# xCAT 2 Guide for the CSM System Administrator

- xCAT architecture overview
- xCAT 2 quick deployment example
- CSM to xCAT transition scenarios

Octavian Lascu
Andrey Brindeyev
Dino E. Quintero
Velmayil Sermakkani
Robert Simon
Timothy Struble



ibm.com/redbooks

Redpaper

International Technical Support Organization

# xCAT 2 Guide for the CSM System Administrator

November 2008



REDP-4437-00

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



**First Edition (November 2008)**

This edition applies to Version 2, Release 0, Extreme Cluster Administration Toolkit (xCAT).

# 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 give 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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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.

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:

| | | |
|---|---|---|
| 1350™ | iDataPlex™ | Redbooks® |
| AIX® | Passport Advantage® | Redbooks (logo)  ® |
| BladeCenter® | POWER™ | RS/6000® |
| Blue Gene® | Power Architecture® | System p® |
| eServer™ | POWER Hypervisor™ | System x™ |
| General Parallel File System™ | Power Systems™ | WebSphere® |
| GPFS™ | POWER5™ | xSeries® |
| HACMP™ | POWER6™ | |
| IBM® | PowerVM™ | |

The following terms are trademarks of other companies:

SUSE, the Novell logo, and the N logo are registered trademarks of Novell, Inc. in the United States and other countries.

VMware, the VMware "boxes" logo and design are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions.

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

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

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, other countries, or both.

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

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

# Preface

This IBM® Redbooks® publication positions the new Extreme Cluster Administration Toolkit 2.x (xCAT 2) against the IBM Cluster Systems Management (CSM) for IBM Power Systems™ and IBM System x™ in a High Performance Computing (HPC) environment.

This paper provides information to help you:

- Understand, from a broad perspective, a new clustering management architecture. The paper emphasizes the benefits of this new solution for deploying HPC clusters of large numbers of nodes.
- Install and customize the new xCAT cluster management in various configurations.
- Design and create a solution to migrate from existing CSM configurations to xCAT-managed clusters for various IBM hardware platforms.

## The team that wrote this paper

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

**Octavian Lascu** is a Project Leader associated with the ITSO, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on all areas of IBM System p® and Linux® clusters. His areas of expertise include High Performance Computing, Blue Gene® and Clusters. Before joining ITSO, Octavian worked in IBM Global Services Romania as a software and hardware Services Manager. He holds a Masters degree in Electronic Engineering from the Polytechnical Institute in Bucharest, and is also an IBM Certified Advanced Technical Expert in AIX/PSSP/HACMP™. He has worked for IBM since 1992.

**Andrey Brindeyev** is a Senior IT Specialist in IBM Russia. He has been with IBM since 2005. He acts as a Presales Specialist for STG and as a Software Engineer for GTS for delivering service projects for IBM Russia. Prior to that role, he was a Technical Consultant in joint initiative between the IBM and Intel® Energy Competence Center for the oil and gas industry for System x products, with a major focus on IBM System Cluster 1350™ solution. His areas of expertise are the IBM BladeCenter®, System x servers, VMware®, and Linux. He graduated from the Moscow State Institute of Radioengineering, Electronics

and Automation and holds a Masters degree in Applied Mathematics. He is a Senior Accredited IT Specialist.

**Dino E. Quintero** is an IBM Senior Certified IT Specialist in the Worldwide Technical Support Marketing for the IBM System Blue Gene Solution and IBM HPC Software. Before joining the Deep Computing team, he was a Clustering Solutions Project Leader for the IBM ITSO. His areas of expertise include enterprise backup and recovery, disaster recovery planning and implementation, and clustering architecture and solutions. He is a Certified Specialist on System p Administration, System p Clustering, High Availability. He is a Master Certified IT Specialist by the Open Group. Currently, he focuses on planning, influencing, leading, managing, and marketing the IBM Blue Gene and the IBM HPC Software Solutions. He also delivers technical lectures worldwide.

**Velmayil Sermakkani** is an HPC Specialist at India Software Lab (ISL), working in the High Performance Computing Group and STG Lab Services. He has 12 years of experience, joining IBM in 2007. He currently provides support for HPC Benchmark clusters on System p and System x. He has implemented several GPFS™, CSM clusters and provided worldwide customer training in HACMP, GPFS, and CSM. Prior to that he worked as a Consultant for IBM Austin and Poughkeepsie, and as a Technical Tester and Developer for GPFS, CSM, xCAT products in Poughkeepsie. He holds a degree in Computer Science and Engineering from Manonmaniam Sundaranar University, India.

**Robert (Bob) Simon** is a Senior Software Engineer in STG working in Poughkeepsie, New York. He has worked with IBM since 1987. He currently is a Team Leader in the Software Technical Support Group, which supports the High Performance Clustering software (LoadLeveler, CSM, GPFS, RSCT and PPE). He has extensive experience with IBM System p hardware, AIX®, HACMP, and High Performance Clustering software. He has participated in the development of two other IBM Redbooks.

**Timothy (Tim) Struble** is the Team Lead of the Level-2 CSM and xCAT Service Team in Poughkeepsie, New York, a position he has held for five years. Prior to that he was the Level-3 Team Lead. He has been in the service organization for 16 years and has worked extensively with cluster management software dating back to RS/6000® SP. Tim graduated from Syracuse University with a Bachelors degree in Computer Science in 1989 and joined IBM that same year.

Thanks to the following person for his contribution to this project:

David Watts
International Technical Support Organization, Raleigh Center

Thanks to the following people, from IBM Poughkeepsie, for their contributions:

Brian Croswell
Egan Ford
Linda Mellor
Ganesan Narayanasamy
Bruce M Potter
Scot Sakolish

## Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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

**ibm.com**/redbooks/residencies.html

## Comments welcome

Your comments are important to us!

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

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

  **ibm.com**/redbooks

- Send your comments in an e-mail 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

1

# Introduction

Extreme Cluster Administration Toolkit 2.x (xCAT 2) is a scalable distributed computing management and provisioning tool that provides a unified interface for hardware control, discovery, and operating system diskful and diskless deployment. Since 1999, xCAT has been used for deploying and managing large Linux systems. Many IBM customers and others have enjoyed its powerful customizable architecture long before other management solutions came into existence.

As an open source management tool, xCAT 2 benefits from the efforts of collaborative development by an open community that brings leading-edge architectural design you can use with confidence.

Highlights include:

- Licensed under Eclipse Public License Agreement
- Client/server architecture
- Role-based administration
- Stateless and iSCSI support
- Scalability
- Automatic discovery
- Plug-in architecture for:
  - Notification
  - Monitoring (RMC/RSCT, SNMP and others)
- Centralized console and system logs

## 1.1 Open source xCAT 2: For HPC clusters management

IBM developed xCAT 2 as an open source initiative to support deployment of large High Performance Computing (HPC) clusters based on various hardware platforms. Besides the performance requirements, the modern HPC environments must also be flexible, easy to administer, and highly reliable. In addition, power consumption for such large environments is a restricting factor.

Each cluster component has a specific reliability, usually expressed as the mean time between failures (MTBF). Cluster nodes are, in fact, computers that have memory, processors, I/O, fans and cooling devices, and persistent storage. All these components affect the overall node (and implicit cluster) reliability. Thus, by reducing the number of components in each node, we can improve the overall cluster reliability and power consumption.

One of the components that is, by nature, less reliable and also *power hungry* is the persistent storage, usually a hard disk. The hard disk stores a copy of the operating system and is used mainly to boot the nodes and provide temporary storage for job and operating system data. If this component has to be eliminated, we must be able to provide the same functionality to the cluster: operating system (OS) load source and temporary file system space for jobs and OS data.

In this case, a copy of the OS can be provided to the nodes through the network, and temporary files can be stored either in each node's RAM or on a remote file system, accessible also through the network.

In addition to power and reliability benefits, this approach to storage also provides administrative flexibility, eliminating the necessity to deploy and maintain multiple copies of the operating system (one for each individual node). Deploying OS patches, fixes, or customized kernels becomes an easy, manageable task because these operations have to be performed only on the centralized repository (mainly on one OS copy).

xCAT 2 supports this approach for deploying HPC clusters by also supporting diskful and diskless (both stateful and stateless) nodes.

In addition, xCAT has a modular approach to cluster management and monitoring, allowing system administrators to customize the *degree of intrusion* (and thus the amount of processing consumed for something other than running jobs) of the cluster management software stack.

Because of its modular architecture, xCAT enables the plugging in of various components, allowing for rapid integration of new hardware and management software.

For xCAT 2 documentation, see the following Web page:

http://xcat.svn.sourceforge.net/svnroot/xcat/xcat-core/trunk/xCAT-client/share/doc

xCAT 2 offers scalability through a single management node, which can have any number of stateless service nodes to increase the provisioning throughput and management of large clusters. IBM Support for xCAT can help keep those clusters running with added ease. IBM Support for xCAT focuses on customers who are interested in using pure open source technology with the comfort of having IBM support available when they want it. Also, IBM Support enables customers to optimize the value they get from the open source community and IBM, giving them unparalleled choice of software and support.

### 1.1.1 For CSM System Administrators

Cluster Systems Management (CSM) is the licensed IBM product used in managing clusters for AIX and Linux. CSM is a very powerful, highly integrated system management and monitoring product that provides a centralized control interface for large computer clusters for commercial and HPC environments.

However, for HPC environments, the best features of CSM and xCAT 1 are being combined into xCAT 2. For example, xCAT 2 combines IBM System x hardware configuration from xCAT, and documentation and SLP discovery from CSM.

The cluster management experts within IBM have pooled their expertise to produce new leading-edge capabilities in xCAT 2, such as:

- Extreme scalability through automatic hierarchy
- Compressed, stateless nodes for efficient operation and ease of management
- Virtually all of the capabilities of CSM, plus much more
- Support for more hardware and operating systems
- Administrator and operator roles
- Pluggable database support, with SQL reporting capabilities
- Power levels management for green computing
- Web interface

The open source nature of xCAT 2 allows additional contributors, not from IBM, to improve xCAT 2 even further, and to increase the adoption of xCAT throughout the industry as a recognized leader in cluster management. Although xCAT 2 can likely run on hardware that is not from IBM, its support for IBM hardware is more thorough and complete.

CSM will continue to be supported for existing and new IBM Power Systems System p hardware, AIX, and Linux on System p releases throughout the life of the entire POWER6™ program.

Existing CSM customers do not have to migrate current CSM deployments to xCAT 2. IBM can analyze and suggest whether xCAT 2 or an alternative is right for a new cluster deployment.

xCAT 2 targets the traditional HPC market, and therefore should not be considered as a full replacement for CSM. However, xCAT is open source, and greatly increasing its scalability and capability can well serve the HPC market.

### 1.1.2 Distinction between IBM Director and xCAT 2

Although IBM is moving CSM expertise to strategic products such as IBM Director and xCAT, we must maintain a clear distinction between xCAT 2 and IBM Director:

- xCAT 2 runs parallel jobs on large clusters. This is an open source product.
- IBM Director is involved in commercial, small and medium business (SMB), server consolidation, and others. This is an IBM product.

We have the opportunity to make xCAT 2 the exploratory branch of IBM Director, by moving new features pioneered in xCAT into Director.

## 1.2 Overview of xCAT 2 architecture

The heart of the xCAT architecture is the xCAT daemon (xcatd) running on the management node, shown in Figure 1-1 on page 5.

The daemon receives requests from the client, validates the requests, and then invokes the operation. The xcatd daemon also receives status and inventory information from the nodes as they are being discovered, installed, or booted.

xCAT 2 is not an evolution or a modified version of xCAT 1.x. Rather, xCAT 2 is a revolutionary new project created by the best cluster management developers at IBM. The team consists of IBM developers from CSM, xCAT 1.x, System x, System p, Cluster 1350, and iDataPlex™ technologies.

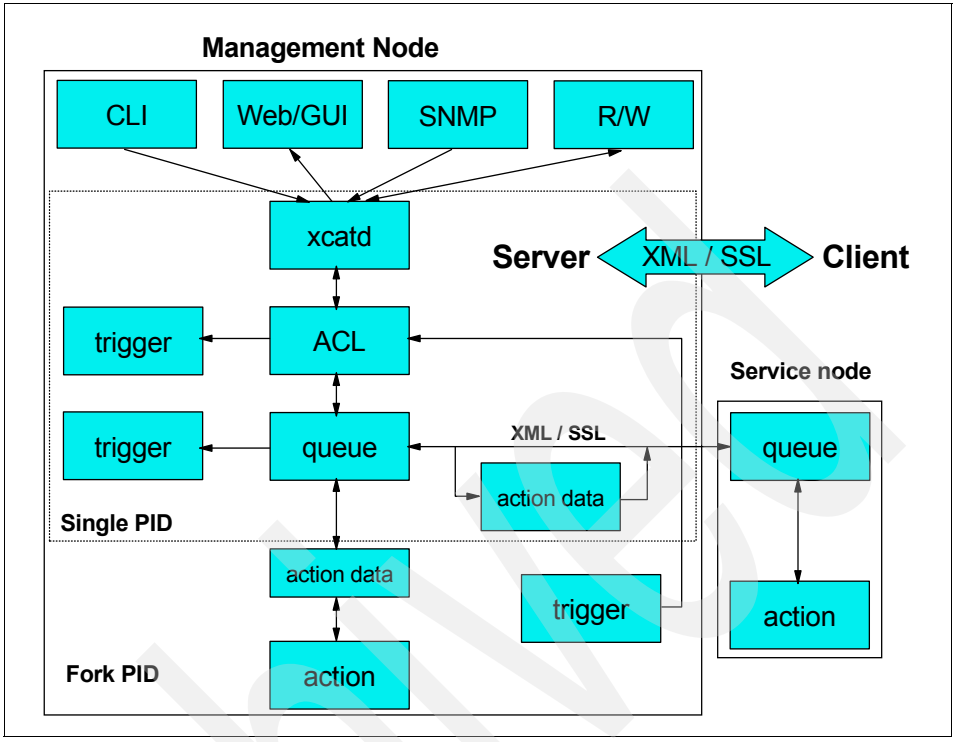


*Figure 1-1 xCAT 2 architecture*

## 1.2.1 Overview of architecture features

Features of the xCAT architecture are provided in the following list. For details, see 2.2.1, “xCAT features” on page 23.

- xCAT 2 is implemented using client/server architecture.
- Role-based administration allows assignment of various administrative roles.
- xCAT 2 supports stateless and iSCSI nodes.
- Designed for scalability, the xCAT 2 cluster can be built with 100k and more nodes by using the Hierarchical Management Cloud (HMC).
- Automatic node configuration discovery helps ease integration.
- xCAT 2 supports data storage in a database, and you have choices of database programs (back-end) such as SQLite, Postgresql, MySQL™, and future choices.
- Flexible monitoring infrastructure so you can easily integrate vendor monitoring software into the xCAT cluster.

- Default monitoring that uses SNMP trap handlers to handle all SNMP traps.
- xCAT provides a centralized console and systems logs.

### 1.2.2 Operating system and distributions supported

The following distributions and operating systems are supported:

- Multiple distribution and OS support include AIX, SLES, openSUSE, RHEL, CentOS, Scientific Linux, Fedora Core, and Windows® through imaging. Mixed operating systems are allowed in a single cluster.
- Stateless diskless nodes including ramfs root, compressed ramfs root, and NFS root with ramfs overlay support (SLES 10, RHEL 5, CentOS 5, Fedora8, and experimental Fedora 9 support).
- Stateful diskless using iSCSI for SLES10, Fedora8, RHEL5, and CentOS5.
- Traditional local disk and SAN provisioning using native deployment methods of SLES10, RHEL4, RHEL5, CentOS4, CentOS5, Fedora 8, and Fedora 9.

**Note:** The list of supported distributions and operating systems is updated regularly.

### 1.2.3 Hardware supported

The following hardware is supported:

- IBM BladeCenter (including HS21, HS21 XM, LS21, LS41, QS21, QS22, JS21, JS22)
- IBM System x (x3455, x3550, x3650, x3655, x3755, and more)
- IBM Power System p (including Cell)
- iDataplex
- Machines based on the Intelligent Platform Management Interface (IPMI)

**Note:** The list of hardware supported is updated regularly.

### 1.2.4 Hardware control features

The hardware control features include:

- Power control
- Event logs

- Boot device control (full boot sequence on IBM System BladeCenter, next boot device on other systems)
- Sensor readings (Temperature, Fan speed, Voltage, Current, and fault indicators as supported by systems)
- Collection of MAC addresses
- LED status and modification (ability to identify LEDs on all systems, and diagnostic LEDs on select IBM rack-mount servers)
- Serial-over-LAN (SOL)
- Service processor configuration
- Hardware control point discovery using Service Location Protocol (SLP): BladeCenter Advanced Management Module (AMM), IBM Power Systems Hardware Management Console (HMC), and Flexible Service Processor (FSP)
- Virtual partition creation

### 1.2.5 Additional features

Additional features include:

- Command line interface (CLI) mode, scripts, simple text-based configuration files
- Perl, used primarily for easy debugging and development
- xCAT installed Linux nodes are identical to kickstart (Red Hat) or autoyast (SUSE®) installed nodes. Nothing is altered from the standard distribution; stock kernels are used, thus maintaining support for commercial distributions.
- Works well with any manufacturer's x86 or x86-64 server
- IBM BladeCenter is fully supported
- Custom extensions (adding a new distribution) require minimal effort
- Portable Batch System (PBS), IBM General Parallel File System™ (GPFS), and Myrinet installations
- Post installation scripts for both diskful and diskless environments

## 1.3 Support offering terms

Consistent with the support tier purchased, IBM provides support only for the issues that arise on copies of xCAT 2 installed on the servers for which your IBM Support for xCAT support contract has been purchased. Support is limited to the

specified operating environment and is available only pursuant to the terms and conditions of the support agreement.

The annual renewable support offering is priced per server. Technical support is available world wide and can be purchased through Passport Advantage® and Passport Advantage Express.

Because the IBM Support for xCAT support offerings are for an open source software project, all fixes and code are provided through the following official xCAT SourceForge Web site. (IBM plans to deliver all fixes to the open source project.)

http://xcat.sourceforge.net/

The IBM Support for xCAT 2 is offered for xCAT Version 2.0 and beyond. This IBM support contract is not available for xCAT 1.3. Clients who have downloaded the resource monitoring and control (RMC) plug-in from IBM for monitoring capabilities for use with xCAT 2 and have purchased IBM Support for xCAT will be provided support for RMC pursuant to the terms and conditions of the customer's xCAT support contract.

For xCAT monitoring capabilities, the resource monitoring and control (RMC) plug-in from the IBM Reliable Scalable Cluster Technologies (RSCT) component can be download from the following Web page:

http://www14.software.ibm.com/webapp/set2/sas/f/cluster/home.html

Table 1-1 lists the two tiers of IBM Support for xCAT that are based on your particular requirements. Both enhanced and elite support tiers are delivered through remote support teams; on-site support is not provided.

*Table 1-1 Two support tiers*

| Selected Support Offerings | IBM Enhanced Support | IBM Elite Support |
|---|---|---|
| Electronic problem submission | Yes | Yes |
| Voice problem submission | Yes | Yes |
| Number of electronic of voice submitted problems | Unlimited | Unlimited |
| Support hours | 8 a.m. to 5 p.m. Monday - Friday | 8 a.m. to 5 p.m. Monday - Friday (24x7x365 for sev-1[a]) |
| Response target | Four business hours | Two business hours |
| Technical contacts | Two | Unlimited |

| Selected Support Offerings | IBM Enhanced Support | IBM Elite Support |
|---|---|---|
| Developer assistance incidents | Variable | Variable |
| Availability | Worldwide | Worldwide |

a. Severity 1 (sev-1) defect definition: Production system is down, critical business impact, unable to use the product in a production environment, no workaround is available.

Refer to the following Web page for information about IBM Select Support offerings:

http://www-306.ibm.com/software/lotus/passportadvantage/

# 2

# xCAT 2 architecture

This chapter describes the various xCAT 2.0 components and how they work together to manage and operate large HPC clusters. Additionally, we describe several node installation concepts that will be used in the chapters.

Topics covered in this chapter include:

- General cluster concepts
  - High performance computing (HPC) clusters
  - HPC cluster components
  - HPC cluster node types
  - HPC cluster network types
  - Hardware and power control, and console access
- xCAT 2.0 implementation
  - xCAT features
  - xCAT database
  - Network installation fundamentals
  - Monitoring infrastructure
  - Parallel remote command execution and file copy
  - Conclusion

## 2.1 General cluster concepts

According to dictionary definitions, a *cluster* is a a number of things of the same sort gathered together or growing together, a bunch. In computing, a cluster is a group of computing devices (computers) connected and managed together as an entity, and working closely to solve a specific IT requirement. Clusters can be:

- Load balancing
- High availability

From the management software perspective, clusters can be classified as peer domain clusters (all nodes in clusters are *peers*) and management domain clusters (one node assumes the role of *management* node).

A special flavor of load balancing clusters is the high performance computing (HPC) cluster.

### 2.1.1 High performance computing (HPC) clusters

HPC clusters are designed to use parallel computing to apply more processor power for the solution of a problem. Many examples exist of scientific computing using multiple low-cost processors in parallel to perform large numbers of operations.

#### Cluster management considerations:

An HPC cluster is typically made up of a large number of nodes. Clusters of hundreds of nodes are not uncommon. Creating an architecture for this kind of cluster brings its own challenges, which include how to accomplish the following challenges:

- Install and maintain the operating system and the application environment on all nodes
- Pro-actively manage these nodes that are issuing commands and gracefully handling failures
- Meet the requirement for parallel, concurrent, and high-performance access to the same file system
- Ensure inter-process communication between the nodes to coordinate the work that must be done in parallel

The goal is to provide the image of a single system by managing, operating, and coordinating a large number of discrete computers.

Often in this environment, a user interacts with a specific node to initiate or schedule a job to be run. The application, in conjunction with various functions within the cluster, then determines how this job is spread across the various nodes of the cluster to take advantage of the resources available to produce the desired result.

### Horizontal scaling

Horizontal scaling means adding duplicated servers to handle additional load. This applies to multitier scenarios and normally requires the use of a load balancer that sits in front of the Web server, so that the physical number of WebSphere® commerce servers and Web servers in your site are hidden from the Web client.

## 2.1.2 HPC cluster components

A cluster consists of nodes (computers) having different roles (further described in 2.1.2, “HPC cluster components” on page 13).

Nodes are connected through networks of different types, which also have different roles (further described in 2.1.4, “HPC cluster network types” on page 15).

Application data can be loaded onto the nodes by using shared storage. Shared storage can be added to the cluster in the form of Network File System (NFS), cluster file system (such as the IBM General Parallel File System (GPFS)), or directly attached (rarely used). To add GPFS for your xCAT diskless nodes, see 3.3, “Adding an existing GPFS to a diskless xCAT cluster” on page 63.

Cluster management software performs management, monitoring, security, storage, and other tasks for the cluster. Management software has several components, and xCAT uses a modular, pluggable approach for various software management components. The major management software components are:

- Cluster daemon, remote command execution, and file transfer
- Cluster database, including database tools and management commands
- Hardware control and remote console, described in 2.1.5, “Hardware and power control, and console access” on page 16.
- Monitoring, described in 2.2.4, “Monitoring infrastructure” on page 26.
- Operating system and services
- Scientific and technical specialized application software

## 2.1.3 HPC cluster node types

This section presents the types of nodes in HPC clusters. They are:

- User
- Control
- Compute node
- Management
- Storage
- Installation

**Note:** The node roles can be combined, for example, a control node may also be a storage node. Also the Management Node can play multiple roles.

### User

Individual nodes of a cluster are often on a private network that cannot be accessed directly from the outside or corporate network. Even if they are accessible, most cluster nodes are not necessarily configured to provide an optimal user interface. The user node is the one type of node that is configured to provide that interface for users (possibly on outside networks) who may gain access to the cluster to request that a job be run, or to review the results of a previously run job.

### Control

Control nodes provide services that help the other nodes in the cluster work together to obtain the desired result. Control nodes can provide two sets of functions:

- Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS), and other similar functions for the cluster. These functions enable the nodes to easily be added to the cluster and to ensure they can communicate with the other nodes.
- Scheduling what tasks are to be done by what compute nodes. For instance, if a compute node finishes one task and is available to do additional work, the control node might assign that node the next task requiring work.

### Compute node

The compute node is where the real computing is performed. The majority of the nodes in a cluster are typically compute nodes. To provide an overall solution, a compute node can execute one or more tasks, based on the scheduling system.

### Management

Clusters are complex environments, and the management of the individual components is very important. The management node provides many capabilities, including:

- Monitoring the status of individual nodes
- Issuing management commands to individual nodes to correct problems or to provide commands to perform management functions, such as power on and off. You should not underestimate the importance of cluster management. It is an imperative when trying to coordinate the activities of large numbers of systems.

### Storage

For some applications that are run in a cluster, compute nodes must have fast, reliable, and simultaneous access to the storage system. This can be accomplished in a variety of ways depending on the specific requirements of the application. Storage devices can be directly attached to the nodes or connected only to a centralized node that is responsible for hosting the storage requests.

### Installation

In most clusters, the compute nodes (and other nodes) might have to be reconfigured or reinstalled with a new image relatively often. The installation node provides the images and the mechanism for easily and quickly installing or reinstalling software on the cluster nodes.

## 2.1.4 HPC cluster network types

The main virtual local area network (VLAN) types and roles in an HPC cluster are:

- Management VLAN
- Cluster VLAN
- Public VLAN

### Management VLAN

The management VLAN connects all the management devices to the management node. Management network is used for controlling and monitoring nodes, as well as for other administrative tasks (parallel command execution, file transfers, and so on).

### Cluster VLAN

The cluster VLAN provides the internal network for all of the nodes in the cluster. All of the nodes in the cluster use the cluster VLAN for network booting and

network installation, so it is important that you use a network adapter that is capable of network booting, such as HPC and application traffic.

### Public VLAN

The public VLAN allows users to access the cluster through the management node or, optionally, the user nodes (also known as front-end nodes). The cluster administrators can also use the public VLAN for remote management.

**Note:** Generally, users do not access individual nodes in the cluster, rather access is through the user nodes. Moreover, user access to the cluster must be secured, as in general, the tightly coupled clusters (HPC or horizontal scaling are designed for inter-node performance communication rather than security).

