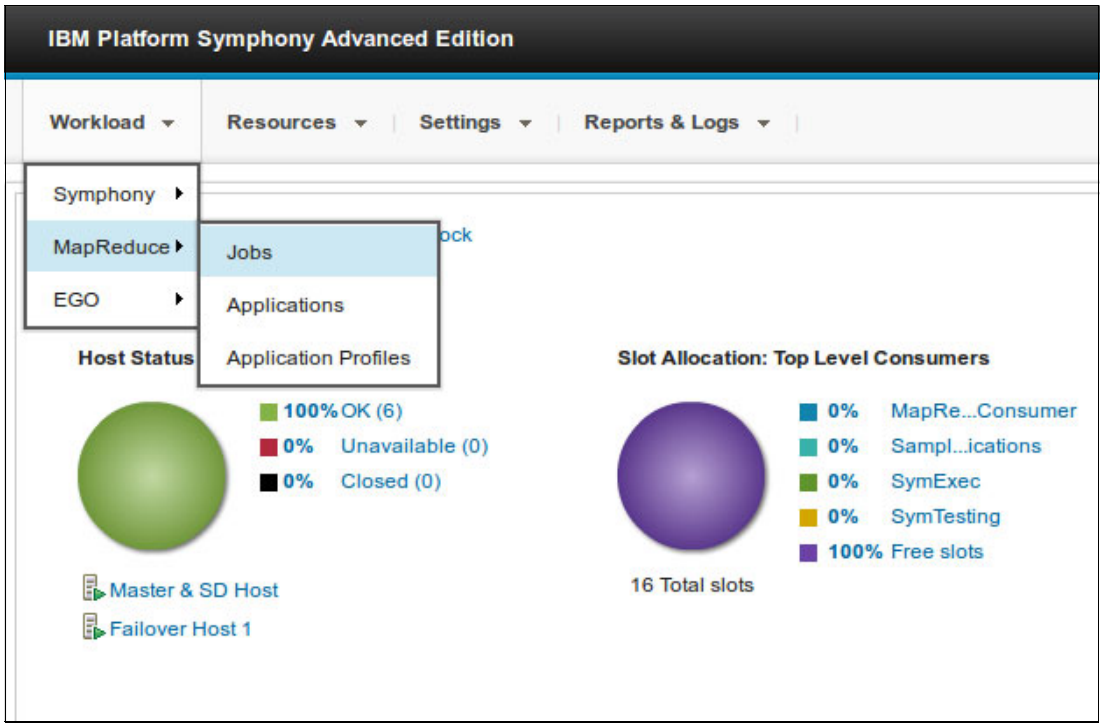

Figure 4-12 New job submission menu

4. In the Submit Job window, set the Main Class field as `wordcount` and Main Class Options as `gpfs:///input gpfs:///output`, as shown in Figure 4-13.