## 2.1.5 Hardware and power control, and console access

This section briefly describes the cluster components involved in hardware control and remote console access.

### Hardware control and monitoring software

Hardware control and monitoring software provide remote hardware control and monitoring functions for cluster nodes and devices from a single point of control, which is typically referred to as the *management node* or server. The management node allows you to remotely control or monitor cluster nodes.

Hardware control and monitoring functions vary by platform and depend on specific hardware, software, network, and configuration requirements. The requirements for remote power are separate and distinct from the requirements for remote console.

### Hardware control points

Hardware control software on the management node must communicate with hardware control points to request node power status, reboot, and power on and off functions. A hardware control point is the specific piece of hardware through which the management server controls node hardware. Hardware control points are on the management VLAN, and connected to the hardware that ultimately controls the power functions.

The hardware control points are:

- Hardware Management Console (HMC): for HMC-attached System p nodes
- Integrated Virtualization Manager (IVM): for IVM-managed System p nodes

- Advanced System Manager/flexible service processor (ASM/FSP): for System p direct attach nodes
- Advanced Management Module (AMM): for BladeCenter nodes
- Baseboard management controller (BMC): for IBM xSeries® 336, 346, System x3455, x3550, and x3650 servers
- Remote Supervisor Adapter II (RSA II): for System x nodes

### *HMC control point*

The Hardware Management Console (HMC) controls managed systems, logical partitions, Capacity on Demand (CoD), and updates for an IBM System p server.

Creating and managing partitions requires an interface through which to communicate with the POWER™ Hypervisor. The HMC accomplishes this through the service processor, which routes the messages up to the Hypervisor.

The HMC provides a large array of management (control) and service applications (monitoring) functions. It additionally provide remote console support.

### *IVM control point*

The Integrated Virtualization Manager (IVM) is a component of the Virtual I/O Server, which is included with the Advanced Power Virtualization feature. With the use of IVM, customers can manage partitions on an IBM POWER5™ or POWER6 server (or servers) without an HMC.

As mentioned previously, to create or manage partitions, an interface is necessary for communicating with the POWER Hypervisor™ in an IVM environment. A virtual device called the Virtual Management Channel (VMC) was created to enable communication between the IVM and the Hypervisor.

IVM provides many of the same functions as the HMC, but is limited to basic entry-level partitioning and a single VIO server.

### *ASM/FSP control point*

For a System p server that is not managed by an HMC, you must connect the server to a terminal or PC and apply power. You can power the system on and off using the power button on the control panel (operator panel) or the Advanced System Management Interface (ASMI).

The Advanced System Management Interface (ASMI) is the interface to the flexible service processor (FSP) that is required to perform general and administrator-level service tasks, such as reading service processor error logs, reading vital product data, setting up the service processor, and controlling the system.

The hardware control and monitoring is not as extensive as the HMC and IVM, and it does not provide remote console support.

### *BladeCenter Management Module control point*

The BladeCenter Management Module is a hot-swappable hardware device plugged into the BladeCenter chassis management bay. The management module functions as a system-management processor (service processor), and keyboard, video, and mouse (KVM) multiplexor for blade servers.

The management module provides the following features:

- System-management processor functions for the BladeCenter system
- Ethernet connection to a management network
- Video port (local and remote console)
- IBM PS/2 keyboard and mouse ports
- 10/100 Mbps Ethernet connection

Functions provided by the management module include, but are not limited to, the following items:

- Chassis configuration
- Chassis cooling (blower control and temperature sensing)
- Power module control
- Blade initialization
- Switch module initialization
- Media selection and control (CD-ROM or floppy disk drive)
- Remote and local console control
- Customer interface panel
- Chassis-level power management
- Power on/off control
- Chassis thermal sensing (monitor thermal status and post alerts)
- Serial-over-LAN (SOL) session control and terminal server

**Note:** For details about the BladeCenter Management Module, see:

- The IBM Journal of Research and Development Web page:

  `http://www.research.ibm.com/journal/rd/496/desai.html`

- *IBM eServer xSeries and BladeCenter Server Management*, SG24-6495

### *BMC control point*

The baseboard management controller (BMC) is located on a blade and works in conjunction with the management module to manage the blade.

Functions provided by the blade BMC include, but are not limited to, the following items:

- Power on/off control
- Media control (request and enable or disable CD-ROM or floppy disk drive access)
- Keyboard and mouse control (access to USB bus on the midplane)
- Video control (access to video bus on the midplane)
- Thermal sensing (monitor thermal status and post alerts)
- Management module interface (communications with management module)
- Blade power management
- SOL session

**Note:** For details about the BMC, see:

- The IBM Journal of Research and Development Web page:

  http://www.research.ibm.com/journal/rd/496/desai.html

- *IBM eServer xSeries and BladeCenter Server Management*, SG24-6495

### *RSA II control point*

Remote Supervisor Adapter II (RSA II) is similar to the BMC, but it is not integrated. The most useful functions and features of the RSA II are:

- Automatic notification and alerts
- Continuous health monitoring and control
- Event log
- LAN and Advanced Systems Management (ASM) interconnect remote access
- Operating system failure window capture
- Remote media
- Remote power control
- Server console redirection

**Note:** For details about RSA II, see:

- The IBM Remote Supervisor Adapter II Slimline and RSA II User's Guide - System x Web page:

  http://www-304.ibm.com/systems/support/supportsite.wss/docdisplay?brandind=5000008&lndocid=MIGR-57091

- *IBM eServer xSeries and BladeCenter Server Management*, SG24-6495

## Console access device (terminal server)

A management node typically communicates with console server hardware to open a console window for a node. Console servers should be on the

management VLAN, which connects the management node to the cluster hardware, and connected to node serial ports. This out-of-band network configuration allows a remote console to be opened from the management node even if the cluster VLAN is inaccessible. For example, if the cluster VLAN is offline, remote console can still access the target node to open a console window.

A standalone terminal server is not required for servers where remote console support or serial-over-LAN (SOL) support is available.

Review the following notes about remote console support:

- For HMC-attached System p, the HMC is the remote console server.
- For non HMC-attached System p, where an independent device does not exist that can serve as a remote console server, console traffic can be managed by the FSP.
- For BladeCenter, blade servers support remote console through the Ethernet Switch Module, using SOL. Refer to your BladeCenter documentation for information about enabling and configuring SOL.
- For standalone System x servers, use any of the following console servers for remote console access:
  - MRV IR-8020, IR-8040, LX-4008S, LX-4016S, and LX-4032S
  - Avocent CPS1600
  - Cyclades AlterPath ACS48

The BladeCenter SOL solution preserves the simplicity of serial-connected LAN management of servers within the cabling constraints of dense, rack-mounted blade servers. Logical serial connectivity to LAN-based terminal applications is preserved without dedicated serial cabling. Serial data can be transparently forwarded to remote terminal applications through the existing Ethernet network by routing that data over the internal Ethernet fabric within the chassis between the server local service processor or BMC and chassis management modules. In the chassis, basic terminal server functions are implemented in the management module through an imbedded SOL Telnet server.

**Note:** For information about SOL and general BladeCenter networking, see:

`http://www.research.ibm.com/journal/rd/496/hunter.html`

## 2.2 xCAT 2.0 implementation

xCAT is a scalable distributed computing management and provisioning tool that provides a unified interface for hardware control, discovery, and OS diskful/diskless deployment.

As mentioned previously, xCAT has been deploying and managing large Linux systems since 1999. Its powerful customizable architecture has been in use long before other management solutions came into existence. xCAT 2 is a new project created by IBM cluster management developers. It is not an upgrade from xCAT 1.x version.

The overall xCAT architecture is shown in Figure 2-1. For a more comprehensive cluster diagram, see also Figure 2-9 on page 32.

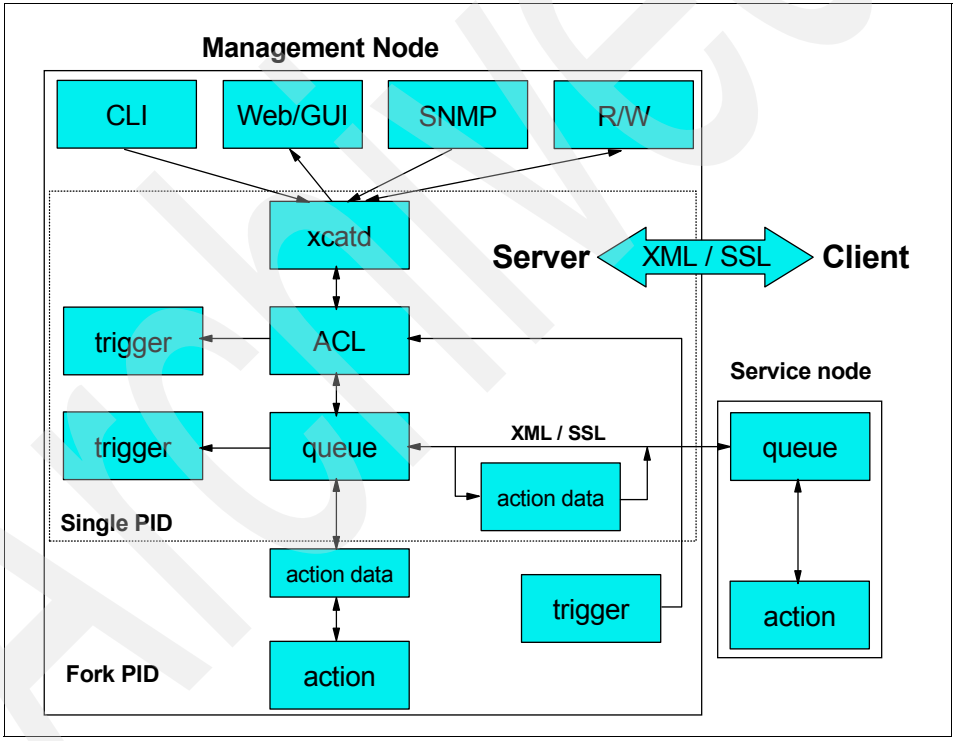


*Figure 2-1 xCAT 2 architecture*

In the xCAT client/server application, data flow between client and server is controlled by the xCAT daemon (xcatd) on the management node, shown in

Figure 2-1. Although the flow has not yet been entirely implement, the basic order of data flow is as follows:

1. When user invokes an **xcat** command on the client, the command can either be a symbolic link to xcatclient/xcatclientnnr or a thin wrapper that calls:

   `xCAT::Client::submit_request()`

2. The **xcatclient** command packages the data into XML and passes it to xcatd.
3. When xcatd receives the request, it forks to process the request.
4. The ACL (role policy engine) determines whether this person is allowed to execute this request by evaluating the following information:
   - The command name and arguments
   - Who executed the command on the client machine
   - The host name and IP address of the client machine
   - The node range passed to the command
5. If the ACL check is approved, the command is passed to the queue:

   The queue can run the action in either of the following two modes. The client command wrapper decides which mode to use (although it can give the user a flag to specify):

   - For the life of the action, keep the socket connection with the client open and continue to send back the output of the action as it is produced.
   - Initiate the action, pass the action ID back to the client, and close the connection. At any subsequent time, the client can use the action ID to request the status and output of the action. This action is intended for commands that are long-running.

   The queue logs every action performed, including date and time, command name, arguments, who, and so on.

   In phase two, the 2ueue will support locking (semaphores) to serialize actions that should not be run simultaneously.
6. To invoke the action, the data (XML) is passed to the appropriate plug-in Perl module, which performs the action and returns results to the client.

For details about xCAT architecture, see the xCAT wiki:

http://xcat.wiki.sourceforge.net/

## 2.2.1 xCAT features

This section lists the main features of xCAT architecture.

- Client/server architecture

  Clients can run on any Perl-compliant system (including Windows). All communications are SSL encrypted.

- Role-based administration

  Different users can be assigned various administrative roles for different resources.

- New stateless and iSCSI nodes support

  Stateless nodes can be RAM-root, compressed RAM-root, or stacked NFS-root. Linux software initiator iSCSI support for Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES) is included. Systems without hardware-based initiators can also be installed and booted using iSCSI.

- Scalability

  xCAT 2.x was designed to scale to 100,000 and more nodes with xCAT's Hierarchical Management Cloud. A single management node may have any number of stateless service nodes to increase the provisioning throughput and management of the largest clusters. All cluster services such as LDAP, DNS, DHCP, NTP, Syslog, and so on, are configured to use the Hierarchical Management Cloud. Outbound cluster management commands (for example, **rpower**, **xdsh**, **xdcp**, and so on) utilize this hierarchy for scalable systems management. See Figure 2-2.

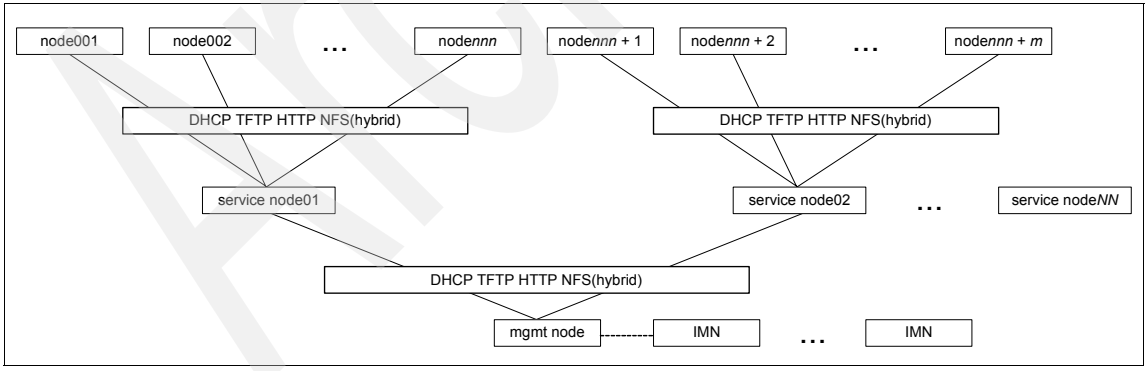


*Figure 2-2 Scalable hierarchal management cloud*

- Automagic discovery

  Single power button press physical location based discovery and configuration capability. Although this feature is mostly hardware-dependent, xCAT 2 has been developed to ease integration for new hardware. A plug-in software architecture provides easy development of new features. You can extend xCAT by adding your own functionality.

- Plug-in architecture for compartmental development

  Add your own xCAT functionally to do what ever you want. New plug-ins extend the xCAT vocabulary available to xCAT clients.

- Monitoring plug-in infrastructure

  This allows you to easily integrate third-party monitoring software into the xCAT cluster.

- Notification infrastructure

  This allows you to be able to watch for xCAT DB table changes.

- SNMP monitoring

  Default monitoring uses SNP trap handlers to handle all SNMP traps.

- Flexible monitoring infrastructure.

  You can easily integrate vendor monitoring software into the xCAT cluster. Currently, the following plug-ins are provided with xCAT: SNMP, RMC (RSCT), Ganglia, and Performance Copilot. In addition, the notification infrastructure is able to watch for xCAT DB table changes.

- Centralized console and systems logs

  xCAT provides console access to managed nodes and centralized logging. Documentation available includes cookbooks, how-to information, complete man pages, and database table documentation

### 2.2.2 xCAT database

To access the database, xCAT uses the Perl database interface (DBI) so that any database can be used to store the xCAT tables. SQLite is the default database.

The **`tabedit`** command is provided to simulate the 1.x table format.

To access the tables, all xCAT code should use Table.pm, which implements the following features (refer to Figure 2-3):

- Notifications for table changes (triggers): A separate table lists the table name (or lists *****) and a command that is run when that table is changed. When the command is run, the changed rows are piped into its standard input (stdin).

- A begin and end mechanism: The xCAT code can use the mechanism when it updates many rows. This allows Table.pm to optimize the update to the database and call the notifications only once for all the updates.
- Ability to support other non-Perl programs: These programs read the database using packages like ODBC (for C program access).

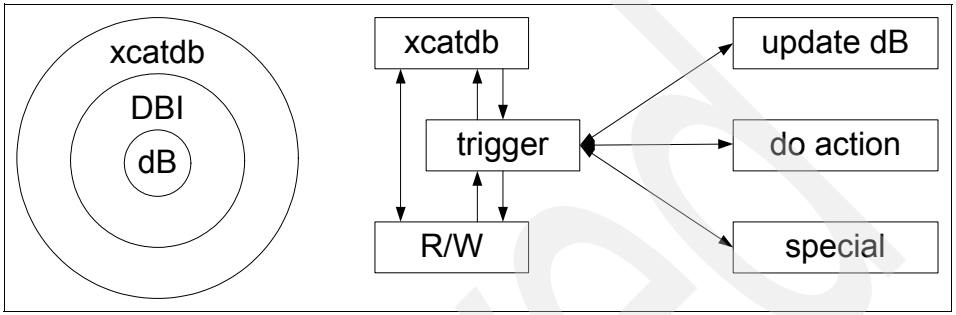


*Figure 2-3 Database interaction*

## 2.2.3 Network installation fundamentals

One fundamental aspect of operating an HPC cluster is the ability to efficiently boot, install nodes, and manage software (OS and application). xCAT has the ability to control the node software from boot, to installation, to software maintenance. Figure 2-4 presents a diagram of the node boot and installation process.

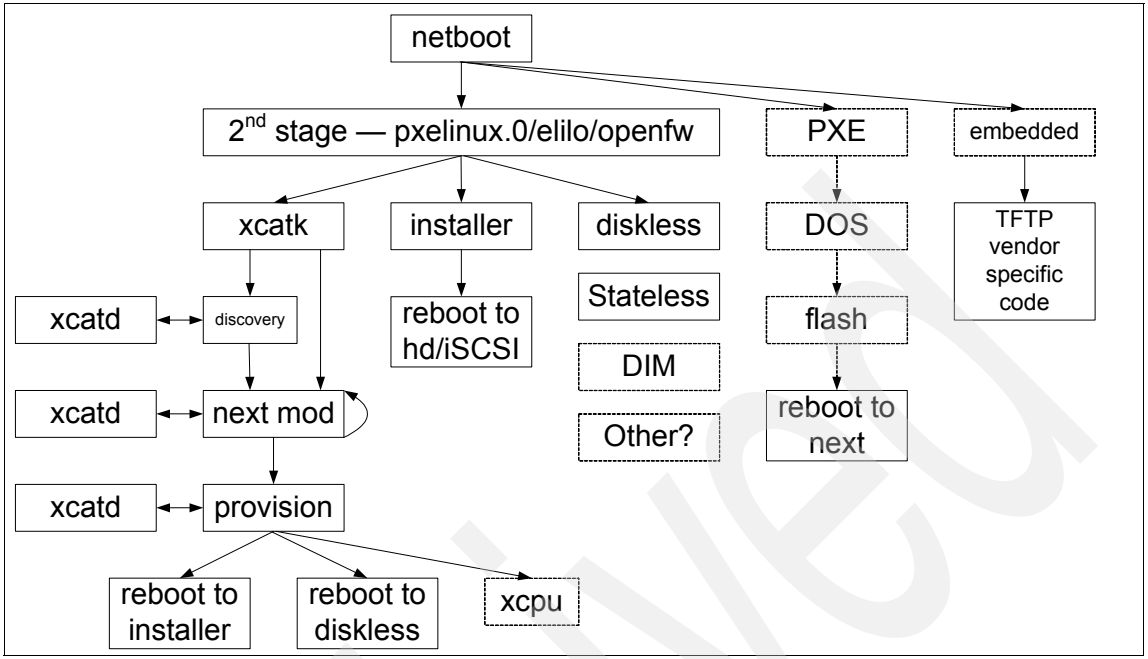


*Figure 2-4 Node netboot process*

### Preboot eXecution Environment

The Preboot eXecution Environment (PXE[1]), is an environment to boot computers using a network interface independently of available data storage devices or installed operating systems.

### Other platforms boot

The IBM Power Systems architecture in named Common Hardware Reference Platform (CHRP). CHRP was published jointly by IBM and Apple in 1995. The CHRP boot process is different from PXE. You can find more details about the CHRP boot process and bootinfo config variables at:

http://playground.sun.com/1275/bindings/chrp/chrp1_7a.ps

## 2.2.4 Monitoring infrastructure

Two monitoring infrastructures are available in xCAT 2:

- xCAT monitoring plug-in infrastructure: This allows you to plug in one or more vendor monitoring software such as Ganglia, RMC, SNMP and others to

[1] PXE is mainly used in platforms based on Intel.

monitor the xCAT cluster. This section describes the xCAT monitoring plug-in infrastructure.

- xCAT notification infrastructure: This allows you to watch for the changes in xCAT database tables.

For details, select the xCAT2-Monitoring.pdf file from the following Web page:

http://xcat.svn.sourceforge.net/svnroot/xcat/xcat-core/trunk/xCAT-client/share/doc/

## xCAT monitoring plug-in infrastructure

With xCAT 2.0, you can integrate vendor monitoring software into your xCAT cluster. The idea is to use monitoring plug-in modules that act as bridges to connect xCAT and the vendor software. Although you may write your own monitoring plug-in modules, over the time, xCAT will supply a list of built-in plug-in modules for the most common monitoring software, such as:

- xCAT (xcatmon.pm): This provides monitoring node status using **`fping`**.
- SNMP (snmpmon.pm): This is the SNMP monitoring.
- RMC (rmcmon.pm): This provides resource monitoring and control through IBM RSCT (Reliable Scalable Cluster Technology).
- Ganglia (gangliamon.pm): This is an open source monitoring suite for HPC environments.
- Nagios (nagiosmon.pm): This is another open source host, service and network monitoring program.
- Performance Co-pilot (pcpmon.pm): This is a family of products from SGI for system-level performance monitoring and management services.

You may choose one or more monitoring plug-ins to monitor the xCAT cluster.

## xCAT monitoring commands overview

Seven commands are available for monitoring purposes. They are listed in Table 2-1 on page 27.

*Table 2-1 xCAT monitoring commands*

| Command | Description |
|---|---|
| monls | Lists the current or all the monitoring plug-in names, their status, and description. |
| monadd | Adds a monitoring plug-in to the monitoring table. This also adds the configuration scripts for the monitoring plug-in, if any, to the postscripts table. |

| Command | Description |
|---|---|
| monrm | Removes a monitoring plug-in from the monitoring table. It also removes the configuration scripts for the monitoring plug-in from the postscripts table. |
| moncfg | Configures the vendor monitoring software on the management server and the service node for the given nodes to include the nodes into the monitoring domain. It does all the necessary configuration changes to prepare the software for monitoring the nodes. The `-r` option configures the nodes too. |
| mondecfg | Deconfigures the vendor monitoring software on the management server and the service node for the given nodes to remove the nodes from the monitoring domain. The `-r` option deconfigures the nodes too. |
| monstart | Starts vendor software on the management server and the service node for the given nodes to monitor the xCAT cluster. It includes starting the daemons. The `-r` option starts the daemons on the nodes too. |
| monstop | Stops vendor software on the management server and the service node for the given nodes from monitoring the xCAT cluster. The `-r` stops the daemons on the nodes too. |

The two ways to deploy third-party software to monitor the xCAT cluster are:

- Configure the software on the nodes during the node deployment phase.
- Configure the software after the node is up and running.

## xCAT monitoring options

The three xCAT monitoring options are:

- Start and stop monitoring
- Automated node update
- Node status monitoring

### *Start and stop monitoring (see Figure 2-5 on page 29)*

The monitorctrl module is the control center. The **`xcatd`** command invokes monitorctrl, which gets node hierarchy from the xCAT database, then invokes plug-in modules with the node hierarchy information. The plug-in module invokes and initializes third-party monitoring software, compares the node hierarchy information with the current nodes in the third-party monitoring domain, and makes sure they are in sync.

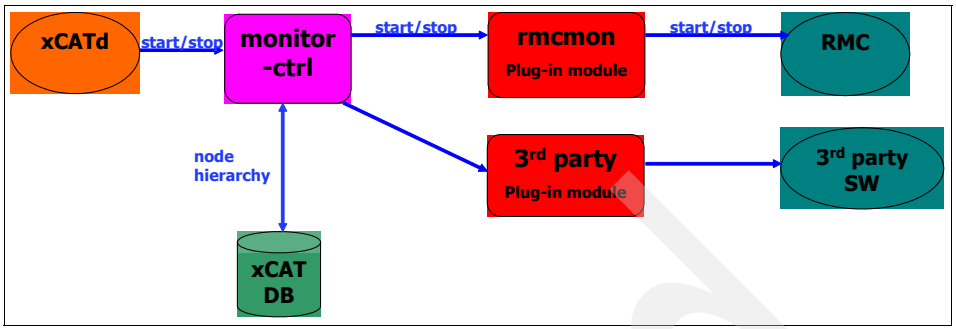


*Figure 2-5 xCAT start and stop monitoring*

### *Automated node update (see Figure 2-6)*

xCAT notification infrastructure notifies monitorctrl nodes being added or removed from the xCAT cluster, and monitorctrl informs all the plug-in modules. The plug-in module calls relevant functions in third-party software to add or remove the nodes in the monitoring domain.

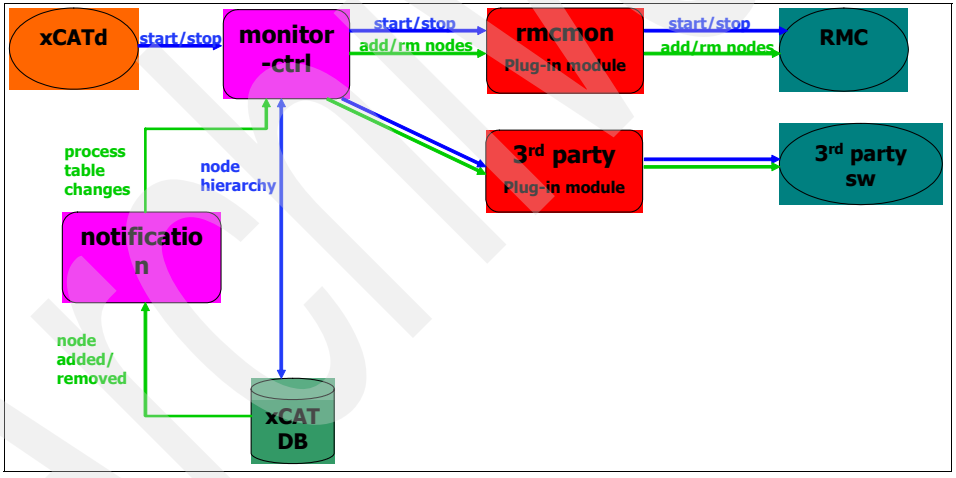


*Figure 2-6 xCAT automated node update*

### *Node status monitoring (see Figure 2-7 on page 30)*

xCAT maintains current node status for each node. Possible node status values are defined, installing, booting, active, off, and others. Nodes status is stored in the status column in the nodelist table. Optionally, a third-party software can be chosen to help by providing the node liveliness status if it has the capability. The monitorctrl invokes the third-party node status monitoring through its plug-in module. Also, third-party software can update the xCAT database with the node

status changes through its plug-in module or directly access the database by using **chtab** command.

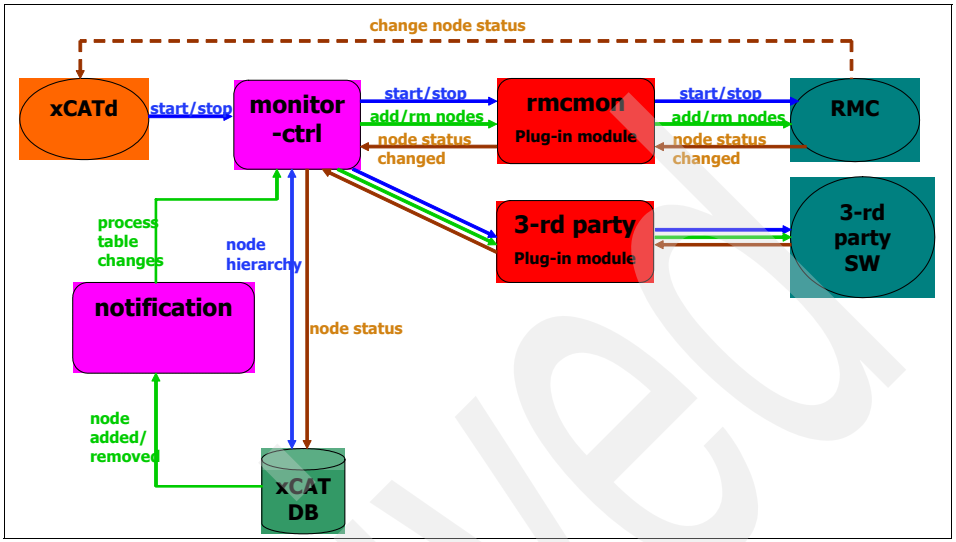


*Figure 2-7 xCAT node status monitoring*

## 2.2.5 Parallel remote command execution and file copy

One of the most important aspects of the cluster management is the parallel remote command execution and file copy from Management Node to managed nodes. This is required for node configuration changes and other administrative operations. This operation must be secured using available OS tools and libraries (for example, OpenSSL, OpenSSH). Figure 2-8 on page 31 the parallel remote command and file copy infrastructure in an xCAT cluster.

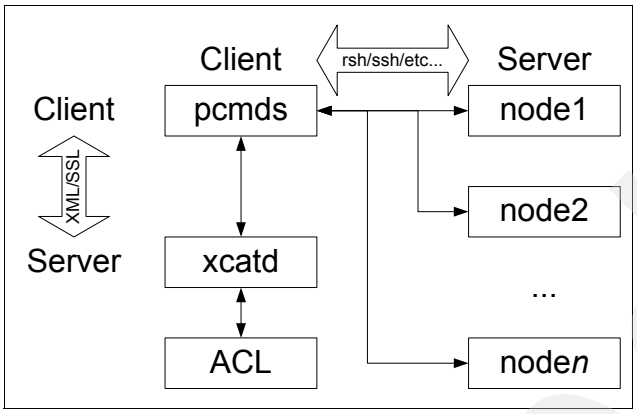


*Figure 2-8 Parallel remote command execution*

## 2.2.6 Conclusion

The Extreme Cluster Administration Toolkit (xCAT) is an open source Linux, AIX, and Windows scale-out cluster management solution design that is based on community requirements (mostly HPC) and the following principles:

- Build upon the work of others
- Leverage best practices
- Scripts only (no compiled code)
- Portability
- Modular - can be easily integrated and enhanced

A typical xCAT cluster component architecture is shown in Figure 2-9 on page 32. In the figure, the following abbreviations are indicated:

- MN is management node.
- SN is service node.
- CN is compute node.

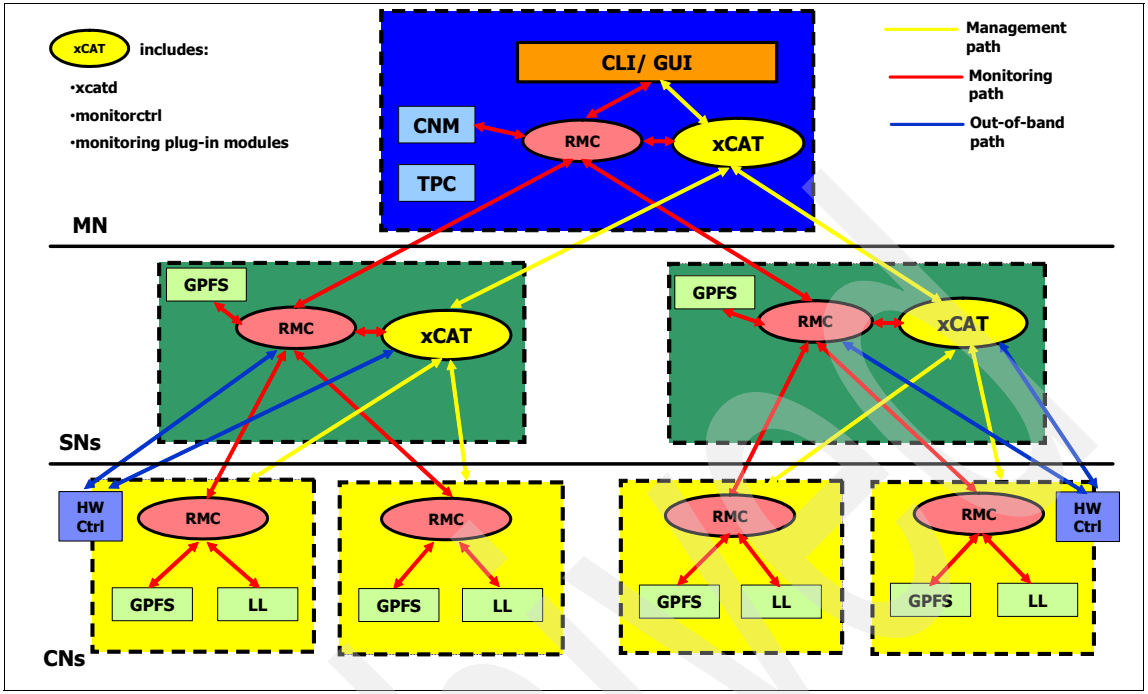


*Figure 2-9 Component architecture of xCAT (the big picture)*

# 3

# CSM to xCAT 2 transition

This chapter provides instructions to migrate from IBM Cluster Systems Management (CSM) to xCAT version 2. (We refer to xCAT 2 as simply xCAT in the remainder of this paper.)

**Note:** Currently, IBM suggests that you implement xCAT with new clusters (hardware), and maintain your current CSM configuration when possible. xCAT has been primarily designed for HPC environments, thus if you have CSM in a general computing environment, such as a commercial installation, you should probably *not* take the transition path to xCAT. Transition is a disruptive process and we recommend it only if you have specific reasons.

The only supported method for migrating your nodes from CSM to xCAT is to use two different manager servers (nodes): one physical machine for your xCAT management node, and another machine used as the CSM management server.

However, we have also tested a scenario where the CSM management server is the same as the xCAT management node.

The following scenarios are presented in this chapter:

- CSM to xCAT on IBM System x and BladeCenter
- CSM to xCAT for HMC-controlled System p nodes
- Adding an existing GPFS to a diskless xCAT cluster
- CSM disk-based transition to xCAT diskful nodes

# 3.1 CSM to xCAT on IBM System x and BladeCenter

This section describes the transition from a CSM cluster to xCAT using the same physical server. Although using the same physical server is not recommended nor supported by IBM, you may use this environment if not enough hardware is available.

To demonstrate the transition steps, we used the following test platform:

- IBM BladeCenter E chassis (M/T 8677)
- IBM BladeCenter HS20 (M/T 8832) as management server and nodes
- BladeCenter Advanced Management Module, FC 1604, which is required for xCAT support
- BladeCenter network switch (we used Cisco switch)

Figure 3-1 presents the BladeCenter configuration used in our testing.

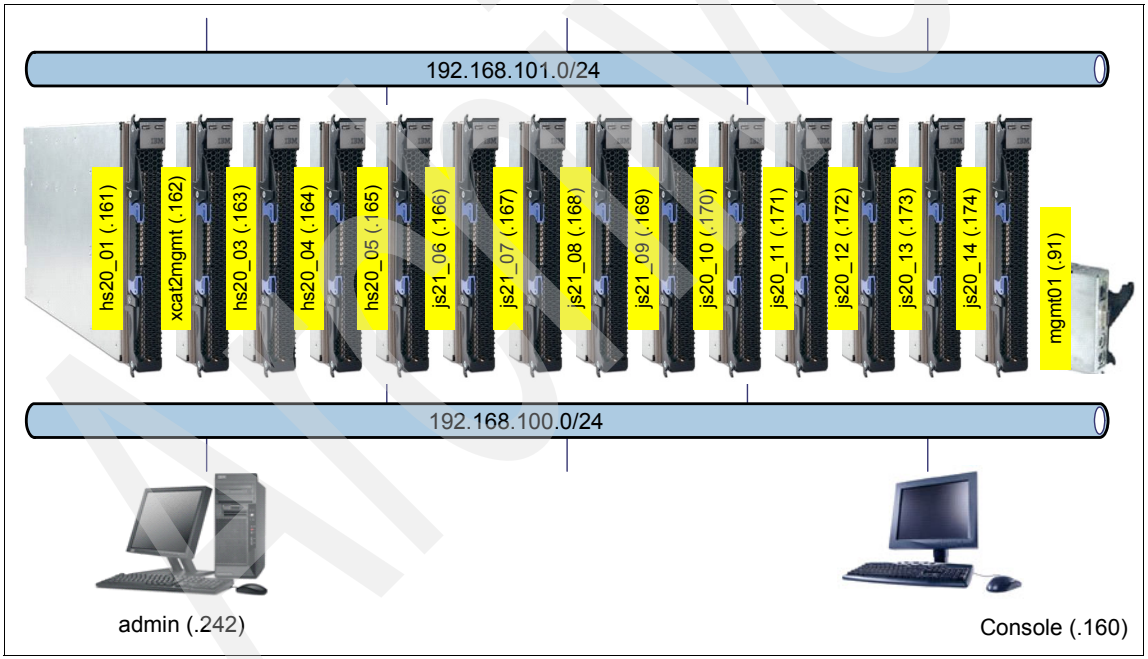


*Figure 3-1 ITSO test environment*

### 3.1.1 Preparing the platform

We assume that your CSM management server is configured to manage your CSM cluster.

Ensure that you have the operating system (we used Red Hat Enterprise Linux Server 5.1) ISO images for the corresponding architecture (x86, 32-bit and 64-bit) ready and your hardware has been updated to the latest firmware.

Download firmware updates from:

http://www.ibm.com/systems/support

Back up your CSM management server configuration using the **csmbackup** command.

If you also want to transfer your GPFS configuration to xCAT - do not forget to install GPFS dependencies. See the GPFS documentation page at:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.infocenter.doc/library.html

### 3.1.2 Installing xCAT packages

Download the latest archive xCAT packages from:

http://xcat.sourceforge.net/yum/download.html

Create a yum[1] repository and install the xCAT packages (RPM format), as shown in Example 3-1:

*Example 3-1 Adding the xCAT yum repository*

```
# mkdir /root/xcat-repo
tar jxf ~/core-rpms-snap.tar.bz2
tar jxf ~/dep-rpms-snap.tar.bz2
cd core-snap
./mklocalrepo.sh
cd ../dep-snap/rh5/x86
./mklocalrepo.sh
yum clean metadata
yum install xCAT.i386
```

Ensure that conserver and tftpd packages have been updated automatically through yum. Otherwise, perform this manually. See the following important note.

[1] yum (Yellow dog Updater, Modified)

**Important:** The default RHEL Server 5.1 tftp-server package conflicts with the atftp package, so you must remove it as follows:

```
rpm -Uvh ~/xcat-repo/dep-snap/rh5/x86/conserver-8.1.16-5.i386.rpm
rpm -e tftp-server
rpm -Uvh ~/xcat-repo/dep-snap/rh5/x86/atftp-0.7-4.i386.rpm
```

In the PATH variable, make sure that the xCAT commands directory is listed first, before the CSM binary directory. The profile we used is shown in Example 3-2.

*Example 3-2 Modified system profile*

```
# cat /etc/profile.d/xcat.sh
XCATROOT=/opt/xcat
PATH=$XCATROOT/bin:$XCATROOT/sbin:$PATH
MANPATH=$XCATROOT/share/man:$MANPATH
export XCATROOT PATH MANPATH
export PERL_BADLANG=0
```

### 3.1.3 Converting CSM database to xCAT database

Currently, a one-to-one relationship does not exist between the CSM database and the xCAT tables. At the time of this writing, the following CSM features are *not* supported in xCAT 2:

- Dynamic groups are not supported.
- xCAT uses MAC addresses instead of UUID when it configures the DHCP daemon.

We have developed a tool named csm2xcatdb to simplify transition from the CSM database to the xCAT tables. You may run this utility on the CSM management server. When invoked without any parameters, this utility dumps data into the xcatdb directory by default. Download the tool from the IBM Redbooks Web site:

`ftp://www.redbooks.ibm.com/redbooks/REDP4437/xCAT_ITSO_tools.tar`

The tool, which was created based on our experience, is not an automated tool so you must review the results. After review, the cluster administrator should also carefully review both the utility conversion log file (`conversion.log`) and the data dumped in the CSV files.

The final step of conversion is done using the **`tabrestore`** command to populate the xCAT database.

**Attention:** Passwords stored in the CSM database (hardware control points passwords such as HMC, BladeCenter Advanced Management Module, and so on) will not be extracted, because they are encrypted with a one-way-function. Our utility creates default hardware control point access passwords that you can later change to your actual passwords. Refer to your hardware documentation.

The xCAT command **tabrestore** does not preserve any data already in the xCAT tables; data is overwritten. If you do not want your data to be lost, use the **tabedit** command and manually insert data.

### 3.1.4 Preparing network boot support daemons

On the xCAT management node, configure the following required network services:

- Configure DHCP as follows:

```
service dhcpd stop
mv /etc/dhcpd.conf /etc/CSM.dhcpd.conf
```

- Configure TFTP as follows:

```
touch /var/log/atftpd.log
chown -R tftpd.tftpd /tftpboot /var/log/atftpd.log
```

The standard tftp server user and group is ftpd (see the /etc/passwd file).

By default, atftp starts as user *nobody*, which is not desired for our configuration. We changed this by editing the following lines in the atftp daemon startup script /etc/init.d/tftpd:

```
$START_DAEMON atftpd --user tftpd.tftpd --logfile
/var/log/atftpd.log --daemon && $LOG_SUCCESS || $LOG_FAILURE
```

**Note:** If you plan to update BladeCenter Advanced Management Module (AMM) firmware using Telnet and TFTP server, then you should use the default TFTP server supplied by Red Hat. We have learned that ATFTP cannot handle the AMM firmware image correctly.

### 3.1.5 Configuring service processors

To configure your Advanced Management Module (AMM) in BladeCenter chassis from your xCAT management node, use the commands shown in Example 3-3 on page 38.
In the example, `mm` indicates the AMM node name in the nodelist table.

*Example 3-3 Configuring BladeCenter AMM*

```
# rspconfig mm snmpcfg=enable sshcfg=enable
bca01: SNMP enable: OK
bca01: SSH enable: OK
# rspconfig mm pd1=redwoperf pd2=redwoperf
bca01: pd2: redwoperf
bca01: pd1: redwoperf
# rpower mm reset
bca01: reset
```

The mac table stores the hardware Ethernet address, MAC, for the nodes. To populate the mac table, run the **getmacs** command for your nodes, as follows:

```
getmacs bladenodes
makedhcp
```

Check that /etc/dhcpd.conf and /var/lib/dhcpd/dhcpd.leases have valid data and then start the DHCP daemon:

```
service dhcpd start
```

### 3.1.6 Preparing diskless image

The xCAT compute nodes can be classified as *diskful* and *diskless*, depending on how they load the operating system:

- Diskful nodes each have an individual disk (internal or accessible in a storage area network (SAN)), which is used for loading the operating system and saving temporary files (for example, logs, status, and application).
- Diskless nodes are further classified as stateful and stateless:
  - Stateful nodes: Although we have implemented diskless nodes using iSCSI, this does *not* require nor use hardware or firmware iSCSI. The firmware iSCSI initiators generally can work with the iscsi profile, but configuring them is not a requirement. The hardware iSCSI initiators should be used in conjunction with a non-iscsi profile.
  - Stateless nodes: Can be even further classified as:
    - Plain RAM file system (RAMFS), where the node RAM used is equal to the size of file system image.
    - Squashfs with RAMFS overlay (compressed RAMFS) is still all-memory hosted, but the nominal memory consumption is on the order of a tar.gz of the file system image, rather than the uncompressed size. Combining these two filesystems requires aufs (Another Union File System).

- NFS hybrid replaces the underlying squashfs with a read-only nfs mount, to further reduce memory consumption, but puts greater load on the NFS server.

In our example, we use a stateless compressed RAMFS node configuration. We start by creating a repository for xCAT using the following command:

```
# copycds /path_to_distro/distro.iso
```

Next, we generate the image for stateless booting, condition the nodes for netboot, and boot the nodes, as shown in Example 3-4.

*Example 3-4 Creating the boot image and booting nodes (x86 platform)*

```
# /opt/xcat/share/xcat/netboot/rh/genimage -i eth1 -n tg3 -o rhels5.1
-p compute -k 2.6.18-53.el5
# packimage -o rhels5 -p compute -a x86
# nodeset noderange netboot
# rpower noderange boot
```

## 3.2 CSM to xCAT for HMC-controlled System p nodes

As previously mentioned, the only supported method for migrating your nodes from CSM to xCAT is to use a different physical machine for your xCAT management node than the machine used as the CSM management server. However, in a homogeneous IBM System p cluster, we have been able to use the same node. If you want to use the same node, we strongly recommend backing up both the CSM database (**csmbackup** command) and the RSCT database (**/usr/sbin/rsct/bin/ctbackup** command) before you start.

If you decide to use a different machine from your xCAT management node, we recommend that you do not define your old CSM management server as a managed node in the new xCAT cluster, at least until you feel confident enough with xCAT and you consider that reverting to CSM is unlikely. If your CSM management server is a separate, standalone machine you probably will not want to bring it into the xCAT cluster.

### Our test environment

In our test environment, we used a System p6 520 with a dual-core POWER6 processor running at 4.2 GHz. We used logical partitions (LPARs) with virtualized LAN and SCSI resources. Virtualization software is PowerVM™ Virtual I/O Server version 1.5.2.1-FP-11.1.

This scenario was developed on an HMC-controlled Power 6 cluster and used Red Hat Enterprise Linux Server 5.1. Our test environment is configured as shown in Figure 3-2.

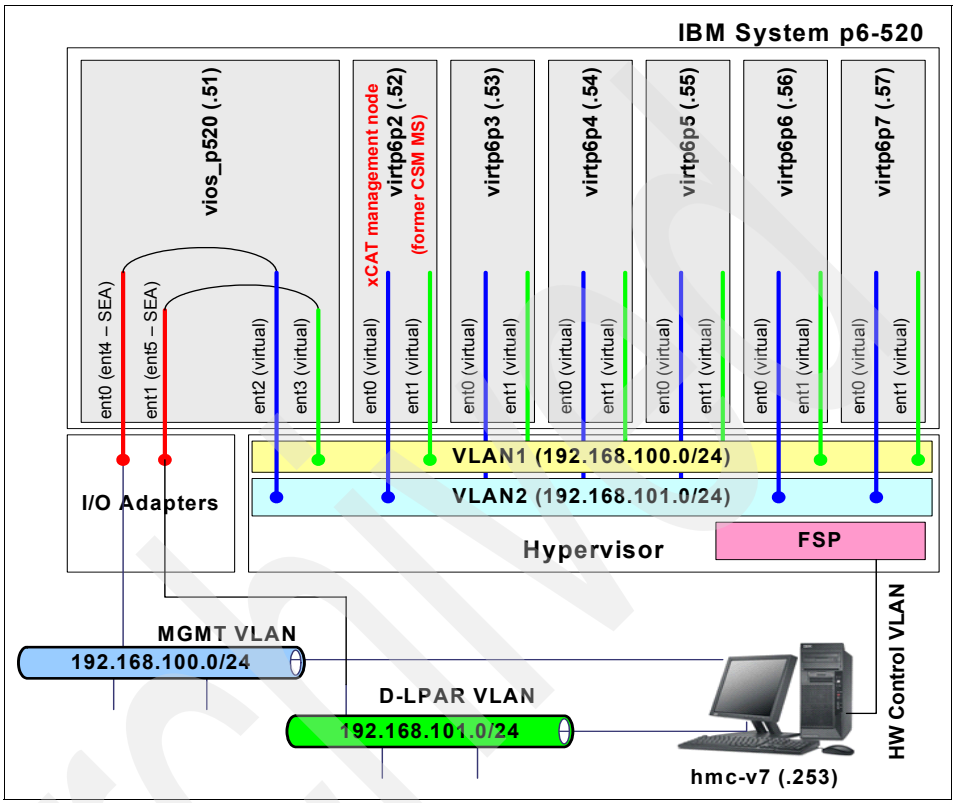


*Figure 3-2 ITSO test cluster (System p6)*

## 3.2.1 Installing xCAT code and backing up CSM database

To install xCAT and back up CSM:

1. Download and extract the xCAT 2 tarballs.
   See 4.1.1, “Downloading and extracting the xCAT 2 tarballs” on page 74.
2. Install xCAT 2 on the management node.
   See 4.1.2, “Installing xCAT 2 on the management node” on page 75.
3. Back up the original database tables.
   See 4.1.3, “Backing up the original database tables” on page 77.
4. Set the xCAT environment variables.
   See 4.1.4, “Setting the xCAT environment variables” on page 78.

## 3.2.2 Setting xCAT environment variables

In the PATH variable, make sure that the xCAT commands directory is listed first, before the CSM binary directory. The profile we used is shown in Example 3-5.

*Example 3-5 Modified system profile*

```
# cat /etc/profile.d/xcat.sh
XCATROOT=/opt/xcat
PATH=$XCATROOT/bin:$XCATROOT/sbin:$PATH
MANPATH=$XCATROOT/share/man:$MANPATH
export XCATROOT PATH MANPATH
export PERL_BADLANG=0
```

## 3.2.3 Migrating the node definitions from CSM

In this scenario (CSM to xCAT transition for HMC-controlled System p notes), we present two methods of initializing the xCAT database:

- Using the **rscan** command (utility) to scan the hardware for the initial data
- Using a conversion tool to migrate from the CSM database

You also have the choice to enter all the data manually. Other scenarios presented in this paper use the manual method. We suggest reading through this section first before you decide, and backing up the database frequently as you work through, in case you change your mind.

The conversion tool used is /opt/xcat/share/xcat/tools/csm2xcatdefs. At the time you decide to use this tool, it might have been enhanced. However, at the time of this writing, **rscan** filled fields with more information than the conversion tool did. The tool also only worked for CSM on POWER.

### Using the rscan utility

The node names that **rscan** obtains from the HMCs (LPAR names) might differ from the host names of the LPARs and the node names assigned in CSM, so you might have to add the new names to /etc/hosts. If you choose to use the **rscan** utility, skip the remainder of this section and start with 3.2.4, “Defining the Hardware Management Console (HMC)” on page 43.

### Using the conversion tool

Save your CSM node definitions into a flat ASCII file:

```
# lsnode -l > csm_defs
# csm2xcatdefs -z csm_defs > xcat_defs
```

This creates an xCAT stanza file with stanzas similar to the ones shown in Example 3-6.

*Example 3-6 xCAT stanza file created from CSM definitions*

```
virtp6p4:
        objtype=node
        xcatmaster=192.168.100.52
        serial=10E4B61
        arch=ppc64
        mtm=8203-E4A
        cons=hmc
        os=Linux
        power=hmc
        hcp=192.168.100.253
        mac=2E:D2:52:DA:B9:02
```

**Note:** At the time of this writing, the line `os=Linux` in the xCAT stanza file output was incorrect. The `os` variable must be set to a more specific value, such as `rhels5.1`. An easy way to fix it is just to *grep* it out (and fill it in later) or replace it with the correct value by using an editor such as sed. For example:

```
# csm2xcatdefs -z csm_defs | sed s/os=Linux/os=rhels5.1/ > xcat_defs
```

Now you can transfer the xCAT stanza file to another machine (if necessary) and create the initial definitions using the **mkdef** command. However, at the time of this writing, we first populated the database using the **nodeadd** command:

```
#nodeadd list_of_nodes groups=all,osi,lpars mgt=hmc
```

The `list_of_nodes` is a list of node names. It can be a range or a comma-separated list. The groups definitions let you perform operations on multiple targets at once. The groups shown are standard for LPARS:

```
# nodeadd virtp6p2-virtp6p7 groups=all,osi,lpar mgt=hmc
nodetype.nodetype=lpar,osi
```

This command creates the node definitions shown in Example 3-7.

*Example 3-7 Nodes listed using the nodels command*

```
# nodels
virtp6p2
virtp6p3
virtp6p4
virtp6p5
virtp6p6
virtp6p7
```

Now you can add the data from the CSM conversion tool:

```
# cat xcat_defs | mkdef -z
Object definitions have been created or modified.
```

To view all the data added for a particular node, use the **lsdef** command, as shown in Example 3-8.

*Example 3-8 Node definition in xCAT*

```
# lsdef virtp6p2
Object name: virtp6p2
        arch=ppc64
        cons=hmc
       groups=all,osi,lpar
        hcp=192.168.100.253
        mac=2E:D2:58:BF:B7:02
        mgt=hmc
        mtm=8203-E4A
        nodetype=lpar,osi
        os=Linux
        power=hmc
        serial=10E4B61
        xcatmaster=192.168.100.52
```

**Note:** This is just a starting point to populating the tables. Even if a later version of this conversion tool fills in additional fields, you will still need to verify everything that is needed is present and has correct values.