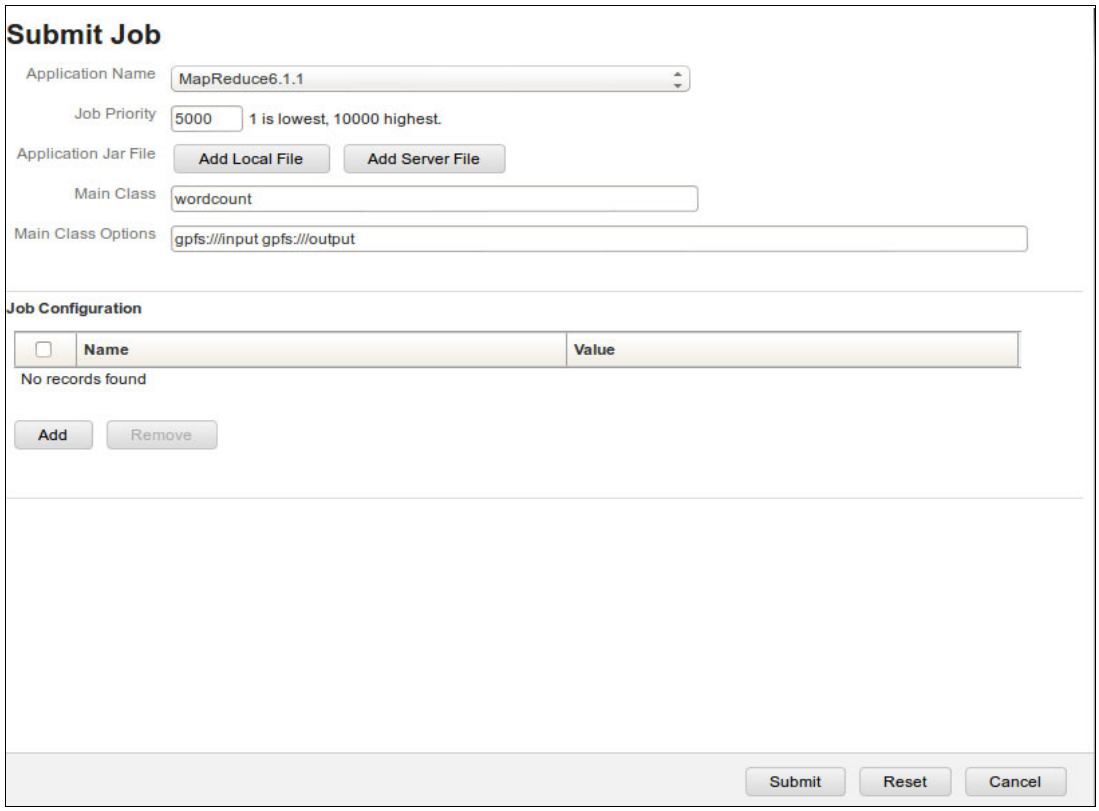

Figure 4-13 Submit Job window

5. Click **Add Server File** → **hadoop-examples-1.1.1.jar**, as shown in Figure 4-14. Click **OK**.

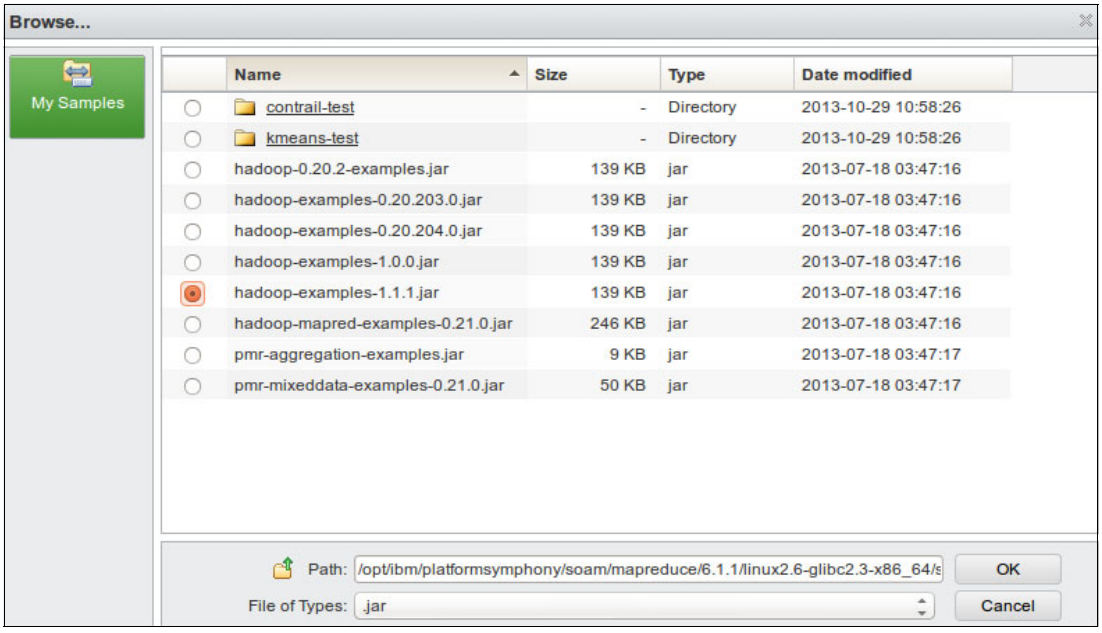

Figure 4-14 Selecting the .jar file