## 3.2.4 Defining the Hardware Management Console (HMC)

You may define one or multiple Hardware Management Consoles.

Set up a group called hmcs (or similar) so you can easily perform operations on all HMCs. Each HMC must have its mgt field in the nodehm table, and its nodetype field in the nodetype table set to hmc. The following command sets the field in both tables:

```
# mkdef -t node mgt=hmc nodetype=hmc hmcs
Object definitions have been created or modified.
```

A group definition goes into the same tables as the individual nodes. You can use the **tabdump** command on the nodetype and nodehm tables to check their contents, as shown in Example 3-9.

*Example 3-9 Checking the node and group definition*

```
# tabdump nodetype
   #node,os,arch,profile,nodetype,comments,disable
   "hmcs",,,,"hmc",,
# tabdump nodehm
   #node,power,mgt,cons,termserver,termport,conserver,serialport,serial
   speed,serialflow,getmac,comments,disable
   "hmcs",,"hmc",,,,,,,,,
```

Next, define the individual HMCs and assign them into the hmcs group. If you have multiple HMCs, consider assigning them names in sequence, such as hmc1, hmc2, and so on. That way, you can specify them with a range but you can also list them individually in a comma separated list. These names will have to resolve to IP addresses so add them to /etc/hosts. If all of your HMCs are using the same username and password then you can add that here too. You can always edit the individual entries in the ppchcp table later.

To define hmc1, hmc2, and hmc3, use one of the following commands:

- `# nodeadd hmc1-hmc3 groups=all,hmcs ppchcp.username=hscroot ppchcp.password=abc123`
- `# nodeadd hmc1,hmc2,hmc3 groups=all,hmcs ppchcp.username=hscroot ppchcp.password=abc123`

These definitions now appear in the tabdump nodelist output shown in Example 3-10.

*Example 3-10 The HMC definitions*

```
#node,groups,status,comments,disable
"hmc1","all,hmcs",,,
"hmc2","all,hmcs",,,
"hmc3","all,hmcs",,,
```

HMC authentication information stored in the ppchcp table is displayed in Example 3-11.

*Example 3-11 HMC access information*

```
# tabdump ppchcp
#hcp,username,password,comments,disable
"hmc1","hscroot","abc123",,
"hmc2","hscroot","abc123",,
"hmc3","hscroot","abc123",,
```

### 3.2.5 Running discovery (rscan command)

Use the **rscan** command to query the HMCs for frame information. You can use the group name hmcs to query all HMCs at once, or you can query individual HMCs, as shown in Example 3-12.

You have the choice to store this data directly in the xCAT database. However, if you have converted from CSM using the conversion tool, the LPAR names will conflict and you will end up with duplicates, so you should not run both the conversion tool and the **rscan** command.

*Example 3-12 Querying frame information using the rscan command*

```
# rscan hmc1
type    name          id      type-model   serial-number  address
hmc     hmc1                  7310-C03     KCWC22A        hmc1
fsp     p6_520_itso           8203-E4A     10E4B61        10.0.0.254
lpar    p6_p7         8       8203-E4A     10E4B61
lpar    p6_p6         7       8203-E4A     10E4B61
lpar    p6_p5         6       8203-E4A     10E4B61
lpar    p6_p4         5       8203-E4A     10E4B61
lpar    p6_p3         4       8203-E4A     10E4B61
lpar    p6_p2         3       8203-E4A     10E4B61
lpar    VIOS_p6       2       8203-E4A     10E4B61
lpar    p6_p1         1       8203-E4A     10E4B61
```

If you decide to use the **rscan** method, you can use the **-w** option to write the data directly to the xCAT database, but we recommend you use an intermediary file so you can make any modifications before finalizing the definitions. The default LPAR names might contain the underscore (_) character, which can cause problems with DNS.

To filter out undesired characters, use the **sed** command, for example:

```
# rscan hmcs -z | sed s/_//g > /tmp/scanout
```

This deletes any occurrences of the underscore character.

**Note:** You should inspect the intermediary file carefully and compare it to the generated one (before running the **sed** command) to make sure it hasn't made any unwanted modifications. You may have to modify the example sed or manually edit the file.

When you are satisfied with the intermediary file, pipe it to the **mkdef** command to create the definitions:

```
# cat /tmp/scanout | mkdef -z
Object definitions have been created or modified.
```

To see the node definitions it created, use the **nodels** command, shown in Example 3-13:

*Example 3-13 Listing xCAT nodes*

```
# nodels
hmc1
hmc2
hmc3
VIOSp5
p6520itso
p6p7
p6p6
p6p5
p6p4
p6p3
p6p2
VIOSp6
p6p1
```

View the details of the individual LPARs (nodes) by using the **lsdef** command, as shown in Example 3-14:

*Example 3-14 Checking individual node definition*

```
# lsdef p6p5
Object name: p6p5
        groups=lpar,osi,all
        hcp=hmc1
        id=6
        mgt=hmc
        nodetype=lpar,osi
        parent=p6520itso
        pprofile=p5profile
```

Based on system configuration, the rscan command has discovered the p6 CEC. The mkdef command takes this information into account and automatically adds the LPARs to the groups: lpar, osi, and all. The fsp group is also created and populated with CEC's Flexible Service Processor (FSP) information.

Check the FSP information created for this CEC as shown in Example 3-15.

*Example 3-15 Checking the FSP information*

```
# lsdef p652Oitso
Object name: p652Oitso
        groups=fsp,all
        hcp=hmc1
        mgt=hmc
        mtm=82O3-E4A
        nodetype=fsp
        serial=1OE4B61
```

### 3.2.6 Deleting the management node from the database

If applicable, delete the management node from the database. There is really no advantage in having the management node defined as a managed node in the xCAT cluster. The management node cannot install itself, for example. If the management node was a managed node in the CSM cluster, or it is one of the LPARs and inserted into the database using rscan, then it has been defined as a managed node. You can easily remove it by using the **rmdef** command:

```
# rmdef p6p2
```

If you decide to keep your xCAT management node defined as a managed node, a recommendation is to differentiate it from the other managed nodes with node groups so you can perform operations without affecting the management node. Use the **tabedit nodelist** command to accomplish this.

### 3.2.7 Populating tables, if you used the conversion tool

In migrating the node definitions from CSM, if you used the rscan utility to scan the hardware for the initial data (initializing the xCAT database), skip this section. You do not have to populate the tables.

**Note:** Follow the instructions in this section only if you have converted from CSM using the conversion tool.

xCAT keeps definitions of the FSPs in the node database, but CSM did not. For each FSP, you must enter its name (as known by the HMC), its HMC (ppc.hcp), its serial number, and its mtm (machine type and model, also known as M/T).

Since this paper was written, the conversion tool might have been enhanced to enter FSP data automatically. To check this, run the **lsdef** command and provide

an FSP name as an argument. To determine the FSP names, you can run the **rscan** command and check its output, or you can find the same information from the HMC graphical user interface.

Figure 3-3 displays the FSP names for the systems connected to our HMC. These are p550_itso1, p550_itso2, and p6_520_itso. The panel on the right shows the LPARs of FSP p6_520_itso.

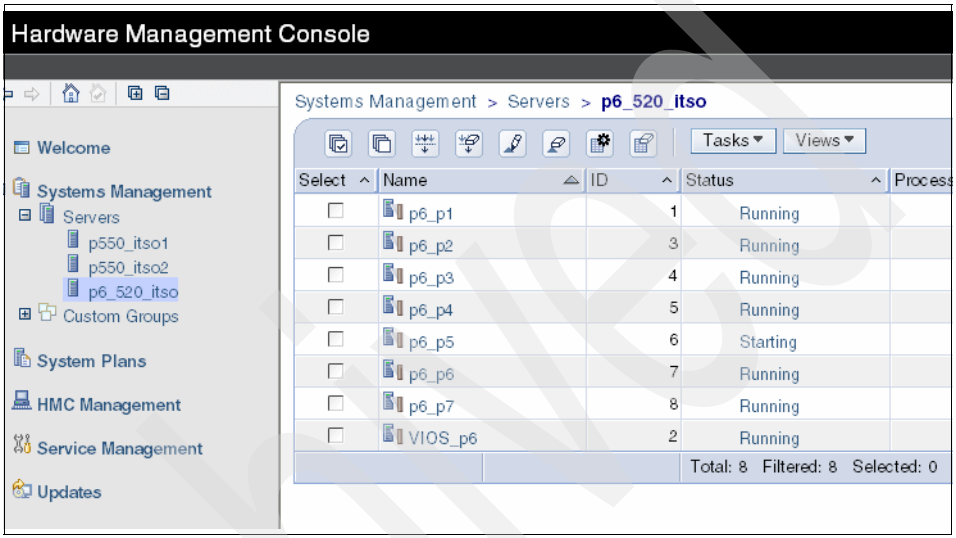


*Figure 3-3 Checking the FSP names in HMC interface*

You can see whether the data was already stored in the xCAT database by entering:

```
# lsdef p6_520_itso
```

If you select one of the LPARs, for example p6_p1, a pop-up window indicates the FSP serial number and M/T (machine type and model), listed in the System field. See Figure 3-4 on page 49.

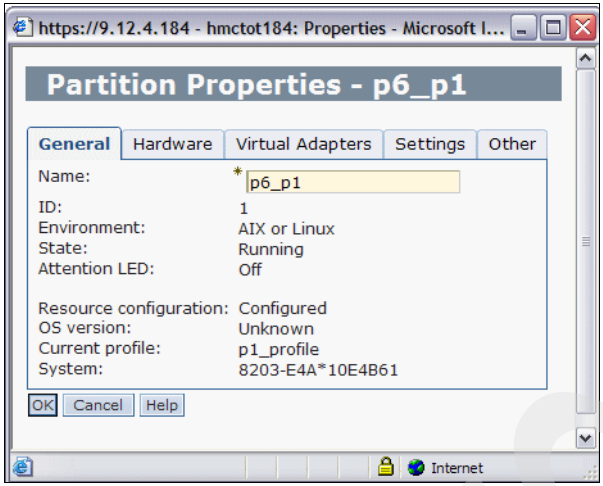


*Figure 3-4 FSP and machine type and model information*

In Figure 3-4, the serial number is 10E4B61 and the M/T is 8203-E4A. Note that this is the same for all the LPARs in this FSP.

Next, run the **nodeadd** command to enter the information into the database. For example:

```
# nodeadd p6_520_itso groups=fsp,all nodehm.mgt=hmc
nodetype.nodetype=fsp ppc.hcp=hmc1 vpd.serial=10E4B61 vpd.mtm=8203-E4A
```

You also have to fill in the ppc table. For each LPAR you must provide:

- HMC name (hcp)
- Slot number (ID)
- Profile (pprofile)
- FSP name (parent)

This information can be obtained from the HMC interface (see Figure 3-4, Information is available in the other tabs, such as Hardware, Virtual Adapters, and others) or from the rscan output. In our case we have decided to manually enter data using the **tabedit ppc** command. When finished, we checked the ppc table contents by using the command shown in Example 3-16.

*Example 3-16 Contents of ppc table*

```
# tabdump ppc
#node,hcp,id,pprofile,parent,comments,disable
"p6_520_itso","hmc1",,,,,
"virtp6p7","hmc1","8","p6_profile","p6_520_itso”,,
"virtp6p6","hmc1","7","p6_profile","p6_520_itso",,
"virtp6p5","hmc1","6","p5_profile","p6_520_itso",,
```

```
"virtp6p4","hmc1","5","p4_profile","p6_520_itso",,
"virtp6p3","hmc1","4","p3_profile","p6_520_itso",,
"virtp6p2","hmc1","3","p2_profile","p6_520_itso",,
"VIOS_p6","hmc1","2","vios_p6","p6_520_itso",,
"virtp6p1","hmc1","1","p1_profile","p6_520_itso",,
```

**Note:** The conversion tool might have already filled in the hcp field of the ppc table. Make sure this field contains the HMC name and *not* the IP address, or this can cause the **rpower** command to fail.

## 3.2.8 Configuring DNS

The host names of the nodes must be resolvable for **rpower** and **rconsole** to work. Make sure that all the LPARs and HMCs as known by xCAT (as shown in the **nodels** command output) have valid entries in /etc/hosts. Make sure your nameserver can also resolve these names. If you plan to set up DNS from scratch on the xCAT Management Node follow the instructions in section 4.1.12, "Setting up network name resolution" on page 86. If you are using a different machine as your name server make sure /etc/resolv.conf points to it. In our environment, we used the following command:

```
# cat /etc/resolv.conf
search itso.ibm.com
nameserver 192.168.100.52
```

Check that nameserver resolution works by using the **nslookup** command:

```
# nslookup virtp6p2
Server:         192.168.100.52
Address:        192.168.100.52#53
Name:   virtp6p2.itso.ibm.com
Address: 192.168.100.53
```

## 3.2.9 Testing the rpower command

After the database has been populated with correct information, you may use the **rpower** command to control node power. You can use the lpar group name to check the power status of all LPARs, as shown in Example 3-17.

*Example 3-17 Checking node power*

```
# rpower lpar stat
VIOS_p6: Running
virtp6p1: Running
virtp6p4: Running
virtp6p5: Running
```

```
virtp6p2: Running
virtp6p7: Open Firmware
virtp6p6: Running
virtp6p3: Running
```

## 3.2.10 Setting up and testing the remote console

Set up and then test the remote console.

### Set up the remote console

First, make sure the `cons` field is set in the nodehm table for the LPARs. For HMC, power method must be set to hmc. Verify this by using the **`tabdump nodehm`** command. You can set the remote console for all LPARS in one step:

```
# chtab node=lpar nodehm.cons=hmc
```

If you use a different machine for your xCAT management node than for the CSM management server, activate the console server as shown in Example 3-18.

*Example 3-18 Activating the console server on the xCAT management node*

```
# service conserver stop
# makeconservercf
# service conserver start
```

If you are migrating by using the same machine (CSM management server), you must upgrade the conserver package (RPM format) to the level that is supplied with xCAT. The conserver levels used by CSM and xCAT are not completely compatible. If you have to return to CSM, you will have to reinstall the older level of conserver. You can find the conserver for xCAT in the following path:

/root/xcat2/xcat-dep/$DISTRO/$ARCH/

For example, change to the directory and run the command:

```
#cd /root/xcat2/xcat-dep/rh5/ppc64
#rpm -U conserver-8.1.16-5.ppc64.rpm
```

Next, start the conserver service:

```
#service conserver stop
#makeconservercf
#service conserver start
```

If you have to go back to the CSM level, run the following commands:

```
# service conserver stop
# rpm -i --force conserver-8.1.7-2.ppc64.rpm
# rpm -e conserver-8.1.16-5
# rm /root/.consolerc
# rconsolerefresh -r
```

### Test the remote console

To test remote console functionality, press `Enter` to get the login prompt, then use the **`rcons`** command, as shown in Example 3-19.

*Example 3-19 Checking remote console*

```
# rcons virtp6p2
[Enter `^Ec?' for help]

Red Hat Enterprise Linux Server release 5.1 (Tikanga)
Kernel 2.6.18-53.el5 on an ppc64

virt-p6-p2 login:
```

## 3.2.11 Copying the distribution source

Copy the distribution RPMs by using the **`copycds`** command. By default the images are copied to /install. You can change the default by changing the installdir field in the site table.

The following example assumes you have a copy of the ISO DVD in /tmp. In this example, the files are copied to /install/rhels5.1/ppc64. Note that CSM also has a **`copycds`** command. Make sure you are using the correct one.

```
# copycds /tmp/RHEL5.1-Server-20071017.0-ppc-DVD.iso
Copying media to /install/rhels5.1/ppc64/
Media copy operation successful
```

If you use multiple Linux distributions, repeat the copying process for each one.

## 3.2.12 Specifying other installation information

This section describes how to enter additional installation information, such as node characteristics, root password, and resources to the xCAT database.

### Specify node characteristics

Enter the operating system, architecture, profile, and node type values into the nodetype table. For example, add the following line to the nodetype table by using the **tabedit nodetype** command:

```
# tabedit nodetype "lpar","rhels5.1","ppc64",”compute”
```

Using `lpar` for the node value causes this command line to apply to all the nodes in the group named lpar. Our homogeneous cluster contains all System p LPARs (ppc64) that will be installed with Red Hat Enterprise Linux Server 5.1 (rhels5.1) using the compute kickstart template. You can create different entries for different distributions and nodes as you need them. Remember that a more specific entry (nodename) overrides a more general one (group name).

### Specify installation root password

Add the following line to the passwd table to set the root password to `cluster` on nodes during installation:

```
"system","root","cluster"
```

### Specify node installation resources (noderes table)

You must specify the management network adapter (primarynic). You must also specify the installation adapter (installnic) if other than primarynic. For Power Systems, the netboot field must be set to `yaboot`. The noderes.nfsserver value should be set to the IP address of your management server, for example:

```
# chtab node=lpar noderes.netboot=yaboot noderes.installnic=eth0 \
noderes.primarynic=eth0 noderes.nfsserver=192.168.100.55
```

## 3.2.13 Acquiring node MAC addresses

Enter the LPAR’s installation adapter MAC addresses if they are not already in the database. To check if they are in the database, use the **tabdump mac** command. To obtain the addresses, use the **getmacs** command.

**Note:** Using the **getmacs** command will reboot your LPARs. If you have another way of obtaining them, you can enter them directly into the mac table.

Enter the MAC addresses using the colon-format, for example:

```
AA:BB:CC:DD:EE:FF
```

At the time of this writing, we could only populate the entries for eth0. In an LPAR environment, the MAC address of eth1 usually differs only in the last two digits

from eth0, thus, if you have to use eth1, you can write the eth0 MAC address to the database and then modify it manually.

The command shown in Example 3-20 on page 54 obtains the MAC addresses for all nodes in the lpar group by powering them off. Verify this is what you want before proceeding. For example, if your management node is also a managed mode in the lpar group, you should not do it this way because it powers off the management node. You should remove the xCAT management node definition from the managed nodes database or from the lpar group.

*Example 3-20 Obtaining LPAR MAC addresses*

```
# rpower lpar off
# getmacs lpar
virtp6p4: mac.mac set to 2E:D2:52:DA:B9:02
virtp6p5: mac.mac set to 2E:D2:5D:DA:41:02
virtp6p3: mac.mac set to 2E:D2:56:6B:E4:02
virtp6p2: mac.mac set to 2E:D2:58:BF:B7:02
virtp6p7: mac.mac set to 2E:D2:50:F6:F7:02
virtp6p6: mac.mac set to 2E:D2:54:FF:5B:02
```

## 3.2.14 Configuring DHCP

If the bind-chroot package (RPM) is installed, remove it because it interferes with xCAT's DNS configuration.

If security-enhanced Linux (SELinux) is enabled, it causes problems with dhcpd. You disable it by editing /etc/selinux/config, then rebooting.

In /etc/selinux/config, look for either of the following lines:

- `SELINUX=permissive`
- `SELINUX=enforcing`

Change the line to the following example, then reboot the management node:

```
SELINUX=disabled
```

### Add entries to the networks table

At this time, the networks table has been partially filled in (at installation time). Check its contents as shown in Example 3-21.

*Example 3-21 Contents of network table*

```
# tabdump networks
#netname,net,mask,mgtifname,gateway,dhcpserver,tftpserver,nameservers,
```

```
dynamicrange,nodehostname,comments,disable
,"192.168.100.0","255.255.255.0","eth0",,,"192.168.100.55",,,,,
```

If the table has no entry for the xCAT installation or management network, add one. For the installation network, assign the IP address of the management node to the DHCP server and gateway fields. Also specify a dynamic node range for DHCP to use during the initial network boot. You should have at least as many dynamic addresses as there are nodes but make sure none of them are being used. You can use the **chtab** command, as shown in Example 3-22.

*Example 3-22 Changing the gateway and DHCP dynamic address range*

```
# chtab mgtifname=eth0 networks.dhcpserver=192.168.100.55
networks.gateway=192.168.100.55
networks.dynamicrange=192.168.100.100-192.168.100.150
```

## Configure DHCP service

If you are using the CSM management server as your xCAT management node, you must stop dhcpd, move the following files, and then save them in case you have to revert to CSM:

- /etc/dhcpd.conf
- /var/lib/dhcpd/dhcp.leases

To configure dhcp for xCAT, use the following command:

```
# makedhcp -n
```

This starts dhcpd and generates the /etc/dhcpd.conf and an empty /var/lib/dhcpd/dhcpd.leases files. The **makedhcp** command adds lines to the /etc/dhcpd.conf file to specify the network boot mechanism, as shown in Example 3-23.

*Example 3-23 /etc/dhcpd.conf file stanza*

```
if option client-architecture = 00:00 { #x86
       filename "pxelinux.0";
} else if option client-architecture = 00:02 { #ia64
       filename "elilo.efi";
} else if substring(filename,0,1) = null { #otherwise, provide yaboot if the
client isn't specific
       filename "/yaboot";
}
```

**Note:** For ppc64 architecture, /yaboot is always be used.

## 3.2.15 Installing diskful compute nodes

At this point, you can install nodes that will run the operating system on their own disks (diskful). To do this, run the **nodeset** command to set the node state to `install` and then issue the **rnetboot** command. For example:

```
# nodeset virtp6p3 install
virtp6p3: install rhels5.1-ppc64-compute
```

This adds a new entry to /var/lib/dhcpd/dhcpd.leases, as shown in Example 3-24:

*Example 3-24 The dhcpd.leases stanza added by xCAT commands*

```
host p6p3 {
    dynamic;
    hardware ethernet 2e:d2:56:6b:e4:02;
    fixed-address 192.168.100.54;
        supersede host-name = "p6p3";
        supersede server.next-server = c0:a8:64:37;
}
```

Make sure that the MAC address (`hardware ethernet`) is correct. There should also be a file in /tftpboot/etc that matches the hex number in the last line of the stanza (c0a86437). The command also creates, in the same directory, a kickstart file with the same name as the node (p6p3), as shown in Example 3-25.

*Example 3-25 Kickstart configuration file*

```
# cat /tftpboot/etc/p6p3
#install rhels5.1-ppc64-compute
timeout=5
image=xcat/rhels5.1/ppc64/vmlinuz
        label=xcat
        initrd=xcat/rhels5.1/ppc64/initrd.img
        append="nofb utf8
ks=http://192.168.100.55/install/autoinst/p6p3 ksdevice=eth0 noipv6"
```

The image (kernel) file and the ramdisk (initrd) are relative to /tftpboot. The p6p3 file referred to in the last line is the kickstart template file.

We recommend powering the node off first before netbooting:

```
# rpower virtp6p3 off
virtp6p3: Success
# rnetboot virtp6p3
virtp6p3: Success
```

After the **rnetboot** command has returned, you can open a console to watch the installation progress. If you want to see the details of the open firmware handshaking between the HMC and the LPAR, run **rnetboot -V** command.

## 3.2.16 Building the diskless image and netbooting nodes

The three types of stateless diskless images differ in the amount of RAM available:

- First type: The RAM size is equal to the size of the image file system.
- Second type: squashfs with RAM overlay provides substantially more RAM by using a compressed file system.
- Third type: NFS Hybrid uses a read-only NFS mount to further reduce memory consumption, but puts greater load on the NFS sever.

This section presents the first two types. NFS Hybrid requires patching the kernel, not currently supported with Red Hat. To learn more about NFS Hybrid and diskless in general, see the *xCAT 2 Cookbook for Linux*, by accessing the following Web page:

`http://xcat.sourceforge.net`

The cookbook also discusses the case in which the image you want to install does not match the operating system or architecture of the management node. (In the cookbook, see the section about building the stateless image.)

### Generate the image

Change to the following directory:

`/opt/xcat/share/xcat/netboot/$osname`

The `$osname` is your operating system name, for example `rh`.

Run the **genimage** command to generate the image, specifying:

`-i` The network boot interface; for example `eth0`

`-n` The network driver. For virtual Ethernet, this is ibmveth.

`-o` The opsys name. This should match the os field in the nodetype table, in our case, rhels5.1 is the name.

`-p` The profile. This can be either service or compute. We do not address service nodes in this scenario.

The output of this command (shown in Example 3-26 on page 58) is quite lengthy and has been selectively edited for brevity.