6. Click **Submit** and follow the job running progress, as shown in Figure 4-15.

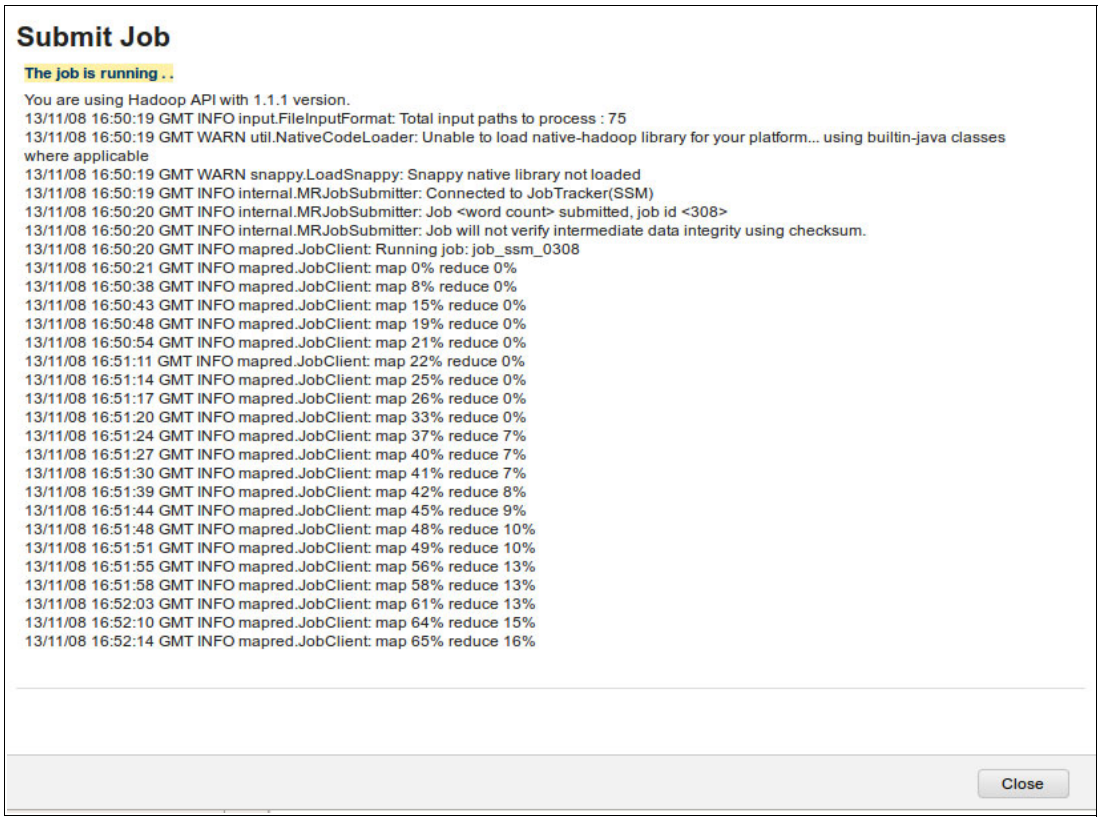

Figure 4-15 MapReduce sample job progress

When the job completes, review the run information in the job list, as shown in Figure 4-16.