*Example 3-26 Generating diskless node boot/load image*

```
# ./genimage -i eth0 -n ibmveth -o rhels5.1 -p compute
Setting up Install Process
Setting up repositories
rhels5.1-ppc64-0          100% |=========================| 1.1 kB    00:00
Reading repository metadata in from local files
Primary.xml.gz            100% |=========================| 805 kB    00:00
2825/2825
Parsing package install arguments
Resolving Dependencies
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for nfs-utils to pack into transaction set.
nfs-utils-1.0.9-24.el5.pp 100% |=========================|  33 kB    00:00
---> Package nfs-utils.ppc 1:1.0.9-24.el5 set to be updated
---> Downloading header for dhclient to pack into transaction set.
dhclient-3.0.5-7.el5.ppc. 100% |=========================|  32 kB    00:00
---> Package dhclient.ppc 12:3.0.5-7.el5 set to be updated
---> Downloading header for kernel to pack into transaction set.
kernel-2.6.18-53.el5.ppc6 100% |=========================| 243 kB 00:00

......... <<<<< OMITTED LINES >>>>> .......................

--> Processing Dependency: libwrap.so.0 for package: openssh-server
--> Processing Dependency: libc.so.6(GLIBC_2.2) for package: wget
--> Processing Dependency: libtermcap.so.2 for package: bash
--> Processing Dependency: libc.so.6(GLIBC_2.3.4) for package: dhclient

......... <<<<< OMITTED LINES >>>>> .......................

Dependencies Resolved
============================================================= Package
Arch      Version         Repository        Size
=============================================================
Installing:
bash                    ppc        3.1-16.1         rhels5.1-ppc64-0  1.9 M
busybox-anaconda        ppc        1:1.2.0-3        rhels5.1-ppc64-0  600 k
...
util-linux              ppc        2.13-0.45.el5    rhels5.1-ppc64-0  1.9 M
zlib                    ppc        1.2.3-3          rhels5.1-ppc64-0   53 k
zlib                    ppc64      1.2.3-3          rhels5.1-ppc64-0   56 k

Transaction Summary
=============================================================
Install    109 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 120 M
Downloading Packages:
```

```
Running Transaction Test
warning: e2fsprogs-libs-1.39-10.el5: Header V3 DSA signature: NOKEY, key ID
37017186
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
Installing: libgcc                       ##################### [  1/109]
Installing: setup                        ##################### [  2/109]

......... <<<<< OMITTED LINES >>>>> ........................

Installing: kernel                       ##################### [109/109]
Installed: bash.ppc 0:3.1-16.1 busybox-anaconda.ppc 1:1.2.0-3 dhclient.ppc
12:3.0.5-7.el5 kernel.ppc64 0:2.6.18-53.el5 nfs-utils.ppc 1:1.0.9-24.el5
ntp.ppc 0:4.2.2p1-7.el5 openssh-clients.ppc 0:4.3p2-24.el5 openssh-server.ppc
0:4.3p2-24.el5 stunnel.ppc 0:4.15-2 vim-minimal.ppc 2:7.0.109-3.el5.3 wget.ppc
0:1.10.2-7.el5
......... <<<<< OMITTED LINES >>>>> ........................

readline.ppc 0:5.1-1.1 redhat-release.ppc 0:5Server-5.1.0.2 sed.ppc
0:4.1.5-5.fc6 setup.noarch 0:2.5.58-1.el5 shadow-utils.ppc 2:4.0.17-12.el5
sysklogd.ppc 0:1.4.1-40.el5 tar.ppc 2:1.15.1-23.0.1.el5 tcp_wrappers.ppc
0:7.6-40.4.el5 termcap.noarch 1:5.5-1.20060701.1 tzdata.noarch 0:2007d-1.el5
udev.ppc 0:095-14.9.el5 util-linux.ppc 0:2.13-0.45.el5 zlib.ppc 0:1.2.3-3
zlib.ppc64 0:1.2.3-3
Complete!
9916 blocks
```

## Change the /etc/fstab file

Navigate to the following location:

/install/netboot/$OS/$ARCH/compute/rootimg/etc

$OS This is your operating system name, for example rhels5.1
$ARCH This is the architecture name, in this case ppc64.

For example:

```
# cd /install/netboot/rhels5.1/ppc64/compute/rootimg/etc
```

We modified the /etc/fstab as shown in Example 3-27.

*Example 3-27 Modified /etc/fstab*

```
# cat /etc/fstab
proc /proc proc rw 0 0
sysfs /sys sysfs rw 0 0
devpts /dev/pts devpts rw,gid=5,mode=620 0 0
#tmpfs /dev/shm tmpfs rw 0 0
```

```
compute_ppc64 / tmpfs rw 0 1
none /tmp tmpfs defaults,size=10m 0 2
none /var/tmp tmpfs defaults,size=10m 0 2
```

## Pack the boot image

Run the **packimage** command specifying the operating system (such as rhels5.1), the profile (such as compute) and the architecture (such as ppc64):

```
# packimage -o rhels5.1 -p compute -a ppc64
Packing contents of /install/netboot/rhels5.1/ppc64/compute/rootimgp
```

## Netboot the node

We suggest to try one node first before netbooting multiple nodes at once:

```
# nodeset p6p5 netboot
p6p5: netboot rhels5.1-ppc64-compute
```

This causes an entry to be added to the dhcpd.leases file similar to the one shown in the diskful example (in 3.2.15, "Installing diskful compute nodes" on page 56). However, this time the contents of the /tftpboot/etc file (Example 3-28) is different because: it is now pointing to the diskless kernel (image) and ramdisk (initrd); and, instead of pointing to the kickstart template on the last line, it points to the root image.

*Example 3-28 Boot config file for diskless compute node*

```
# netboot rhels5.1-ppc64-compute
timeout=5
image=xcat/netboot/rhels5.1/ppc64/compute/kernel
        label=xcat
        initrd=xcat/netboot/rhels5.1/ppc64/compute/initrd.gz

append="imgurl=http://192.168.100.55/install/netboot/rhels5.1/ppc64/com
pute/rootimg.gz "
```

We have found it to work better if we power off before netboot:

```
# rpower p6p5 off
p6p5: Success
# rnetboot p6p5
p6p5: Success
```

After you verify that all is well, you can repeat for the rest of the nodes by using noderanges. The following example, sets the boot response to netboot for all nodes in the group compute, except p6p5 (which is our management node):

```
# nodeset compute,-p6p5 netboot
```

### 3.2.17 Building the compressed image for diskless compute nodes

Read this section if you want to use squashfs for your diskless image to make more RAM in your nodes available to running applications.

On your xCAT management node, install kernel-devel, gcc, and squashfs-tools package, if they are not already installed:

1. Download aufs-2-6-2008.tar.bz2 and aufs-standalone.patch into /tmp/aufs from:

   http://xcat.svn.sourceforge.net/svnroot/xcat/xcat-dep/trunk/aufs/

2. Extract the files, as shown in Example 3-29.

*Example 3-29 Unpacking the aufs packages*

```
# tar jxvf aufs-2-6-2008.tar.bz2
aufs/
aufs/patch/
aufs/patch/ubuntu-2.6.24-5.8.patch
aufs/patch/sysfs_get_dentry.patch
aufs/patch/rt-compat.patch
...
aufs/sample/watchguard/probe/probe.c
aufs/sample/watchguard/probe/probe.h
aufs/sample/watchguard/probe/Makefile
aufs/Kconfig.in
aufs/sflogo.html
```

3. Apply the patch using the commands shown in Example 3-30.

*Example 3-30 Applying aufs patch*

```
# cd aufs
# mv include/linux/aufs_type.h fs/aufs
# cd fs/aufs
# patch -p1 < ../../../aufs-standalone.patch.txt
patching file Makefile
patching file aufs.h
patching file aufs_type.h
patching file branch.h
patching file build.sh
patching file cpup.h
patching file dentry.h
patching file dir.h
patching file file.h
patching file hinode.h
patching file inode.h
patching file misc.h
```

```
patching file opts.h
patching file super.h
patching file whout.h
```

4. And build the patch, as shown in Example 3-31.

*Example 3-31 Building the aufs patch*

```
chmod +x build.sh
# ./build.sh
make: Entering directory `/usr/src/kernels/2.6.18-53.el5-ppc64'
/tmp/aufs/aufs/fs/aufs/Makefile:56: Ignoring TMPFS_MAGIC
CC [M]  /tmp/aufs/aufs/fs/aufs/module.o
CC [M]  /tmp/aufs/aufs/fs/aufs/super.o
/tmp/aufs/aufs/fs/aufs/super.c: In function aufs_show_options:
/tmp/aufs/aufs/fs/aufs/super.c:197: warning: format %Lu expects type long
long
...
CC [M]  /tmp/aufs/aufs/fs/aufs/misc.o
LD [M]  /tmp/aufs/aufs/fs/aufs/aufs.o
Building modules, stage 2.
MODPOST
CC      /tmp/aufs/aufs/fs/aufs/aufs.mod.o
LD [M]  /tmp/aufs/aufs/fs/aufs/aufs.ko
make: Leaving directory `/usr/src/kernels/2.6.18-53.el5-ppc64'
```

5. Generate the ramdisk image, as shown in Example 3-32:

*Example 3-32 Generating the ramdisk image*

```
# strip -g auto.ko
# cp aufs.ko /opt/xcat/share/xcat/netboot/rh
# cd /opt/xcat/share/xcat/netboot/rh
# ./geninitrd -i eth0 -n ibmveth,squashfs,aufs,loop -o rhels5.1 -p compute
-l $(expr 100 \* 1024 \* 1024)
15354 blocks
```

**Note:** The argument `-l $(expr 100 \* 1024 \* 1024)` represents the size of the /file system when expanded in the nodes' RAM.

6. Pack the image by using the following command:

```
# packimage -a ppc64 -o rhels5.1 -p compute -m squashfs
Packing contents of /install/netboot/rhels5.1/ppc64/compute/rootimg
```

### 3.2.18 Netbooting the nodes

We recommend testing the diskless image on one node first:

```
# nodeset p6p5 netboot
p6p5: netboot rhels5.1-ppc64-compute
```

This time, the /tftpboot/etc file that is generated looks nearly identical to the regular diskless scenario. The only difference is that it now points to the squashed root image (rootimg.sfs), as shown in Example 3-33.

*Example 3-33 The /tftpboot/etc file for diskless nodes using squashfs*

```
#netboot rhels5.1-ppc64-compute
timeout=5
image=xcat/netboot/rhels5.1/ppc64/compute/kernel
        label=xcat
        initrd=xcat/netboot/rhels5.1/ppc64/compute/initrd.gz

append="imgurl=http://192.168.100.55/install/netboot/rhels5.1/ppc64/com
pute/rootimg.sfs "
```

Next, netboot the node, as follows:

```
# rpower p6p5 off
p6p5: Success
# rnetboot p6p5
p6p5: Success
```

When the test is complete, use node ranges to repeat on multiple nodes.

## 3.3 Adding an existing GPFS to a diskless xCAT cluster

The idea of this scenario is that your nodes can remain diskless and stateless with the ability to mount your GPFS file systems.

### 3.3.1 Prerequisites

On diskless stateful nodes running GPFS, only two directories on nodes are required for the node to be stateful:

- `/var/mmfs`, which contains GPFS configuration
- `/var/adm`, which contains GPFS functional log files

One particular aspect of GPFS on Linux Open Source portability layer. This consists of several modules that are loaded into the Linux kernel when GPFS daemon starts. The portability layer modules must be compiled against the Linux kernel version, which is used by the diskless image.

**Note:** The kernel used by the diskless image might be different from the one running on the xCAT management server.

Usually, diskless images are small and limited in space to minimize memory consumption on nodes. This means that all unnecessary packages are not installed into the image, including development packages that are required to build the GPFS kernel portability layer.

To address this issue we have developed a utility named incorporategpfs that installs GPFS inside a diskless image. We cannot use yum or YaST to install GPFS because its installation requires a special procedure: Install base GPFS RPM from CDs supplied by IBM; update it to the current program temporary fix (PTF) version using updated RPMs (downloaded from the IBM Web site); and compile GPFS modules. Support and downloads are at the following Web page:

`http://www14.software.ibm.com/webapp/set2/sas/f/gpfs/home.html`

**Note:** You must compile the portability layer modules with every PTF level that will be applied to your GPFS cluster.

To compile the GPFS kernel portability layer, we created an additional diskless node image (sandbox) called $profile.devel that has all the prerequisites required to build GPFS portability layer. The typical size of this *devel* image, based on the default compute profile, is about 700 MB for RHEL 5 Server.

**Note:** We have developed a Perl script named incorporategpfs utility that we tested for RHEL Server 5.1 on both x86 and ppc64 platforms. Diskless nodes are not yet supported for SLES distributions. You can find this utility in the additional material:

`ftp://www.redbooks.ibm.com/redbooks/REDP4437/xCAT_ITSO_tools.tar`

The development image tree is created inside a chroot image.

## 3.3.2 Migrating GPFS configuration

To migrate:

1. Generate an image for your GPFS diskless nodes, as follows:
   a. Copy the package:

      ```
      cd /opt/xcat/share/xcat/netboot/rh
      cp compute.pkglist gpfsdiskless.pkglist
      ```

   b. Add the following packages (`compat-libstdc++-33`, `rsh`, `ksh`, `binutils`, `fileutils`) in the `gpfsdiskless.pkglist` file:

      ```
      cp compute.exlist gpfsdiskless.exlist
      ```

      In xCAT 2, removing a directory hierarchy and including a particular directory or file from that hierarchy is not possible. GPFS requires locale en_US.UTF8 and Perl for utilities; by default, Perl does not work in the compute node profile.

      To get Perl working, delete following strings from `gpfsdiskless.exlist` file:

      ```
      ./usr/lib/locale*
      ./usr/lib/perl5*
      ./boot*
      ```

   c. Generate a basic image without GPFS. Example 3-34 is for BladeCenter HS20 (MT 8832) and 32-bit version of RHEL 5 Server

      **Attention:** In RHEL 5 32-bit x86 architecture, standard installation process installs the physical address extension (PAE) kernel. The image generated with the **`genimage`** command installs both *default* and *PAE* kernels. However, during initrd creation, the network modules copied into the initrd image belong to the default kernel, which results in an invalid initrd image (kernel flavor is PAE, whereas the network modules are from default kernel).

      To use the correct kernel and modules, you must specify the full kernel version and flavor, as shown in bold in Example 3-34.

*Example 3-34 Full kernel version*

```
./genimage -i eth0 -n tg3 -p gpfsdiskless -o rhels5 -a x86 -k
2.6.18-53.el5
```

2. Configure Secure Shell (SSH) keys on the management server for password-less access:

```
[root@mgmt ~]# cd ~/.ssh
[root@mgmt .ssh]# cat id_rsa.pub >> authorized_keys
[root@mgmt .ssh]# cp -a authorized_keys id_rsa*
/install/postscripts/_ssh
cp: overwrite `/install/postscripts/_ssh/authorized_keys'? y
[root@mgmt .ssh]# chmod a+r /install/postscripts/_ssh/*
```

3. Set up a Clustered NFS to keep GPFS configuration of diskless nodes on it. Refer to *A Guide to the IBM Clustered Network File System*, REDP-4400.

   We used following configuration:

   | | |
   |---|---|
   | /gpfs | GPFS filesystem |
   | /gpfs/cnf | Directory that is exported with Clustered NFS to nodes. |

   We recommend that you have two GPFS file systems: one for storing user data and another one for keeping GPFS configuration files and logs for diskless nodes.

4. Incorporate GPFS into the diskless image.

   To accomplish this task we developed a utility, incorporategpfs, that incorporates GPFS binaries and makes necessary adjustments to the diskless image to allow GPFS to work in hybrid mode. Hybrid mode means that everything on the compute nodes is stateless except GPFS configuration data and logs.

   The utility performs the following tasks:

   – Checks whether the diskless node image, which was prepared by the genimage utility, has the RPMs required for GPFS
   – Installs (update mode also supported) GPFS in the diskless image.
   – Builds an isolated development environment for your diskless image, installs GPFS in it and builds GPFS Portability Layer modules, and then copies them back to the original diskless image

   **Note:** Cross-compilation is not supported; your management server should have the same architecture as your compute nodes.

   Use the incorporategpfs utility, as follows:

```
cd /opt/xcat/share/xcat/netboot/rh
mkdir GPFS GPFS.update
cp /repo/GPFS_BASE_RPMS/gpfs*.rpm GPFS
cp /repo/GPFS_UPDATE_RPMS/gpfs*update*.rpm GPFS.update
./incorporategpfs -o rhels5 -p gpfsdiskless -k 2.6.18-53.el5
```

Wait until the message "`All done`" appears on your terminal, then continue.

5. Create a configuration repository for your diskless GPFS nodes:

```
# for n in $(seq 1 128); do mkdir -p /gpfs/cnf/node${n}/var/mmfs
/gpfs/cnf/node$n/var/adm; done
```

6. Add a startup script for your GPFS diskless nodes in the `postscripts` table:

   a. Place script `startgpfs` in `/install/postscripts` directory. In the following example, **`cnfs`** is your Clustered NFS server host name:

```
#!/bin/sh
mount -t nfs cnfs:/gpfs/cnf/$NODE/var/mmfs /var/mmfs
mount -t nfs cnfs:/gpfs/cnf/$NODE/var/adm  /var/adm
/usr/lpp/mmfs/bin/mmstartup
```

   b. Enable the post installation script for nodes in group `gpfsdiskless`:

```
# chtab node=gpfsdiskless postscripts.postscripts=startgpfs
```

7. Pack the image for diskless nodes:

```
# packimage -o rhels5 -p gpfsdiskless -a x86
```

8. Boot the GPFS diskless nodes:

```
# nodeset gpfsdiskless netboot
rpower gpfsdiskless boot
```

9. Add the diskless nodes to GPFS cluster and start the GPFS daemon:

```
# for n in $(seq 1 128); do echo "node$n" >> nodes.txt; done
# mmaddnode -N nodes.txt
# mmstartup -N nodes.txt
```

## 3.4 CSM disk-based transition to xCAT diskful nodes

This section describes a transition scenario we tested in our environment. We started with a CSM cluster with each managed node running its own copy of the operating system from the internal disk. We ended up with the same hardware cluster but managed by xCAT. This section discusses how to add nodes without reinstalling the operating system into xCAT 2, as follows.

- Prepare the platform.
- Update the /etc/hosts file.
- Determine nodes attributes.
- Add nodes xCAT 2.
- Check the nodes status.
- Update the SSH.

### 3.4.1 Platform preparation and considerations

For this scenario, we used BladeCenter with IBM System x blades (HS20). Requirements and considerations for preparation are listed in this section.

Before beginning, you should have already set up the xCAT 2 management node on a different machine from the CSM management server. Having both xCAT 2 management node and CSM on the same machine is not yet supported.

In this example, we add a node, which has an operating system already installed by CSM, into xCAT 2 management server. This example is based on the Intel platform.

**Note:** The node that is transferred from CSM to xCAT 2 must have proper network configuration. We are assuming that the node to be transferred to xCAT has been properly removed from the CSM configuration. For details, see the CSM manuals:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.csm.doc/clusterbooks.html

#### Management node

The management node must have a CD-ROM, DVD-ROM or DVD-RAM. A sufficient number of network adapters is required to accommodate the management, cluster, and public virtual local are networks (VLANs). We recommend a System x server with at least one dual-core processor, 1 GB of RAM, and one internal disk.

#### Operating system disk partition

Table 3-1 lists the suggested operating system disk partitions and their sizes.

*Table 3-1 OS disk partitions recommendation*

| Partitions | Size |
|---|---|
| /boot | 100 MB |
| /install | 1 GB plus 3 GB for each 32-bit Linux Distribution, and 4 GB for each 64-bit Linux distribution |
| /opt | 1 GB |
| /var | 1 GB (depending on logging configuration, more may be needed) |
| / | 2 GB |
| swap | 1024 to 2048 MB (depending on RAM size) |

| Partitions | Size |
|---|---|
| /tmp | 512 MB |

We used RHEL Server 5.1 CDs in our test.

### Networking environment

Our networking environment consisted of two VLANs: one for hardware control (RSA, BMC, AMM) and console access, and the other one for node installation and control. See Figure 3-5.

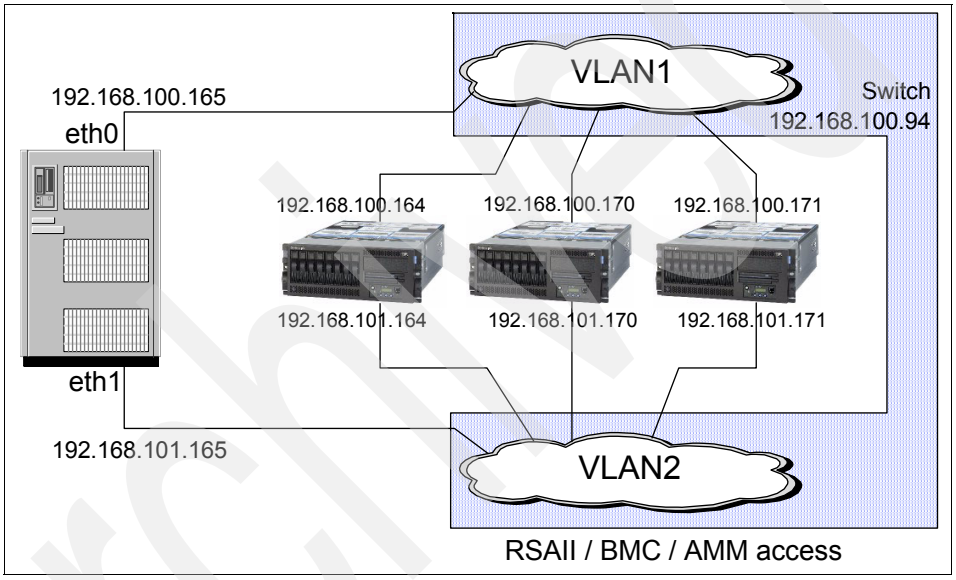


*Figure 3-5 Our test environment*

## 3.4.2 Updating the /etc/hosts file

Update the /etc/hosts file with host names and IP addressees that you will add into the xCAT 2 management node, as shown in Example 3-35.

*Example 3-35 /etc/hosts*

```
[root@xcat2mgmt ~]# cat /etc/hosts
127.0.0.1       localhost.localdomain   localhost
192.168.100.162     xcat2mgmt.itso.ibm.com       xcat2mgmt #XCAT MS
192.168.100.165     hs20b05.itso.ibm.com        hs20b05
192.168.100.164     hs20b04.itso.ibm.com        hs20b04
```

```
192.168.100.170     hs21b10.itso.ibm.com        hs21b10
192.168.101.91      blademm.itso.ibm.com        blademm
```

## 3.4.3 Determining the node attributes

As part of the xCAT 2 cluster, gather information that will be used to add nodes into xCAT 2. To collect the node's minimum required attributes for xCAT 2 tables see Table 4-2 on page 114.

## 3.4.4 Adding nodes to xCAT 2

You may add a single node or group of nodes.

### Add a single node

Add nodes into xCAT 2. Run the **nodeadd** command to add a single node into xCAT 2 management server as shown in Example 3-36.

*Example 3-36 Adding a single node into xCAT 2*

```
[root@xcat2mgmt ~]#nodeadd hs20b04 groups=blade,compute \ mp.mpa=192.168.100.91
nodehm.power=blade nodehm.mgt=blade \ nodetype.os=rhels5 nodetype.arch=x86
nodetype.profile=compute \ nodetype.nodetype=osi noderes.nfsserver=hs20b05
noderes.primarynic=eth1
```

If you want to change or add any attributes, use the **nodech** command. After adding the nodes or changing the attributes, run either the **tabdump** or **nodels** command to verify the node attributes in the xCAT tables, as shown in Example 3-37.

*Example 3-37 Checking table and node attributes*

```
[root@xcat2mgmt ~]# tabdump networks
#netname,net,mask,mgtifname,gateway,dhcpserver,tftpserver,nameservers,dynamicra
nge,nodehostname,comments,disable
,"192.168.100.0","255.255.255.0","eth0",,,"192.168.100.162","192.168.100.162","
192.168.100.180-192.168.100.185",,,
,"192.168.101.0","255.255.255.0","eth1",,,"192.168.101.162","192.168.101.162","
192.168.101.180-192.168.101.185",,,
[root@xcat2mgmt ~]#
[root@xcat2mgmt ~]# nodels hs20b04 nodetype
hs20b04: nodetype.profile: compute
hs20b04: nodetype.arch: x86
hs20b04: nodetype.nodetype: osi
hs20b04: nodetype.node: hs20b04
hs20b04: nodetype.os: rhels5
```

```
[root@xcat2mgmt ~]#
[root@xcat2mgmt ~]# nodels hs20b04 noderes
hs20b04: noderes.primarynic: eth0
hs20b04: noderes.netboot: pxe
hs20b04: noderes.nfsserver: 192.168.100.162
hs20b04: noderes.node: hs20b04
[root@xcat2mgmt ~]#
```

### Add a group of nodes

To add a group of nodes, use the **nodeadd** command and specify a node range, as shown in Example 3-38.

*Example 3-38 Adding a group of nodes*

```
[root@xcat2mgmt ~]#nodeadd hs20b03-hs20b05 groups=blade,compute \
mp.mpa=192.168.100.91 nodehm.power=blade nodehm.mgt=blade \ nodetype.os=rhels5
nodetype.arch=x86 nodetype.profile=compute \ nodetype.nodetype=osi
noderes.nfsserver=hs20b05 noderes.primarynic=eth1
```

Then, get the mp.id value from **rscan** command and update the mp table using **nodech** command as shown in Example 4-52 on page 116. Verify that **rpower** command is working properly as shown in Example 4-16 on page 88.

## 3.4.5 Checking status of the nodes

After adding the nodes, check the node's running status by using the **nodestat** command, as shown in Example 3-39.

*Example 3-39 Checking node status*

```
[root@xcat2mgmt ~]# nodestat hs20b04 stat
hs20b04: sshd
[root@xcat2mgmt ~]#
```

Node status `sshd` means the operating system has been installed on the node and it is ready for login by SSH. Table 3-2 shows the xCAT 2 supported node status values.

*Table 3-2 xCAT 2 node status*

| Node status | Description |
|---|---|
| noping | Node is not pingable |
| ping install | Ready for installation |

| Node status | Description |
|---|---|
| installing | Installing the node |
| sshd | Operating system has installed |
| pbs | pbs scheduler is up |

## 3.4.6 Updating the root's SSH key

Finally, as shown in Example 3-40, update the root user's `authorized_keys` file on the node with the public key of the `root@xcat2mgmt node` to be able to log in to the node without a password.

*Example 3-40 Updating ssh key*

```
[root@xcat2mgmt ~]# scp /root/.ssh/id_rsa.pub
hs20b04:/root/.ssh/id_rsa.pub-ms
root@hs20b04's password:
id_rsa.pub                                    100%  394     0.4KB/s
00:00
[root@xcat2mgmt ~]# ssh hs20b04
[root@h20b04 .ssh]# ls
authorized_keys  authorized_keys2  id_rsa.pub-ms
[root@h20b04 .ssh]# cat id_rsa.pub-ms >>authorized_keys
[root@h20b04 .ssh]#
```

After you exchange the keys, try to use SSH to log in to the node. If everything is correct, you are not prompted for a password.

**Note:** As of this writing, xCAT 2 does not exchange the keys automatically as CSM does. That is why you have to exchange keys manually.

4

# Installing xCAT 2 from scratch on diskful nodes

This chapter describes how to install and configure xCAT 2 for an Intel and Power processor-based cluster that previously did not have CSM or xCAT configured. The distribution used in both cases is RHEL 5.1.

The steps in this chapter describe:

- Installing xCAT 2 on Power Architecture blades

  The installation instructions for POWER processor-based architecture were developed for JS20 and JS21 blades.

- Installing xCAT 2 on Intel blades

  The installation instructions for Intel-based architecture were developed for HS20 and HS21 blades.

# 4.1 Installing xCAT 2 on Power Architecture blades

These instructions assume that the base operating system is already installed on the (future) xCAT Management Node. This scenario was developed using a homogeneous cluster of JS21 Power Architecture® blades. The diagram of our test environment is shown in Figure 4-1.

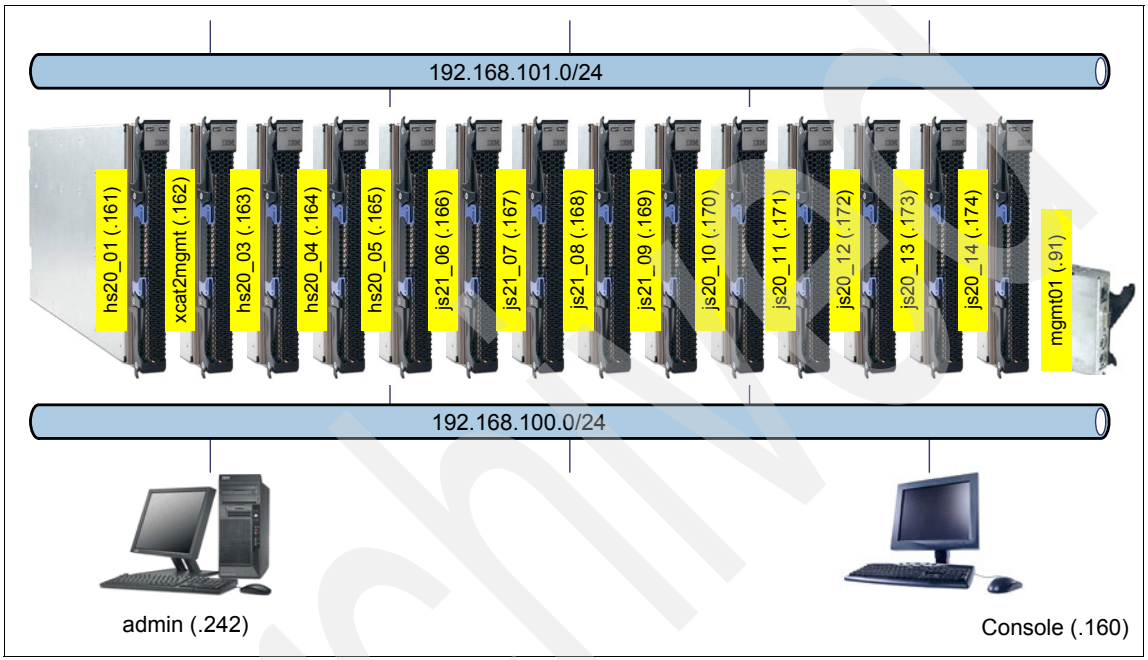


*Figure 4-1 Our test environment*

## 4.1.1 Downloading and extracting the xCAT 2 tarballs

Download the xCAT RPM packages from:

`http://xcat.sourceforge.net/yum/download.html`

Open the source package dependencies that xCAT requires. Store the files in the /root/xcat2 directory on your management node. We downloaded the following compressed tarballs:

- xcat-core-2.0.tar.bz2
- xcat-dep-2.0.tar.bz2

**Note:** The exact navigation instructions can change.

After you have the files in /root/xcat2 on your management node, extract the contents using the **tar jxv**f command, as shown in Example 4-1.Then, repeat the step for xcat-dep-2.0.tar.bz2.

*Example 4-1 Unpacking the xCAT core compressed tarball*

```
# tar jxvf xcat-core-2.0.tar.bz2
xcat-core/
xcat-core/xCAT-2.0-snap200805301026.i386.rpm
xcat-core/repodata/
xcat-core/repodata/other.xml.gz
xcat-core/repodata/repomd.xml
xcat-core/repodata/primary.xml.gz
xcat-core/repodata/filelists.xml.gz
xcat-core/xCAT-nbroot-core-x86-2.0-snap200805211711.noarch.rpm
xcat-core/xCAT-client-2.0-snap200805301555.noarch.rpm
xcat-core/xCAT-2.0-snap200805301026.ppc64.rpm
xcat-core/mklocalrepo.sh
xcat-core/perl-xCAT-2.0-snap200805291415.noarch.rpm
xcat-core/xCAT-2.0-snap200805301026.x86_64.rpm
xcat-core/xCAT-server-2.0-snap200805301357.noarch.rpm
xcat-core/xCAT-nbroot-core-ppc64-2.0-snap200805211711.noarch.rpm
xcat-core/xCAT-nbroot-core-x86_64-2.0-snap200805211711.noarch.rpm
xcat-core/xCATsn-2.0-snap200805211401.x86_64.rpm
xcat-core/xCAT-core.repo
```

## 4.1.2 Installing xCAT 2 on the management node

Prepare yum (Yellow dog Updater, Modified) for installation by running the following commands, shown in Example 4-2.

**Note:** The actual versions might differ from the ones shown in our example.

*Example 4-2 Creating the xCAT yum repository*

```
# cd /root/xcat2/xcat-dep/rh5/ppc64
# ./mklocalrepo.sh
/root/xcat2/xcat-dep/rh5/ppc64
# cd /root/xcat2/xcat-core
#./mklocalrepo.sh
/root/xcat2/core-snap
# yum clean metadata
Loading “installonly” plugin
Loading "security" plugin
```

```
Loading "rhnplugin" plugin
This system is not registered with RHN.
RHN support will be disabled.
0 metadata files removed
```

Install xCAT by running the following command:

```
# yum install xCAT.ppc64
```

The output from the command, shown in Example 4-3, has been selectively edited for brevity.

**Note:** If you did not completing install the operating system, the xCAT installation can fail because of missing dependencies. In our testing, we had the following missing dependencies on a RHEL 5.1 CSM-installed JS21 blade:

```
bind-9.3.3-10.el5.ppc.rpm
dhcp-3.0.5-7.el5.ppc.rpm
expect-5.43.0-5.1.ppc64.rpm
httpd-2.2.3-11.el5.ppc.rpm
net-snmp-perl-5.3.1-19.el5.ppc.rpm
perl-DBI-1.52-1.fc6.ppc.rpm
perl-IO-Socket-SSL-1.01-1.fc6.noarch.rpm
perl-Net-SSLeay-1.30-4.fc6.ppc.rpm
perl-XML-Parser-2.34-6.1.2.2.1.ppc.rpm
perl-XML-Simple-2.14-4.fc6.noarch.rpm
vsftpd-2.0.5-10.el5.ppc.rpm
```

*Example 4-3 Output from yum install .ppc64*

```
# yum install xCAT.ppc64
Loading "installonlyn" plugin
Loading "security" plugin
Loading "rhnplugin" plugin
This system is not registered with RHN.
RHN support will be disabled.
Setting up Install Process
Setting up repositories
cat-dep-snap              100% |=========================|  951 B    00:00
cat-core-snap             100% |=========================|  951 B    00:00
Reading repository metadata in from local files
primary.xml.gz             100% |=========================| 8.5 kB    00:00
############################################### 16/16
primary.xml.gz             100% |=========================| 4.4 kB    00:00
############################################### 10/10
Parsing package install arguments
Resolving Dependencies
```

```
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for xCAT to pack into transaction set.
xCAT-2.0-snap200805151705 100% |=======================|  15 kB 00:00
---> Package xCAT.ppc64 0:2.0-snap200805151705 set to be updated
--> Running transaction check
--> Processing Dependency: xCAT-client for package: xCAT
--> Processing Dependency: perl-DBD-SQLite for package: xCAT
--> Processing Dependency: atftp for package: xCAT
--> Processing Dependency: conserver for package: xCAT
.......... <<<<< Omitted lines >>>>> .........
xCAT is now installed, it is recommended to tabedit networks and set a dynamic
ip address range on any networks where nodes are to be discovered
Then, run makedhcp -n to create a new dhcpd.configuration file, and
/etc/init.d/dhcpd restart
Either examine sample configuration templates, or write your own, or specify a
value per node with nodeadd or tabedit.
Installed: xCAT.ppc64 0:2.0-snap200805151705
Dependency Installed: atftp.ppc64 0:0.7-4 conserver.ppc64 0:8.1.16-5
fping.ppc64 0:2.4b2_to-2 perl-DBD-SQLite.ppc64 0:1.14-1 perl-Expect.noarch
0:1.21-1 perl-IO-Tty.ppc64 0:1.07-1 perl-Net-Telnet.noarch 0:3.03-5.1
perl-xCAT.noarch 0:2.0-snap200805191315 xCAT-client.noarch
0:2.0-snap200805191530 xCAT-server.noarch 0:2.0-snap200805201010
Complete!
```

### 4.1.3 Backing up the original database tables

Immediately back up the xCAT database tables in case you have to restore them to their original state. Some fields are filled in automatically at installation so you cannot simply clear them and start over.

To back up the tables, run the following commands:

```
# mkdir -p /root/xcat2/backups/origina
# dumpxCATdb -p /root/xcat2/backups/original
```

A strong recommendation is to create more backups (in different directories) as you proceed through the installation steps, especially if you are unsure of something, so that you can restore the tables to the previous state if your action does not produce the desired result.

To restore the tables run the following command, where `dir_name` is the directory name where you have previously created the backup:

```
# restorexCATdb -p dir_name
```

### 4.1.4 Setting the xCAT environment variables

A file named xcat.sh was added to /etc/profile.d by the installation. It sets several environment variables for xCAT, including $PATH and $MANPATH:

```
XCATROOT=/opt/xcat
PATH=$PATH:$XCATROOT/bin:$XCATROOT/sbin
MANPATH=$MANPATH:$XCATROOT/share/man
export XCATROOT PATH MANPATH
```

Although it automatically runs at login, you may apply the file to the current login session by using:

```
# source /etc/profile.d/xcat.sh
```

### 4.1.5 Disabling SELinux

If SELinux is enabled during system installation, it conflicts with xCAT programs. Therefore, SELinux must be disabled on the xCAT 2 management server. To disable SELinux, edit the /etc/selinux/config file, as shown in Example 4-30 on page 104.

*Example 4-4 Disabling SELinux in /etc/selinux/config*

```
vi /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#       targeted - Only targeted network daemons are protected.
#       strict - Full SELinux protection.
SELINUXTYPE=targeted
```

**Note:** You might have to reboot the node if you make changes in the/etc/selinux/config file.

## 4.1.6 Seeding the database

The following sections explain how to use the **rscan** command to query the management modules so you can obtain the initial data for the xCAT database, and how to enter this data manually if do not use **rscan**:

- Defining your management modules (MMs)
- Configuring the management module network settings
- Discovering your cluster (scanning)
- Populating the database (using rscan)
- Populating the database manually (no rscan)

**Note:** If you decide to use **rscan**, you must follow several preliminary steps. We suggest reading through this entire section first and deciding which method you think is best for you.

The **rscan** command uses the names of the blades as they are known to the BladeCenter management module, which in our case is the Advanced Management Module (AMM). If it is set to the factory default settings, you may modify them before writing them to the database. We recommend that you use the **dumpxCATdb** command (backup database) often so you can return to a previous state with the **restorexCATdb** command if you change your mind or make a mistake.

## 4.1.7 Defining your management modules (MMs)

First, add the management modules (MMs) to the database, then to the nodehm and mp tables, and finally to /etc/hosts.

### Add the MMs to the database

You can do this manually or use the **mkrrbbc** command (tool). The tool automatically generates nodelist definitions for MMs and switches. The advantage of using this tool is that it creates rack groups for you and can automatically increment the rack number every four MMs, enabling you to perform operations on all MMs in a particular rack. The tool also provides for management subdomains (with the `-C` option), as follows (see the **mkrrbbc** command manpage for more information):

```
# /opt/xcat/share/xcat/tools/mkrrbc -C a -L 1 -R 1,1
```

This creates a definition for one MM and one switch (`-R 1,1`) in sub domain A (`-C a`) in rack one (`-L 1`).

To check the results, use the following command:

```
# tabdump nodelist
#node,groups,status,comments,disable
"bca01","mm,cua,rack01",,,
"swa01","nortel,switch,cua,rack01",,,
```

You can accomplish the same thing using the **nodeadd** command (tool):

```
#nodeadd bca01 groups=”mm,cua,rack01”
#nodeadd swa01 groups=”nortel,switch,cua,rack01”
```

The advantage of the **nodeadd** tool is that you can define multiple MMs and switches in one command. For example, you can add 15 switches and 15 MMs starting in rack two in sub domain B using:

```
/opt/xcat/share/xcat/tools/mkrrbc -C b -L 2 -R 2,16
```

This starts the numbering at bca02 and swa02. It also increments the rack group number every four MMs or switches.

To accomplish the same thing manually you have to run the commands shown in Example 4-5.

*Example 4-5 Manually defining MMs and switches for IBM BladeCenter*

```
# nodeadd bcb02-04 groups=”mm,cub,rack02”
# nodeadd swb02-04 groups=”nortel,switch,cub,rack02”
# nodeadd bcb05-08 groups=”mm,cub,rack03”
# nodeadd swb05-08 groups=”nortel,switch,cub,rack03”
# nodeadd bcb09-12 groups=”mm,cub,rack04”
# nodeadd swb09-12 groups=”nortel,switch,cub,rack04”
# nodeadd bcb13-16 groups=”mm,cub,rack05”
# nodeadd swb13-16 groups=”nortel,switch,cub,rack05”
```

## Add MMs to the nodehm and mp tables

The nodehm table is the hardware management table. You must specify `blade` as the MM's `mgt` field. Because all MMs are in group *mm*, you can create one entry for node mm, which can be used for all of them. To do this use one of the following commands:

- `# chtab node=mm nodehm.mgt=blade`
- `tabedit`

The contents of the nodehm table should look similar to the table in Example 4-6.

*Example 4-6 Contents of nodehm table*

```
#tabdump nodehm
```

```
#node,power,mgt,cons,termserver,termport,conserver,serialport,serialspe
ed,serialflow,getmac,comments,disable
"mm",,"blade",,,,,,,,,
```

You must also create an entry in the mp table:

```
# tabdump mp
#node,mpa,id,comments,disable
“"mm","|(.*)|($1)|",,,
```

This is a regular expression that causes the MM node name to map to the mp.mpa name. As a result, any table scan for mp.mpa of any node in the group mm (bca1, bca2, bca3, and so on) simply returns the node name (bca1, bca2, bca3,...). This is required for **rscan** to work correctly.

## Add the MMs to /etc/hosts

Finally, ensure you have an entry for the MMs in /etc/hosts. In our example, it is:

```
192.168.100.91 bca01.itso.ibm.com bca01
```

If your MM names and their corresponding IP addresses regularly increment, you can use a Perl script to generate the /etc/host entries. This script is shown in Example 4-7.

*Example 4-7 Perl script used to generate /etc/hosts entries*

```
#!/usr/bin/perl
for ( $i = 1; $i <= $ARGV[1]; $i++) {
    $j = $i + $ARGV[0];
    printf ("192.168.100.%d bca%02d.itso.ibm.com bca$02%02d\n",$j,$i ,$i);
}
```

This example hard codes the *base* of the IP address 192.168.100 and the itso.ibm.com domain. The first argument is the offset of the starting IP address; the second argument is the total number of MMs. Change this as necessary for your environment. We ran this script and obtained the results in Example 4-8.

*Example 4-8 Populating the /etc/hosts*

```
#./script 50 16 >> /etc/hosts
....
# cat /etc/hosts
....... <<<<< Omitted lines >>>>> ......
192.168.100.51  bca01.itso.ibm.com bca01
192.168.100.52  bca02.itso.ibm.com bca02
....... <<<<< Omitted lines >>>>> ......
192.168.100.65 bca02.itso.ibm.com bca02
```

**Note:** If the username or password to access the MM was changed from the default, you must manually add it into the passwd table.

### 4.1.8 Configuring the management module network settings

In Example 4-9, we use the group mm to run the commands on all management modules in the mm group. If you want to run it on just one management module, you can use its name instead.

*Example 4-9 Configuring MMs*

```
# rspconfig mm snmpcfg=enable sshcfg=enable
bca01: SNMP enable: OK
bca01: SSH enable: OK

# rspconfig mm pd1=redwoperf pd2=redwoperf
bca01: pd2: redwoperf
bca01: pd1: redwoperf

# rpower mm reset
bca01: reset
```

### 4.1.9 Discovering your cluster (scanning)

You now should be able to run the **rscan** command to discover the blade information, as shown in Example 4-10.

*Example 4-10 Node discovery (rscan command)*

```
# rscan mm
type    name            id      type-model  serial-number  address
mm      SN#0J1U9E5841AA  0       8677-3XU    KPVH850        bca01
blade   SN#ZJ1V5T44415M  1       8832-XX3    6A56860
blade   SN#ZJ1V5T44512L  2       8832-XX3    6A51230
blade   SN#ZJ1TS741S1AP  3       8832-XX3    6A57961
blade   SN#ZJ1V5T43W130  4       8832-XX3    6A52352
blade   SN#ZJ1TS741Y1MZ  5       8832-XX3    6A57946
blade   b03n31           6       8844-51U    KQ0107A
blade   b03n30           7       8844-51X    23A0393
blade   b03n32           8       8844-51X    23A0409
blade   b03n29           9       8844-51X    KQ0109F
blade   SN#YK108069H115  10      8853-ROZ    23A0510
blade   SN#YK308063M103  11      8853-DWZ    23A0368
```

```
blade   SN#ZJ1YEY51H18H  13      8842-21X    KPFYG4D
blade   SN#ZJ1YEY51H18S  14      8842-21X    KPFYG9T
```

## 4.1.10 Populating the database (using rscan)

You may use the **rscan** command with the **-w** option to write the blade data directly to the xCAT database. If you do not want the node names to be the same as they are known to the MM, skip to the next section. If you do decide to use the **rscan** command output, we recommend you create an intermediary file to make modifications before finalizing the definitions.

The default blade names might contain the number sign (#) character, which causes trouble with conserver because it interprets the signs as comment characters. Also, if the MMs have already been defined in a previous step, you will get duplicate definitions. To filter out the undesired entries, use **grep** and **sed** commands with rscan, for example:

```
# rscan mm -z | grep -vwe ^mm | sed s/SN#/SN/ > /tmp/scanout
```

When you consider the intermediary file correct, pipe it to the **mkdef** command to create the definitions:

```
# cat /tmp/scanout | mkdef -z
Object definitions have been created or modified.
```

This creates entries in the mp, nodelist, and nodehm tables. You can view any of these tables using the **tabdump** command, or by running the commands shown in Example 4-11.

*Example 4-11 Checking node information*

```
# nodels
bca01
swa01
SNZJ1V5T44415M
SNZJ1V5T44512L
SNZJ1TS741S1AP
SNZJ1V5T43W130
SNZJ1TS741Y1MZ
b03n31
b03n30
b03n32
b03n29
SNYK108069H115
SNYK308063M103
SNZJ1YEY51H18H
```

```
SNZJ1YEY51H18S

# lsdef SNZJ1YEY51H18S
Object name: SNZJ1YEY51H18S2
    groups=blade,all
    id=14
    mgt=blade
    mpa=bca01
```

If your management node is one of the blades, you probably should remove it:

```
# rmdef Management_Node_name
```

## 4.1.11 Populating the database manually (no rscan)

If you decided not to use **rscan** to populate the database, you have to populate it manually. If your blade names increment regularly, for example blade1, blade2, ..., blade*n*, you can make use of regular expressions when specifying data. First define the nodes, then fill in the mp and nodehm tables.

### Define the nodes

We present three methods for defining nodes:

- Using the mmrrnodes script
- Running **nodeadd** command multiple times (once for each node)
- Writing your own script to run multiple **nodeadd** commands

The mmrrnodes script, which is similar to the mmrbc script, enables you to add nodes. The mmrrnodes script is described in the man page and in the xCAT 2 Cookbook for Linux (xCAT2.pdf), which is downloadable from:

http://xcat.svn.sourceforge.net/svnroot/xcat/xcat-core/trunk/xCAT-client/share/doc/

This script, however, hard codes the node names such as `rra###a` and generates a various group names that you might not be interested in. By defining the groups manually, you have the freedom to choose how to name and group them. You may also modify the mmrrnodes script. The following example shows how to define the nodes manually with groups for the MM and the rack (rack number increments every four MMS):

```
# nodeadd blade1-blade16 nodetype.groups=all,blade,mm01,rack01
# nodeadd blade17-blade32 nodetype.groups=all,blade,mm02,rack01
```

**Note:** We use `mm#` (instead of `bpa##`) as the management module name. This is because we want to keep as separate entities the MM group that the blades belong to and the MM itself.

As an option we have developed our own script, shown in Example 4-12 on page 85.

*Example 4-12 Custom script for defining nodes*

```
#!/usr/bin/perl
$i=$ARGV[0];
for ( $j = 1; $j <= $i/16+1; $j++) {
    $start=($j-1)*16+1;
    $end=$start+15;
    if ($i < $end ) { $end=$i; }
    $frame=($j-1)/4+1;
    $cmd=sprintf(
"nodeadd
blade$start-blade$endnodelist.groups=all,blade,mm%02d,rack%02d"    ,
$j,$frame);
print "$cmd\n";
#       system("$cmd");
```

This script can be modified to your environment, however as shown, it takes a single argument which is the number of blades, as shown in Example 4-13.

*Example 4-13 Running the custom script for node definition*

```
# ./scriptb 132
nodeadd blade1-blade16 nodelist.groups=all,blade,mm01,rack01
nodeadd blade17-blade32 nodelist.groups=all,blade,mm02,rack01
nodeadd blade33-blade48 nodelist.groups=all,blade,mm03,rack01
nodeadd blade49-blade64 nodelist.groups=all,blade,mm04,rack01
nodeadd blade65-blade80 nodelist.groups=all,blade,mm05,rack02
nodeadd blade81-blade96 nodelist.groups=all,blade,mm06,rack02
nodeadd blade97-blade112 nodelist.groups=all,blade,mm07,rack02
nodeadd blade113-blade128 nodelist.groups=all,blade,mm08,rack02
nodeadd blade129-blade132 nodelist.groups=all,blade,mm09,rack03
```

We have chosen not to run the actual commands, rather to print them on stdout. After you consider the commands correct, uncomment the following line and run the script again to populate the database.

```
#       system("$cmd")
```

### Fill in the mp table

You can use the regular expression shown in Example 4-14 on page 86 to map regularly numbered node names to regularly numbered MM names. For a good description of how regular expressions are used, see the database overview:

http://xcat.sourceforge.net/man5/xcatdb.5.html

*Example 4-14 Populating the mp table*

```
# chtab node=blade \
mp.mpa="|\D+(\d+)|bca0((\$1/16)+1)|",mp.id="|\D+(\d+)|(\$1%16)|";
This will cause blade1-blade16 to map to MM bca01 ($1\16+1) blade17-32
to map to bca02, and so on. It will also cause blade1, blade17, balde33
and so on to map to id 1 ($1%16) - the % sign is modulus (remainder).
You can check this mapping yourself by doing:
# nodels blade17 mp.mpa
blade18: bca2
# nodels blade17 mp.id
blade18: 1
```

### Fill in the nodehm table

You can use the *blade* group name to assign the mgt field of the nodehm table. The mgt field specifies the hardware management method and should be set to blade for blades:

```
# chtab node=blade nodehm.mgt=blade
```

## 4.1.12 Setting up network name resolution

Specify DNS information, and populate the /etc/hosts files and start DNS.

### Specify Domain Name System (DNS) information

For DNS, fill in the `nameservers` and `domain` fields in the site table. Some fields in the site table have been filled in automatically at installation, as shown in Example 4-15.

*Example 4-15 Default data in nodehm table*

```
# tabdump site
#key,value,comments,disable
"xcatdport","3001",,
"xcatiport","3002",,
"tftpdir","/tftpboot",,
"master","192.168.101.169",,
"domain",,,
```

```
"installdir","/install",,
"timezone","America/New_York",,
"nameservers",,,
```

Use the **tabedit** or **chtab** command to add the nameservers and domain entries, substituting your management node's IP address and your domain name:

```
# chtab key=nameservers site.value=192.168.101.166
# chtab key=domain site.value=itso.ibm.com
```

If you have additional site name servers, add their comma-separated IP addresses to the site table as forwarders, for example:

```
"forwarders","9.12.6.32,9.12.6.33"
```

## Populate the /etc/hosts file and start DNS

Modify the first line of /etc/hosts to look like the following example:

```
127.0.0.1       localhost.localdomain   localhost
```

Make sure there are entries for all of your blades and MMs as known by xCAT (**nodels** command) in /etc/hosts.

Run the **makedns** command:

```
# makedns
```

Create an /etc/resolv.conf file with your domain name and management node IP address:

```
search itso.ibm.com
nameserver 192.168.100.166
```

Start DNS:

```
# service named start
Starting named:                                            [  OK  ]
# chkconfig --level 345 named on
```

Verify DNS operation with one of your blade's host names, for example:

```
# nslookup b03n29
Server:         192.168.100.166
Address:        192.168.100.166#53

Name:   b03n29.itso.ibm.com
Address: 192.168.101.169
```

## 4.1.13 Configuring remote power control

Ensure that you have configured the network settings on the management modules, as described in 4.1.8, “Configuring the management module network settings” on page 82.

Next, check that the **rpower** command is operational (see Example 4-16).

*Example 4-16 Checking node power status*

```
# rpower blade stat
SNYK108069H115: on
SNYK308063M103: on
SNZJ1TS741S1AP: on
SNZJ1TS741Y1MZ: on
SNZJ1V5T43W130: on
SNZJ1V5T44415M: on
SNZJ1V5T44512L: on
SNZJ1YEY51H18H: on
SNZJ1YEY51H18S: on
b03n29: on
b03n30: on
b03n31: on
b03n32: on
```

## 4.1.14 Configuring the remote console

At the time of this writing, setting the serialspeed field of the nodehm table caused the kernel to fail to boot during installation. Leave the serialspeed field blank. However, you do have to fill in the cons field. For Power Architecture blades, it must be set to blade. Check it with **tabdump nodehm** command. You can make one entry for the entire lpar group:

```
# chtab node=blade nodehm.cons=hmc
```

Run the **conserver** configuration command as shown in Example 4-17.

*Example 4-17 Configuring conserver*

```
# service conserver stop
Shutting down conserver:                                   [  OK  ]
# makeconservercf
# service conserver start
Starting conserver: [Wed May 28 22:16:48 2008] conserver (5257): conserver.com
version 8.1.16
[Wed May 28 22:16:48 2008] conserver (5257): started as`root' by `root'
[Wed May 28 22:16:48 2008] conserver (5257): daemonizing   [  OK  ]
```

This command adds entries to the /etc/conserver.cf file for each of the nodes. See Example 4-18 on page 89.