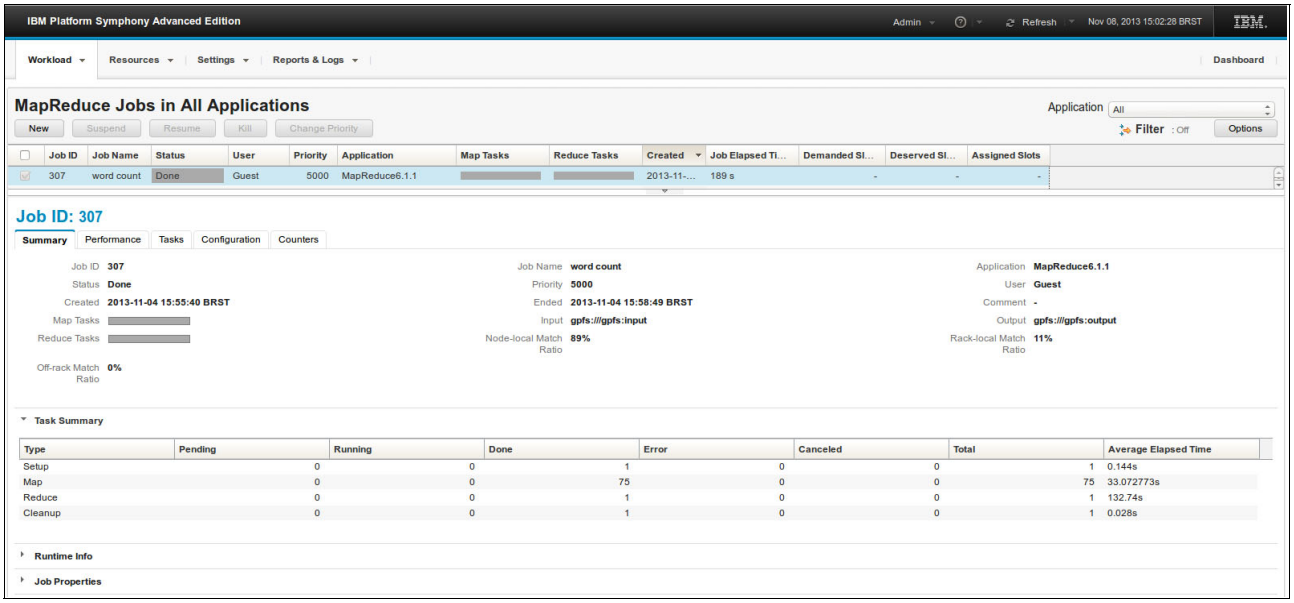

Figure 4-16 Run jobs history

Note: An alternative way to submit a sample job is by using the following command line:

```
mrsh jar
/opt/ibm/platformsymphony/soam/mapreduce/6.1.1/linux2.6-glibc2.3-x86_64/samples
/hadoop-examples-1.1.1.jar wordcount gpfs:///mapred/input gpfs:///mapred/output
```

Related publications

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

IBM Redbooks

The following IBM Redbooks publications provide more information about the topics in this book. Some publications that are referenced in this list might be available in softcopy only:

- *IBM Platform Computing Integration Solutions*, SG24-8081
- *IBM Technical Computing Clouds*, SG24-8144
- *Big Data Networked Storage Solution for Hadoop*, REDP-5010
- *Implementing the IBM General Parallel File System (GPFS) in a Cross Platform Environment,* SG24-7844
- *IBM NeXtScale System Planning and Implementation Guide*, SG24-8152
- *Implementing an IBM System x iDataPlex Solution*, SG24-7629-04
- *IBM ILOG Visualization Integration Scenarios,* SG24-7917
- *Addressing Data Volume, Velocity, and Variety with IBM InfoSphere Streams V3.0*, SG24-8108

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

`http://www.ibm.com/redbooks`

Other publications

The following publications also are relevant as further information sources:

- *IBM Platform HPC, Version 4.1.1 Fix Pack 1, Administration Guide*, SC27-6106
- *IBM Platform Cluster Manager Standard Edition, Version 4.1.1 Fix Pack 1, Administration Guide*, SC27-6104
- *Platform Cluster Manager Advanced Edition Version 4 Release 1: Administering*, SC27-4760
- *Platform LSF Version 9 Release 1.1, Foundations*, SC27-5304
- *Platform LSF Version 9 Release 1.1, Administering Platform LSF*, SC27-5302
- *Platform Application Center Version 9 Release 1, Administering Platform Application Center*, SC22-5396
- *Platform Symphony Version 6 Release 1.1: Platform Symphony Foundations,* SC27-5065
- *Platform Symphony Version 6 Release 1.1: User Guide for the MapReduce Framework,* GC27-5072