*Example 4-18 Checking conserver connectivity*

```
#xCAT BEGIN p6p5 CONS
console p6p5 {
    type exec;
    exec /opt/xcat/share/xcat/cons/hmc p6p5;
}
#xCAT END p6p5 CONS
If a node is already installed, you can test rcons functionality now.
Note that you will need to hit enter to get a login prompt.
# rcons b03n31
[Enter `^Ec?' for help]

Red Hat Enterprise Linux Server release 5.1 (Tikanga)
Kernel 2.6.18-53.el5 on an ppc64
js21_06 login:
```

### 4.1.15 Creating the Red Hat installation source

Use the **copycds** command to copy the distribution source to the xCAT install directory. This is the directory specified by the `installdir` field in the site table. It is set to `/install` by default. In this example, we assume you have a copy of the DVD image (.iso) in /tmp directory. In the following example, the files are copied to /install/rhels5.1/ppc64:

```
# copycds /tmp/RHEL5.1-Server-20071017.0-ppc-DVD.iso
Copying media to /install/rhels5.1/ppc64/
Media copy operation successful
```

If you plan to use multiple distributions, repeat this step for each distribution.

### 4.1.16 Specifying other installation-related information

This section describes how to enter additional installation information, such as node characteristics, root password, and resources to the xCAT database.

### Specify node characteristics

Enter operating system, architecture, profile, and node type values into the nodetype table. Add the following line to the nodetype table. For example, add the following line to the nodetype table by using the **tabedit nodetype** command:

```
# tabedit nodetype "blade","rhels5.1","ppc64","compute","blade"
```

Setting the node value (column 1) to `blade` causes this line to apply to all the nodes in the group `blade`. This example is for a homogeneous cluster containing all Power Architecture blades (ppc64) to be installed with RHEL Server 5.1 (rhels5.1) using the compute kickstart template. You can create different entries for different distributions and nodes as needed.

### Specify installation root password

Add the following line to the passwd table to set the root password to `cluster` on the nodes at installation time:

```
"system","root","cluster"
```

### Specify node installation resources (noderes table)

Run the following command to add the installation resources to the noderes table. For a Power Architecture blade, `installnic` must be `eth1`. Replace the `noderes.nfsserver` IP address with the one belonging to your management node.

```
# chtab node=blade noderes.netboot=yaboot noderes.installnic=eth1
noderes.primarynic=eth0 noderes.nfsserver=192.168.101.169
```

## 4.1.17 Obtaining node MAC addresses

Use the **getmacs** command to obtain each node's Ethernet MAC address, as shown in Example 4-19.

**Note:** To acquire the MAC addresses, the nodes are rebooted.

At the time of this writing, no mechanism is available to write the eth1 addresses directly into the database. Usually the MAC address of eth1 differs only in the last digit, which is one more than that for eth0. After you verify this, you can write the eth0 MAC address to the database and then modify it manually.

*Example 4-19 Obtaining nodes' eth0 MAC addresses*

```
# getmacs blade
SNYK108069H115: mac.mac set to 00:14:5E:D5:25:16
SNYK308063M103: mac.mac set to 00:14:5E:1D:21:16
```

```
SNZJ1TS741S1AP: mac.mac set to 00:0D:60:4E:6D:A8
SNZJ1TS741Y1MZ: mac.mac set to 00:0D:60:4E:83:4A
SNZJ1V5T43W130: mac.mac set to 00:0D:60:9C:21:DE
SNZJ1V5T44512L: mac.mac set to 00:0D:60:9C:3A:E4
SNZJ1YEY51H18H: mac.mac set to 00:0D:60:1E:CA:00
SNZJ1YEY51H18S: mac.mac set to 00:0D:60:1E:BF:84
b03n29: mac.mac set to 00:11:25:C9:11:38
b03n30: mac.mac set to 00:11:25:C9:11:AE
b03n31: mac.mac set to 00:11:25:C9:0D:D0
b03n32: mac.mac set to 00:11:25:C9:11:BE
```

## 4.1.18 Setting up DHCP

If the bind-chroot rpm is installed, remove it because it interferes with the configuration for xCAT's DNS.

### Add entries to networks table manually, if necessary

The networks table is partially filled in at installation. It looks similar to Example 4-20.

*Example 4-20 Networks table*

```
# tabdump networks
#netname,net,mask,mgtifname,gateway,dhcpserver,tftpserver,nameservers,d
ynamicrange,nodehostname,comments,disable
,"192.168.100.0","255.255.255.0","eth0",,,"192.168.100.169",,,,,
,"192.168.101.0","255.255.255.0","eth1",,,"192.168.101.169",,,,,
```

For Power Architecture blade installation, you must use eth0 for Serial Over Lan (SOL) and eth1 for DHCP. If they were not both active when xCAT was installed, you might have to add them manually.

For eth1, assign to the dhcpserver field the IP address of the management node, and specify a dynamic node range for DHCP to use during the initial network boot. Although you should have at least as many dynamic addresses as there are nodes, make sure none of them is being used. Use either **tabedit** command, or the following **chtab** command:

```
# chtab mgtifname=eth1 networks.dhcpserver=192.168.101.169
# networks.dynamicrange=192.168.101.100-192.168.101.150
```

### Stop, configure, and restart dhcpd

To stop, configure, and restart dhcpd:

```
# service dhcpd stop
Shutting down dhcpd:                                       [  OK  ]
# makedhcp -n
# service dhcpd start
Starting dhcpd:                                            [  OK  ]
```

## 4.1.19 Initiating network installation

Set the blades to boot from the network, condition the nodes, and install the remaining blades.

### Set the blade(s) to boot from the network.

We recommend you install one blade before you attempt multiple blades. First, set the bootlist of the blade to boot from the network.

To show the current setting, run:

```
# rbootseq b03n31 list
b03n31: hd0,none,none,none
```

The first two entries in the list must be `net` and a disk, such as `hd0`, for installation to work properly:

```
# rbootseq b03n31 net,hd0
b03n31: net,hd0,none,none
```

When the node reboots after the installation, an entry is automatically added as the last line of the stanza in the dhcpd.leases file to cause the node to fail the netboot so it boots from disk. This line is shown in Example 4-21.

*Example 4-21 dhcpd.leases entry to disable network boot*

```
    supersede server.filename =
"xcat/nonexistant_file_to_intentionally_break_netboot_for_localboot_to_work";
```

Query xCAT's yaboot package:

```
# rpm -q yaboot-xcat
yaboot-xcat-1.3.14-2
```

If not installed, install it from /root/xcat2/dep-snap:

```
# rpm -i /root/xcat2/dep-snap/yaboot-xcat-1.3.14-2.noarch.rpm
```

## Start rinstall to install the blade (condition the nodes)

Use the **rinstall** command to condition the nodes to be installed:

```
# rinstall b03n29
b03n29: install rhels5.1-ppc64-compute
b03n29: on reset
```

You may open a remote console to the node and watch its installation progress:

```
# rcons b03n29
```

An entry is added to /var/lib/dhcdp/dhcpd.leases, similar to the one in Example 4-22.

*Example 4-22 dhcpd.leases configured for node boot*

```
host b03n29 {
    dynamic;
    hardware ethernet 00:11:25:c9:11:39;
    fixed-address 192.168.101.169;
        supersede server.next-server = c0:a8:65:a6;
}
```

There should also be a file in /tftpboot/etc that matches the hex number on the last line of the stanza in the example: c0a865a6. This is a HEX representation of node's IP address. The matching file is shown in Example 4-23.

*Example 4-23 Node boot config file*

```
#install rhels5.1-ppc64-compute
timeout=5
image=xcat/rhels5.1/ppc64/vmlinuz
        label=xcat
        initrd=xcat/rhels5.1/ppc64/initrd.img
        append="nofb utf8 ks=http://192.168.101.166/install/autoinst/b03n29
ksdevice=eth1 noipv6"
```

The image (kernel) file and the ramdisk (initrd) are relative to /tftpboot. The b03n29 file referred to in the last line is the kickstart template file.

## Install the remaining blades

You may install multiple blades in parallel by using xCAT's noderange capabilities. For example, to install the remaining blades of our example, you could use following command:

```
#rinstall blade,-b03n29
```

This installs all of the nodes in group blade, except b03n29.

## 4.2 Installing xCAT 2 on Intel blades

In this section we describe the following information:

- Setting up the management node
- Configuring the management node
- Adding nodes
- Installing nodes

Figure 4-2 shows the logical cluster diagram of our example cluster.

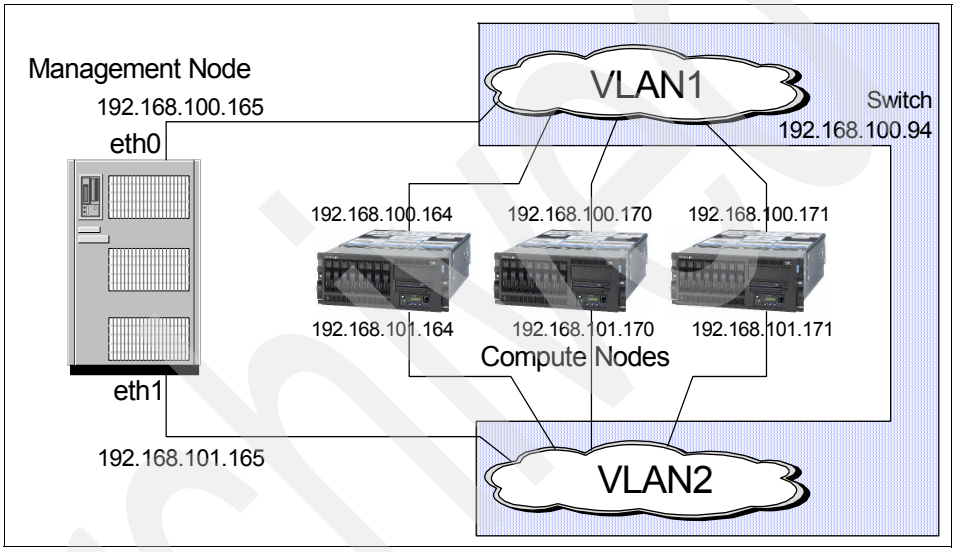


*Figure 4-2 Intel cluster diagram*

The management node requirements are:

- For Linux on Intel (x86, both 32-bit and 64-bit), ensure that you are running Red Hat Enterprise Linux Server 5.1 (RHEL Server 5.1.
- Create the root (/) partition with enough space on the management server to hold xCAT 2 packages and Linux distributions. The size of the partition depends on how you plan to use the cluster and the software it will contain.

### 4.2.1 Setting up the management server

This section provides instructions for installing and configuring xCAT 2 management server using Red Hat Enterprise Linux Server 5.1 on Intel platform (x86, both 32-bit and 64-bit).

Before you install the management server ensure that:

- The Linux operating system (we used RHEL Server 5.1) with all packages are installed on the machine that you plan as your management server.
- The networks are set up correctly.
- As with any clustering product, IP name resolution is critical. Make sure that you have set up proper name resolution, following your location convention.
- You use consistent IP naming for all nodes that will be participating in your cluster. The names must consistently use the short or fully qualified domain name (FQDN).

### 4.2.2 Downloading xCAT 2 packages

Because xCAT software is Open Source (Eclipse GPL License), no media comes with the product. Everything must be downloaded from the xCAT Web site, as follows:

1. Go to xCAT Web page:

   http://xcat.sourceforge.net/yum/

2. Download the xcat-core and xcat-dep tarballs:

   core-rpms-snap.tar.bz2
   dep-rpms-snap.tar.bz2

3. Untar the tarballs to a temporary directory (for example /tmp/xcat2) as shown in Example 4-24.

*Example 4-24 Extracting XCAT2 packages*

```
# mkdir /tmp/xcat2
# cd /tmp/xcat2
[root@hs20b05 xcat2]# tar -jxvf dep-rpms-snap.tar.bz2
[root@hs20b05 xcat2]# tar -jxvf core-rpms-snap.tar.bz2
```

### 4.2.3 Setting up the yum repository

Yellow dog Updater, Modified (yum) is a package manager that helps you create a package repository for your Linux system in a consistent and automated manner. It provides a set of utilities (in a command line interface) you use to install, update, and maintain your Linux system.

To set up the yum repository, go to the /root/xcat2/core-snap and to /etc/xcat2/dep-snap/rh5/x86 directories respectively, and run the local mklocalrepo.sh file in each of these two directories (see Example 4-25).

*Example 4-25 Setting up yum repository*

```
[root@hs20b05 xcat2]# cd dep-snap
[root@hs20b05 rh5]# cd x86
[root@hs20b05 x86]# ./mklocalrepo.sh
/etc/xcat2/dep-snap/rh5/x86
[root@hs20b05 x86]# cd ../../../core-snap/
root@hs20b05 core-snap]# ./mklocalrepo.sh
/etc/xcat2/core-snap
```

### 4.2.4 Removing the tftp-server and OpenIPMI-tools packages

To avoid the conflict with xCAT 2 requirements, you must remove the existing tftp-server and OpenIPMI-tools packages from the previously created yum repository, as shown in Example 4-26.

*Example 4-26 Removing tftp-server and OpenIPMI-tools packages*

```
[root@hs20b05 core-snap]# rpm -qa | grep tftp-server
tftp-server-0.42-3.1
[root@hs20b05 core-snap]# rpm -qa | grep OpenIPMI-tools
OpenIPMI-tools-2.0.6-5.el5.4
root@hs20_02 core-snap]# rpm -e tftp-server; rpm -e OpenIPMI-tools
```

### 4.2.5 Installing xCAT 2 packages

Start the yum installation as shown in Example 4-27.

**Note:** If the yum installation fails for any dependencies or conflicts, rectify the issue either by installing dependent rpms or removing the conflict rpms and then re-running the yum installation command until it completes the installation successfully.

*Example 4-27 Installing XCAT2*

```
[root@hs20b05 core-snap]# yum install xCAT.i386
Loading "changelog" plugin
Loading "protectbase" plugin
Loading "kmod" plugin
Loading "installonlyn" plugin
Loading "skip-broken" plugin
Loading "security" plugin
Loading "rhnplugin" plugin
Loading "downloadonly" plugin
This system is not registered with RHN.
```

```
RHN support will be disabled.
Setting up Install Process
Setting up repositories

cat-core-snap            100% |=========================|  951 B    00:00

cat-dep-snap             100% |=========================|  951 B    00:00

ocal-rhels5-x86-ClusterS 100% |=========================| 1.1 kB    00:00

ocal-rhels5-x86-VT       100% |=========================| 1.1 kB    00:00

ocal-rhels5-x86-Server   100% |=========================| 1.1 kB    00:00

ocal-rhels5-x86-Cluster  100% |=========================| 1.1 kB    00:00
Reading repository metadata in from local files

rimary.xml.gz            100% |=========================| 4.3 kB    00:00

####                                                   1/10
##########                                            2/10
###############                                       3/10
####################                                  4/10
########################                              5/10
############################                          6/10
################################                      7/10
####################################                  8/10
########################################            9/10
############################################### 10/10

rimary.xml.gz            100% |=========================| 8.4 kB    00:00

##                                                   1/16
######                                                2/16
#########                                             3/16
############                                          4/16
###############                                       5/16
##################                                    6/16
####################                                  7/16
########################                              8/16
############################                          9/16
################################                      10/16
####################################                  11/16
########################################            12/16
############################################        13/16
################################################    14/16
################################################    15/16
############################################### 16/16
0 packages excluded due to repository protections
```

```
Parsing package install arguments
Resolving Dependencies
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for xCAT to pack into transaction set.
CAT-2.0-snap200805151705 100% |=========================|  15 kB    00:00
---> Package xCAT.i386 0:2.0-snap200805151705 set to be updated
--> Running transaction check
--> Processing Dependency: atftp for package: xCAT
--> Processing Dependency: xCAT-nbroot-core-x86 for package: xCAT
--> Processing Dependency: xCAT-nbkernel-x86_64 for package: xCAT
--> Processing Dependency: conserver for package: xCAT
--> Processing Dependency: xCAT-nbroot-oss-x86_64 for package: xCAT
--> Processing Dependency: xCAT-client for package: xCAT
--> Processing Dependency: fping for package: xCAT
--> Processing Dependency: xCAT-nbroot-core-x86_64 for package: xCAT
--> Processing Dependency: xCAT-nbroot-oss-x86 for package: xCAT
--> Processing Dependency: ipmitool >= 1.8.9 for package: xCAT
--> Processing Dependency: xCAT-server for package: xCAT
--> Processing Dependency: xCAT-nbkernel-x86 for package: xCAT
--> Processing Dependency: perl-DBD-SQLite for package: xCAT
--> Restarting Dependency Resolution with new changes.
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for atftp to pack into transaction set.
tftp-0.7-4.i386.rpm      100% |=========================| 3.2 kB    00:00
---> Package atftp.i386 0:0.7-4 set to be updated
---> Downloading header for fping to pack into transaction set.
ping-2.4b2_to-2.i386.rpm 100% |=========================| 1.8 kB    00:00
---> Package fping.i386 0:2.4b2_to-2 set to be updated
---> Downloading header for xCAT-nbroot-core-x86 to pack into transaction set.
CAT-nbroot-core-x86-2.0- 100% |=========================| 5.5 kB    00:00
---> Package xCAT-nbroot-core-x86.noarch 0:2.0-snap200804251001 set to be updated
---> Downloading header for xCAT-nbkernel-x86_64 to pack into transaction set.
CAT-nbkernel-x86_64-2.6. 100% |=========================|  30 kB    00:00
---> Package xCAT-nbkernel-x86_64.noarch 1:2.6.18_53-2 set to be updated
---> Downloading header for xCAT-nbroot-oss-x86 to pack into transaction set.
CAT-nbroot-oss-x86-2.0-s 100% |=========================|  30 kB    00:00
---> Package xCAT-nbroot-oss-x86.noarch 0:2.0-snap200804021050 set to be updated
---> Downloading header for xCAT-nbkernel-x86 to pack into transaction set.
CAT-nbkernel-x86-2.6.18_ 100% |=========================|  29 kB    00:00
---> Package xCAT-nbkernel-x86.noarch 1:2.6.18_53-2 set to be updated
---> Downloading header for conserver to pack into transaction set.
onserver-8.1.16-5.i386.r 100% |=========================| 7.3 kB    00:00
---> Package conserver.i386 0:8.1.16-5 set to be updated
---> Downloading header for xCAT-nbroot-oss-x86_64 to pack into transaction set.
CAT-nbroot-oss-x86_64-2. 100% |=========================|  30 kB    00:00
---> Package xCAT-nbroot-oss-x86_64.noarch 0:2.0-snap200801291344 set to be updated
---> Downloading header for xCAT-client to pack into transaction set.
CAT-client-2.0-snap20080 100% |=========================|  34 kB    00:00
---> Package xCAT-client.noarch 0:2.0-snap200805191530 set to be updated
```

```
---> Downloading header for xCAT-server to pack into transaction set.
CAT-server-2.0-snap20080 100% |=========================|  32 kB    00:00
---> Package xCAT-server.noarch 0:2.0-snap200805191531 set to be updated
---> Downloading header for ipmitool to pack into transaction set.
pmitool-1.8.9-3.i386.rpm 100% |=========================|  12 kB    00:00
---> Package ipmitool.i386 0:1.8.9-3 set to be updated
---> Downloading header for xCAT-nbroot-core-x86_64 to pack into transaction set.
CAT-nbroot-core-x86_64-2 100% |=========================| 5.6 kB    00:00
---> Package xCAT-nbroot-core-x86_64.noarch 0:2.0-snap200804251001 set to be updated
---> Downloading header for perl-DBD-SQLite to pack into transaction set.
erl-DBD-SQLite-1.14-1.i3 100% |=========================| 4.5 kB    00:00
---> Package perl-DBD-SQLite.i386 0:1.14-1 set to be updated
--> Running transaction check
--> Processing Dependency: perl(xCAT::data::ipmisensorevents) for package: xCAT-server
--> Processing Dependency: perl(xCAT::PPCdb) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Utils) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Utils) for package: xCAT-client
--> Processing Dependency: perl(xCAT::Client) for package: xCAT-client
--> Processing Dependency: perl(xCAT::Schema) for package: xCAT-server
--> Processing Dependency: perl(xCAT::NodeRange) for package: xCAT-server
--> Processing Dependency: perl(xCAT::data::ibmleds) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Table) for package: xCAT-server
--> Processing Dependency: perl(xCAT::NotifHandler) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Template) for package: xCAT-server
--> Processing Dependency: perl-xCAT = 2.0 for package: xCAT-server
--> Processing Dependency: perl(xCAT::Yum) for package: xCAT-server
--> Processing Dependency: perl(xCAT::DSHCLI) for package: xCAT-client
--> Processing Dependency: perl(xCAT::MsgUtils) for package: xCAT-client
--> Processing Dependency: perl(xCAT::PPC) for package: xCAT-server
--> Processing Dependency: perl(xCAT::MsgUtils) for package: xCAT-server
--> Processing Dependency: perl(xCAT::DSHCLI) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Scope) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Postage) for package: xCAT-server
--> Processing Dependency: perl(xCAT::MacMap) for package: xCAT-server
--> Processing Dependency: perl(xCAT::GlobalDef) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Usage) for package: xCAT-server
--> Processing Dependency: perl(xCAT::PPCcli) for package: xCAT-server
--> Processing Dependency: perl(xCAT::DBobjUtils) for package: xCAT-server
--> Processing Dependency: perl(xCAT::Client) for package: xCAT-server
--> Processing Dependency: perl(xCAT::data::ipmigenericevents) for package: xCAT-server
--> Restarting Dependency Resolution with new changes.
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for perl-xCAT to pack into transaction set.
erl-xCAT-2.0-snap2008051 100% |=========================|  20 kB    00:00
---> Package perl-xCAT.noarch 0:2.0-snap200805191315 set to be updated
--> Running transaction check
--> Processing Dependency: perl(Expect) for package: perl-xCAT
--> Restarting Dependency Resolution with new changes.
--> Populating transaction set with selected packages. Please wait.
```

```
---> Downloading header for perl-Expect to pack into transaction set.
erl-Expect-1.21-1.noarch 100% |=========================| 4.3 kB    00:00
---> Package perl-Expect.noarch 0:1.21-1 set to be updated
--> Running transaction check
--> Processing Dependency: perl(IO::Tty) for package: perl-Expect
--> Processing Dependency: perl(IO::Pty) >= 1.03 for package: perl-Expect
--> Restarting Dependency Resolution with new changes.
--> Populating transaction set with selected packages. Please wait.
---> Downloading header for perl-IO-Tty to pack into transaction set.
erl-IO-Tty-1.07-1.i386.r 100% |=========================| 5.6 kB    00:00
---> Package perl-IO-Tty.i386 0:1.07-1 set to be updated
--> Running transaction check
Dependencies Resolved
================================================================================
 Package                 Arch       Version          Repository       Size
================================================================================
Installing:
 xCAT                    i386       2.0-snap200805151705  xcat-core-snap     32 k
Installing for dependencies:
 atftp                   i386       0.7-4            xcat-dep-snap      34 k
 conserver               i386       8.1.16-5         xcat-dep-snap     196 k
 fping                   i386       2.4b2_to-2       xcat-dep-snap      18 k
 ipmitool                i386       1.8.9-3          xcat-dep-snap     299 k
 perl-DBD-SQLite         i386       1.14-1           xcat-dep-snap     290 k
 perl-Expect             noarch     1.21-1           xcat-dep-snap      72 k
 perl-IO-Tty             i386       1.07-1           xcat-dep-snap      63 k
 perl-xCAT               noarch     2.0-snap200805191315  xcat-core-snap    180 k
 xCAT-client             noarch     2.0-snap200805191530  xcat-core-snap    945 k
 xCAT-nbkernel-x86       noarch     1:2.6.18_53-2    xcat-dep-snap     6.0 M
 xCAT-nbkernel-x86_64    noarch     1:2.6.18_53-2    xcat-dep-snap     6.5 M
 xCAT-nbroot-core-x86    noarch     2.0-snap200804251001  xcat-core-snap     18 k
 xCAT-nbroot-core-x86_64  noarch     2.0-snap200804251001  xcat-core-snap     18 k
 xCAT-nbroot-oss-x86     noarch     2.0-snap200804021050  xcat-dep-snap     2.3 M
 xCAT-nbroot-oss-x86_64  noarch     2.0-snap200801291344  xcat-dep-snap     2.3 M
 xCAT-server             noarch     2.0-snap200805191531  xcat-core-snap    283 k
Transaction Summary
================================================================================
Install     17 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
Total download size: 20 M
Is this ok [y/N]: y
Downloading Packages:
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing: xCAT-nbroot-core-x86_64      ###################### [ 1/17]
  Installing: xCAT-nbroot-oss-x86_64       ###################### [ 2/17]
```

```
  Installing: xCAT-nbroot-oss-x86          ###################### [ 3/17]
  Installing: xCAT-nbroot-core-x86         ###################### [ 4/17]
  Installing: perl-DBD-SQLite              ###################### [ 5/17]
  Installing: ipmitool                     ###################### [ 6/17]
  Installing: perl-IO-Tty                  ###################### [ 7/17]
  Installing: perl-Expect                  ###################### [ 8/17]
  Installing: perl-xCAT                    ###################### [ 9/17]
  Installing: xCAT-client                  ###################### [10/17]
  Installing: xCAT-server                  ###################### [11/17]
  Installing: xCAT-nbkernel-x86_64         ###################### [12/17]
  Installing: xCAT-nbkernel-x86            ###################### [13/17]
  Installing: conserver                    ###################### [14/17]
  Installing: fping                        ###################### [15/17]
  Installing: atftp                        ###################### [16/17]
Stopping ATFTP Starting ATFTP [  OK  ]
  Installing: xCAT                         ###################### [17/17]
Generating SSH1 RSA Key...
Generating public/private rsa1 key pair.
Your identification has been saved in /install/postscripts/hostkeys/ssh_host_key.
Your public key has been saved in /install/postscripts/hostkeys/ssh_host_key.pub.
The key fingerprint is:
f6:d6:11:e2:4a:8b:6c:ed:e7:11:99:75:44:d8:01:ac
Generating SSH2 RSA Key...
Generating public/private rsa key pair.
Your identification has been saved in /install/postscripts/hostkeys/ssh_host_rsa_key.
Your public key has been saved in /install/postscripts/hostkeys/ssh_host_rsa_key.pub.
The key fingerprint is:
d1:7e:9b:eb:52:a8:1b:55:ed:3d:fb:fe:b3:0d:da:14
Generating SSH2 DSA Key...
Generating public/private dsa key pair.
Your identification has been saved in /install/postscripts/hostkeys/ssh_host_dsa_key.
Your public key has been saved in /install/postscripts/hostkeys/ssh_host_dsa_key.pub.
The key fingerprint is:
ff:e6:cb:9d:0f:db:e1:f2:21:aa:a6:41:e0:76:e2:29
Starting vsftpd for vsftpd: [  OK  ]
Shutting down NFS mountd: [FAILED]
Shutting down NFS daemon: [FAILED]
Shutting down NFS quotas: [FAILED]
Shutting down NFS services:  [FAILED]
Starting NFS services:  [  OK  ]
Starting NFS quotas: [  OK  ]
Starting NFS daemon: [  OK  ]
Starting NFS mountd: [  OK  ]
# NOTE use "-newkey rsa:2048" if running OpenSSL 0.9.8a or higher
Generating a 2048 bit RSA private key
..............................+++
....+++
writing new private key to 'private/ca-key.pem'
-----
```

```
yes: standard output: Broken pipe
yes: write error
Generating RSA private key, 2048 bit long modulus
..................+++
.........................................+++
e is 65537 (0x10001)

Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: May 23 20:33:34 2008 GMT
            Not After : May 18 20:33:34 2028 GMT
        Subject:
            commonName                = hs20b05.itso.ibm.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                50:37:F2:BC:AD:96:1A:50:6A:56:B6:2B:EA:0C:8C:A3:01:79:76:EB
            X509v3 Authority Key Identifier:
                keyid:28:09:0B:C2:0F:F1:DB:90:02:31:BA:8E:C1:8F:7E:A2:86:B4:4D:34
Certificate is to be certified until May 18 20:33:34 2028 GMT (7300 days)
Sign the certificate? [y/n]:
1 out of 1 certificate requests certified, commit? [y/n]Write out database with 1 new entries
Data Base Updated

yes: standard output: Broken pipe
yes: write error
Generating RSA private key, 2048 bit long modulus
.....................+++
.....+++
e is 65537 (0x10001)

Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 2 (0x2)
        Validity
            Not Before: May 23 20:33:35 2008 GMT
            Not After : May 18 20:33:35 2028 GMT
        Subject:
            commonName                = root
```

```
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                AA:BF:6F:A6:A0:FC:39:02:9F:29:F5:EB:12:F6:8F:D9:3D:D1:D2:B8
            X509v3 Authority Key Identifier:
                keyid:28:09:0B:C2:0F:F1:DB:90:02:31:BA:8E:C1:8F:7E:A2:86:B4:4D:34
Certificate is to be certified until May 18 20:33:35 2028 GMT (7300 days)
Sign the certificate? [y/n]:
1 out of 1 certificate requests certified, commit? [y/n]Write out database with 1 new entries
Data Base Updated
yes: standard output: Broken pipe
yes: write error
# xCAT settings
Shutting down kernel logger: [  OK  ]
Shutting down system logger: [  OK  ]
Starting system logger: [  OK  ]
Starting kernel logger: [  OK  ]
Starting xCATd Shutting down vsftpd: [  OK  ]
Starting vsftpd for vsftpd: [  OK  ]
[  OK  ]
Creating nbfs.x86.gz in /tftpboot/xcat
Creating nbfs.x86_64.gz in /tftpboot/xcat
ERROR/WARNING: communication with the xCAT server seems to have been ended prematurely
Stopping httpd: [FAILED]
Starting httpd: [  OK  ]
xCAT is now installed, it is recommended to tabedit networks and set a dynamic ip address range
on any networks where nodes are to be discovered
Then, run makedhcp -n to create a new dhcpd.configuration file, and /etc/init.d/dhcpd restart
Either examine sample configuration templates, or write your own, or specify a value per node
with nodeadd or tabedit.

Installed: xCAT.i386 0:2.0-snap200805151705
Dependency Installed: atftp.i386 0:0.7-4 conserver.i386 0:8.1.16-5 fping.i386 0:2.4b2_to-2
ipmitool.i386 0:1.8.9-3 perl-DBD-SQLite.i386 0:1.14-1 perl-Expect.noarch 0:1.21-1
perl-IO-Tty.i386 0:1.07-1 perl-xCAT.noarch 0:2.0-snap200805191315 xCAT-client.noarch
0:2.0-snap200805191530 xCAT-nbkernel-x86.noarch 1:2.6.18_53-2 xCAT-nbkernel-x86_64.noarch
1:2.6.18_53-2 xCAT-nbroot-core-x86.noarch 0:2.0-snap200804251001 xCAT-nbroot-core-x86_64.noarch
0:2.0-snap200804251001 xCAT-nbroot-oss-x86.noarch 0:2.0-snap200804021050
xCAT-nbroot-oss-x86_64.noarch 0:2.0-snap200801291344 xCAT-server.noarch 0:2.0-snap200805191531
Complete!
[root@hs20b05 core-snap]#
```