- *Platform Symphony Version 6 Release 1.1: MultiCluster User Guide,* SC27-5083
- *IBM Platform Symphony 6.1.1: Supported System Configurations*, SC27-5373

Online resources

The following websites also are relevant as further information sources:

- IBM Platform Computing Products Page:

  http://www.ibm.com/systems/technicalcomputing/platformcomputing/products/index.html

- IBM Application Ready Solutions:

  http://www.ibm.com/technicalcomputing/appready

- IBM Platform Product Libraries:

  http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.platform_product_libraries.doc/platform_product.htm

- General Parallel File System (GPFS) Wiki Page:

  https://www.ibm.com/developerworks/community/wikis/home/wiki/General%20Parallel%20File%20System%20(GPFS)/page/GPFS%20Wiki?lang=en

- IBM Platform Symphony Wiki - Focus on BigData:

  https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20Platform%20Symphony%20Wiki/page/Focus%20on%20Big%20Data

- IBM System x and Cluster Solutions configurator (x-config):

  http://www.ibm.com/products/hardware/configurator/americas/bhui/asit/index.html

- IBM NeXtScale System:

  http://www.ibm.com/systems/x/hardware/highdensity/nextscale/index.html

- IBM System x iDataPlex:

  http://www.ibm.com/systems/x/hardware/highdensity/dx360m4/

- IBM Intelligent Cluster:

  http://www.ibm.com/systems/x/hardware/highdensity/cluster/index.html

- IBM X6 enterprise servers:

  http://www.ibm.com/systems/x/x6/index.html

- High volume systems:

  - http://www.ibm.com/systems/x/hardware/rack/x3650m4hd/index.html
  - http://www.ibm.com/systems/x/hardware/rack/x3650m4bd/index.html

- IBM Platform Cluster Manager Standard Edition (PCM-SE) Installation Guide:

  http://publibfp.dhe.ibm.com/epubs/pdf/c2761070.pdf

- IBM General Parallel File System (GPFS) FAQ:

  http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.doc%2Fgpfs_faqs%2Fgpfsclustersfaq.html

- Building the GPFS portability layer on Linux nodes:

  http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.v3r5.gpfs300.doc%2Fbl1ins_bldgpl.htm

Help from IBM

IBM Support and downloads:

http://www.ibm.com/support

IBM Global Services:

http://www.ibm.com/services

IBM Redbooks

IBM Platform Computing Solutions Reference Architectures and Best Practices

(0.2”spine)
0.17”<->0.473”
90<->249 pages

IBM Platform Computing Solutions Reference Architectures and Best Practices

Helps with the foundation to manage enterprise environments

Delivers reference architectures and best practices guides

Provides case scenarios

This IBM Redbooks publication demonstrates and documents that the combination of IBM System x, IBM GPFS, IBM GPFS-FPO, IBM Platform Symphony, IBM Platform HPC, IBM Platform LSF, IBM Platform Cluster Manager Standard Edition, and IBM Platform Cluster Manager Advanced Edition deliver significant value to clients in need of cost-effective, highly scalable, and robust solutions. IBM depth of solutions can help the clients plan a foundation to face challenges in how to manage, maintain, enhance, and provision computing environments to, for example, analyze the growing volumes of data within their organizations.

This IBM Redbooks publication addresses topics to educate, reiterate, confirm, and strengthen the widely held opinion of IBM Platform Computing as the systems software platform of choice within an IBM System x environment for deploying and managing environments that help clients solve challenging technical and business problems.

This IBM Redbooks publication addresses topics to that help answer customer's complex challenge requirements to manage, maintain, and analyze the growing volumes of data within their organizations and provide expert-level documentation to transfer the how-to-skills to the worldwide support teams.

This IBM Redbooks publication is targeted toward technical professionals (consultants, technical support staff, IT Architects, and IT Specialists) who are responsible for delivering cost-effective computing solutions that help optimize business results, product development, and scientific discoveries.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

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

For more information:
ibm.com/redbooks

SG24-8169-00 ISBN 0738439479