### 4.2.6 Setting up the root user profile

Go to the /etc/profile.d directory and run xcat.sh (see Example 4-28).

*Example 4-28 Set up profile*

```
[root@hs20b05 core-snap]# cd /etc/profile.d
[root@hs20b05 profile.d]# ./xcat.sh
```

This file adds /opt/xcat in the path and exports the XCATROOT variable.

**Note:** Log out and log on again to activate the xCAT 2 profile changes.

### 4.2.7 Verifying the installation

Check the xCAT 2 installation by running the **tabdump site** command and comparing it to the output shown in Example 4-29.

*Example 4-29 The site table*

```
[root@hs20b05 ~]# tabdump site
#key,value,comments,disable
"xcatdport","3001",,
"xcatiport","3002",,
"tftpdir","/tftpboot",,
"master","192.168.101.165",,
"domain","itso.ibm.com",,
"installdir","/install",,
"timezone","America/New_York",,
"nameservers","192.168.101.165",,
[root@hs20b05 ~]#
```

### 4.2.8 Disabling SELinux

During installation, if SELinux is enabled, it conflicts with xCAT programs. You must disable SELinux on the xCAT 2 management server. Disable it by editing /etc/selinux/config file, as shown in Example 4-30, then rebooting.

*Example 4-30 /etc/selinux/config*

```
vi /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
```

```
#       disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#       targeted - Only targeted network daemons are protected.
#       strict - Full SELinux protection.
SELINUXTYPE=targeted
```

**Note:** You might have to reboot the node if you modify /etc/selinux/config.

### 4.2.9 Copying the Linux distribution to the management node

Run the **copycds** command to copy the Linux distribution to the /install directory on your management server, as shown in Example 4-31. In our example, we stored the RHEL 5.1 Server image in the /tmp/xcat directory.

**Notes:** Make sure that you have enough space to store the Linux distribution.

At the time of this publication, the **copycds** command supports only ISO images. Therefore, keep the ISO image available before running the **copycds** command.

*Example 4-31 Copying Linux distribution for i386 architecture*

```
[root@hs20b05 xcat]# pwd
/tmp/xcat
[root@hs20b05 xcat]# ls
RHEL5.1-Server-20071017.0-i386-DVD.iso
[root@hs20b05 xcat]# copycds RHEL5.1-Server-20071017.0-i386-DVD.iso
Copying media to /install/rhels5/x86/
Media copy operation successful
[root@hs20b05 xcat]#
```

### 4.2.10 Restoring the predefined tables

The xCAT 2 configuration tables are stored in files in SQLite database format. They are available in /etc/xcat directory and you can edit them by using the **tabedit** command. For an overview of the xCAT architecture, see Chapter 2, “xCAT 2 architecture” on page 11.

Restore the predefined tables from /opt/xcat/share/xcat/templates/e1350, as shown in Example 4-32 and edit them according to your cluster environment.

*Example 4-32 Restoring the predefined xcat table*

```
[root@hs20b05 ~]# cd /opt/xcat/share/xcat/templates/e1350
[root@hs20b05 e1350]# for i in *csv; do tabrestore $i; done
```

The tables we used in our test cluster are listed in Table 4-1.

*Table 4-1 Overview of xCAT 2 configuration tables*

| Tables | Descriptions |
|---|---|
| site | This is the main configuration file and it has the global settings for the whole cluster. |
| networks | This table defines the networks in the cluster and information necessary to set up nodes on that network. |
| nodelist | This table list of all nodes in the cluster. |
| noderes | This table defines the resources and settings to use when installing nodes. |
| nodetype | This table defines the hardware and software characteristics of the nodes. |
| nodehm | This table defines the hardware control of each nodes in the cluster. |
| mp | This table defines the hardware control details specific to blades. |
| mac | This table defines the MAC address of the nodes install adapter. |
| passwd | This table defines the username and password of the nodes. |

## 4.2.11 Configuring management node services

Start the configuration of the xCAT software stack on the management node. After finalizing the configuration of the management node services, proceed to installing the compute nodes.

### Set up the networks table

The networks table is used to store cluster networks information. This information includes name resolution (name server) and DHCP dynamic addresses range that will be used for the compute nodes.

Update your name server and dynamic range in the networks table by using the **chtab** command, as shown in Example 4-33.

*Example 4-33 Updating networks table*

```
[root@hs20b05 e1350]# chtab net=192.168.101.0
networks.nameservers=192.168.101.165
[root@hs20b05 e1350]# chtab net=192.168.101.0
networks.dynamicrange=192.168.101.180-192.168.101.185
[root@hs20b05 e1350]#
```

Check the networks table using the **tabdump** command to verify it has been updated with the correct attributes of your cluster, as shown in Example 4-34.

*Example 4-34 Checking networks table*

```
[root@hs20b05 ~]# tabdump networks
#netname,net,mask,mgtifname,gateway,dhcpserver,tftpserver,nameservers,d
ynamicrange,nodehostname,comments,disable
,"192.168.101.0","255.255.255.0","eth1",,,"192.168.101.165","192.168.10
1.165","192.168.101.180-192.168.101.185",,,
,"192.168.100.0","255.255.255.0","eth0",,,"192.168.100.165","192.168.10
0.165","192.168.100.180-192.168.100.185",,,
[root@hs20b05 ~]#
```

## Update the /etc/hosts file

The /etc/hosts file can be used to generate the configuration files for the DNS server.You must list all IP labels (names) and IP addresses used in your cluster in /etc/hosts file. Example 4-35 shows our sample cluster hosts file.

*Example 4-35 /etc/hosts*

```
[root@hs20b05 e1350]# cat /etc/hosts
127.0.0.1       localhost.localdomain   localhost
192.168.101.165     hs20b05.itso.ibm.com        hs20b05 #xCAT MS
192.168.101.164     hs20b04.itso.ibm.com        hs20b04
192.168.101.171     hs21b11.itso.ibm.com        hs21b11
192.168.100.91      blademm.itso.ibm.com        blademm
192.168.100.94      bladesw01.itso.ibm.com      bladesw01
```

## Set up DNS

xCAT 2 generates the DNS configuration file automatically. You must specify the correct IP address of the management node and the name server in the site table as shown in Example 4-29 on page 104.

Update your /etc/resolv.conf file as shown in Example 4-36.

*Example 4-36 /etc/resolv.conf*

```
[root@hs20b05 ~]# cat /etc/resolv.conf
search itso.ibm.com
nameserver 127.0.0.1
[root@hs20b05 ~]#
```

Comment out the ROOTDIR variable in the /etc/sysconfig/named file, as shown in Example 4-37.

*Example 4-37 /etc/sysconfig/named*

```
[root@hs20b05 e1350]# cat /etc/sysconfig/named
# BIND named process options
# ~~~~~~~~~~~~~~~~~~~~~~~~~~
# Currently, you can use the following options:
#
# ROOTDIR="/some/where"  --  will run named in a chroot environment.
#                           you must set up the chroot environment
#                           (install the bind-chroot package) before
#                           doing this.
#
# OPTIONS="whatever"     --  These additional options will be passed to named
#                           at startup. Don't add -t here, use ROOTDIR instead.
#
# ENABLE_ZONE_WRITE=yes  --  If SELinux is disabled, then allow named to write
#                           its zone files and create files in its $ROOTDIR/var/named
#                           directory, necessary for DDNS and slave zone transfers.
#                           Slave zones should reside in the $ROOTDIR/var/named/slaves
#                           directory, in which case you would not need to enable zone
#                           writes. If SELinux is enabled, you must use only the
#                           'named_write_master_zones' variable to enable zone writes.
#
# ENABLE_SDB=yes         --  This enables use of 'named_sdb', which has support
#                        --  for the ldap, pgsql and dir zone database backends
#                        --  compiled in, to be used instead of named.
#
# DISABLE_NAMED_DBUS=[1y]--  If NetworkManager is enabled in any runlevel, then
#                           the initscript will by default enable named's D-BUS
#                           support with the named -D option. This setting disables
#                           this behavior.
#
# ROOTDIR=/var/named/chroot
```

Now, generate the DNS configuration file and start it by using the following commands:

```
makedns
service named start
```

After DNS is started, test that name resolution by using the **host** command, as shown in Example 4-38.

*Example 4-38 Testing DNS resolution*

```
[root@hs20b05 ~]# host hs20b04
hs20b04.itso.ibm.com has address 192.168.101.164
hs20b04.itso.ibm.com mail is handled by 10 hs20b04.itso.ibm.com.
[root@hs20b05 ~]#
```

**Note:** The **host** command in Linux references only the /etc/resolv.conf file (no flat files). If DNS is not configured properly, this command will time-out. The name service switch (NSS) method is not used by the **host** command.

## 4.2.12 Setting up DHCP

You have to define the DHCP server on your network. This must be accessible to the compute nodes either directly (the same subnet) or through a gateway (a different subnet).

Update your DHCP server by using the **chtab** command in networks table, as shown in Example 4-39:

*Example 4-39 Updating dhcp server in networks table*

```
[root@hs20b05 e1350]# chtab net=192.168.101.0
networks.dhcpserver=192.168.101.165
```

Create the initial dhcpd.conf configuration file by using the following command:

```
makedhcp -n
```

Verify the content of the dhcpd configuration file /etc/dhcpd.conf and remove any unnecessary entries. Our sample file is shown in Example 4-40.

*Example 4-40 /etc/dhcpd.conf*

```
root@hs20b05 ~]# cat /etc/dhcpd.conf
#xCAT generated dhcp configuration

authoritative;
```

```
ddns-update-style none;
option client-architecture code 93 = unsigned integer 16;

omapi-port 7911;
key xcat_key {
  algorithm hmac-md5;
  secret "MTB5OHdJVjlTYlpGOGp5TkR6bEOyeG1TMW9CSlJzNnc=";
};
omapi-key xcat_key;
shared-network eth1 {
  subnet 192.168.101.0 netmask 255.255.255.0 {
    max-lease-time 43200;
    min-lease-time 43200;
    default-lease-time 43200;
    next-server  192.168.101.165;
    option log-servers 192.168.101.165;
    option ntp-servers 192.168.101.165;
    option domain-name "itso.ibm.com";
    option domain-name-servers  192.168.101.165;
    if option client-architecture = 00:00  { #x86
      filename "pxelinux.0";
    } else if option client-architecture = 00:02 { #ia64
       filename "elilo.efi";
    } else if substring(filename,0,1) = null { #otherwise, provide yaboot if
the client isn't specific
       filename "/yaboot";
    }
    range dynamic-bootp 192.168.101.180 192.168.101.185;
  } # 192.168.101.0/255.255.255.0 subnet_end
} # eth1 nic_end
shared-network eth0 {
  subnet 192.168.100.0 netmask 255.255.255.0 {
    max-lease-time 43200;
    min-lease-time 43200;
    default-lease-time 43200;
    next-server  192.168.100.165;
    option log-servers 192.168.100.165;
    option ntp-servers 192.168.100.165;
    option domain-name "itso.ibm.com";
    option domain-name-servers  192.168.100.165;
    if option client-architecture = 00:00  { #x86
      filename "pxelinux.0";
    } else if option client-architecture = 00:02 { #ia64
       filename "elilo.efi";
    } else if substring(filename,0,1) = null { #otherwise, provide yaboot if
the client isn't specific
       filename "/yaboot";
    }
    range dynamic-bootp 192.168.100.180 192.168.100.185;
```

```
  } # 192.168.100.0/255.255.255.0 subnet_end
} # eth0 nic_end
[root@hs20b05 ~]#
```

Next, start the DHCP server by using the following command:

```
service dhcpd start
```

You can watch the /var/log/messages file to determine whether the DHCP daemon is running: Use the **tail -f ... /var/log/messages** command.

## 4.2.13 Setting up TFTP

The next step is to set up a Trivial File Transfer Protocol (TFTP) server that will deliver the boot file to the requesting compute node (or nodes). The default directory for serving TFTP files is /tftpboot.

Next, build the network layout for your architecture by using the **mknb** command, as shown in Example 4-41.

*Example 4-41 Running mknb*

```
[root@hs20b05 ~]# mknb x86
Creating nbfs.x86.gz in /tftpboot/xcat
[root@hs20b05 ~]#
```

Finally, restart the TFTP server, as shown in Example 4-42.

*Example 4-42 Restarting tftp service*

```
[root@hs20b05 ~]# service tftpd restart
Stopping ATFTP Starting ATFTP                              [  OK  ]
[root@hs20b05 ~]#
```

When you restart the tftpd service, you can watch the messages in the file /var/log/messages, as shown in Example 4-43.

*Example 4-43 tail -f /var/log/messages*

```
.....
Jun  2 11:18:41 hs20b05 atftpd[13539]: SIGTERM received, stopping threads and
exiting.
Jun  2 11:18:41 hs20b05 atftpd[13539]: SIGTERM received, stopping threads and
exiting.
Jun  2 11:18:41 hs20b05 atftpd[13539]: tftpd.c: 402: select: Interrupted system call
Jun  2 11:18:41 hs20b05 atftpd[13539]: tftpd.c: 402: select: Interrupted system call
```

```
Jun  2 11:18:41 hs20b05 atftpd[13539]: atftpd terminating
Jun  2 11:18:41 hs20b05 atftpd[13539]: atftpd terminating
Jun  2 11:18:41 hs20b05 atftpd[13539]: Main thread exiting
Jun  2 11:18:41 hs20b05 atftpd[13539]: Main thread exiting
Jun  2 11:18:41 hs20b05 atftpd[13571]: Advanced Trivial FTP server started (0.7)
Jun  2 11:18:41 hs20b05 atftpd[13571]: Advanced Trivial FTP server started (0.7)
Jun  2 11:18:41 hs20b05 atftpd[13571]: Build date: May 14 2008  00:44:04
Jun  2 11:18:41 hs20b05 atftpd[13571]: Build date: May 14 2008  00:44:04
......
```

To check the correct operation of the TFTP server, download the files using an tftp client, as shown in Example 4-44.

*Example 4-44 Testing tftp*

```
[root@hs20b05 ~]# pwd
/root
[root@hs20b05 ~]# echo "Hello" >/tftpboot/mytestfile
[root@hs20b05 ~]# tftp hs20b05
tftp> get mytestfile
tftp> quit
[root@hs20b05 ~]# cat mytestfile
Hello
[root@hs20b05 ~]#
```

## 4.2.14 Defining the BladeCenter management modules

You may add one or more management modules. To add the management module as a node, use the **nodeadd** command, as shown in Example 4-45.

*Example 4-45 Adding management module*

```
[root@hs20b05 ~]# nodeadd blademm groups=mm nodehm.mgt=blade
mp.mpa=192.168.100.91
```

Enable SNMP and SSH on the blade management module for remote power control, as shown in Example 4-46.

*Example 4-46 Enabling SNMP and SSH*

```
[root@hs20b05 ~]# ssh USERID@blademm users  -T mm[1] -1 -at set
Warning: Permanently added 'blademm,192.168.100.91' (RSA) to the list
of known hosts.
system> users -T mm[1] -1 -at set
OK
```

```
[root@hs20b05 ~]# rspconfig mm snmpcfg=enable sshcfg=enable
blademm: SNMP enable: OK
blademm: SSH enable: OK
[root@hs20b05 ~]# rspconfig blademm pd1=redwoperf pd2=redwoperf
blademm: pd2: redwoperf
blademm: pd1: redwoperf
[root@hs20b05 ~]# rpower blademm reset
[root@hs20b05 ~]#
```

### 4.2.15 Setting up conserver

Conserver (console server) is the open source program that manages remote console access to managed systems.

**Note:** For IBM BladeCenter servers, conserver supports SOL only through the BladeCenter Advanced Management Module.

To start the conserver:

1. Set up the user ID and password for management module as shown in Example 4-47. The default user ID and password for management module are USERID and PASSWORD (the 0 is a zero). You can edit the passwd table by using **tabedit** command.

   *Example 4-47 passwd table*

   ```
   [root@hs20b05 ~]# tabdump passwd
   #key,username,password,comments,disable
   "blade","USERID","PASSWORD",,
   "system","root","cluster",,
   "ipmi","USERID","PASSWORD",,
   "blademm","USERID","PASSWORD",,
   "omapi","xcat_key","MTB5OHdJVjlTYlpGOGp5TkR6bE0yeG1TMW9CSlJzNnc=",,
   ```

   **Note:** The default user ID and password of each node is root and cluster.

2. If it is not set up already, set up SSH key for the management module by using the **rspconfig** command, as shown in Example 4-48.

   *Example 4-48 setup ssh key for Management Module*

   ```
   [root@hs20b05 ~]# rspconfig blademm snmpcfg=enable sshcfg=enable
   blademm: SNMP enable: OK
   blademm: SSH enable: OK
   ```

```
[root@hs20b05 ~]#
```

3. Run the following commands, which are also shown in Example 4-49.

```
makeconservercf
service conserver start
```

*Example 4-49 Starting conserver*

```
[root@hs20b05 ~]# makeconservercf
[root@hs20b05 ~]# service conserver start
Starting conserver: [Mon Jun  2 11:33:23 2008] conserver (13772):
conserver.com version 8.1.16
[Mon Jun  2 11:33:23 2008] conserver (13772): started as `root' by `root'
[Mon Jun  2 11:33:23 2008] conserver (13772): daemonizing
                                                           [  OK  ]
```

4. Power on several nodes and determine if conserver is working by using the **rpower** and **rcons** commands.

## 4.2.16 Adding compute nodes

Before you add a node, you must collect the node attributes. Check the valid attributes by running the following command:

```
tabdump -d <table>
```

You should specify at least the attributes in nodehm, noderes, nodetype, and mp tables for a compute node installation.

The attributes we used for node installation are listed in Table 4-2.

*Table 4-2 Node attributes*

| Attributes | Descriptions |
|---|---|
| noderange | Specify the nodes that you want to add to the cluster. |
| groups | Specify the group name for the nodes. |
| mp.mpa | Specify the management module used to control the nodes. |
| mp.id | Specify the slot number of the blade. This attribute is node-specific. To obtain this value, run the **rscan** command. |
| nodehm.power | Specify the method to use to control the power of the node. |
| nodehm.mgt | Specify the method to use to do general hardware management of the node. |

| Attributes | Descriptions |
|---|---|
| nodetype.os | Specify the OS that you want to install on the node or nodes. |
| nodetype.arch | Specify the architecture of the node or nodes. |
| nodetype.profile | Specify the kickstart or autoyast template name. |
| nodetype.nodetype | Specify the characteristics of the node or nodes. |
| noderes.nfsserver | Specify the NFS server for the node or nodes. |
| noderes.netboot | Specify the network booting method. Supported methods are pxe and yaboot. |
| noderes.primarynic | Specify the network adapter of the management node to use for node installation. |

After you have collected attributes for the node or nodes, you can add the nodes by running the **nodeadd** command, as shown in Example 4-50.

*Example 4-50 Adding a node*

```
root@hs20b05 ~]# nodeadd hs20b04 groups=blade,compute mp.mpa=192.168.100.91
nodehm.power=blade nodehm.mgt=blade nodetype.os=rhels5 nodetype.arch=x86 \
nodetype.profile=compute nodetype.nodetype=osi noderes.nfsserver=hs20b05 \
noderes.netboot=pxe noderes.primarynic=eth1
```

You may add the node or nodes without the `mp.id` value the first time; later, you may update them in the mp table by using either the **nodech** or **rscan** command.

The mp.id value is the slot number of the blade. The following example, gets the value by using the **rscan** command, as shown in Example 4-51.

*Example 4-51 Using rscan to obtain the value*

```
[root@hs20b05 install]# rscan blademm
type    name             id      type-model  serial-
mm      SN#0J1U9E5841AA  0       8677-3XU    KPVH850
blade   SN#ZJ1V5T44415M  1       8832-XX3    6A56860
blade   hs20b02          2       8832-XX3    6A51230
blade   SN#ZJ1TS741S1AP  3       8832-XX3    6A57961
blade   hs20b04          4       8832-XX3    6A52352
blade   hs20b05          5       8832-XX3    6A57946
blade   b03n31           6       8844-51U    KQ0107A
blade   b03n30           7       8844-51X    23A0393
blade   b03n32           8       8844-51X    23A0409
blade   b03n29           9       8844-51X    KQ0109F
blade   hs21b10          10      8853-R0Z    23A0510
```

```
blade   hs21b11          11      8853-DWZ    23A0368
blade   SN#ZJ1YEY51H18H  13      8842-21X    KPFYG4D
blade   SN#ZJ1YEY51H18S  14      8842-21X    KPFYG9T
```

We updated the mp.id value by using the **nodech** command, as shown in Example 4-52.

*Example 4-52 Updating mp.id in mp table*

```
[root@hs20b05 ~]# nodech hs20b04 mp.id=4
```

You may check the compute nodes by running the **nodels** command, as shown in Example 4-53.

*Example 4-53 Listing nodes*

```
[root@hs20b05 ~]# nodels
blademm
hs20b04
```

Test the management module by powering on and off the node, as shown in Example 4-54.

*Example 4-54 Testing management module by rpower*

```
[root@hs20b05 ~]# rpower hs20b04 stat
hs20b04: off
[root@hs20b05 ~]# rpower hs20b04 on
hs20b04: on
[root@hs20b05 ~]# rpower hs20b04 stat
hs20b04: on
```

Update the `remoteshell` attribute in the postscripts table, as shown in Example 4-55, so that you can log in to the node without a password.

*Example 4-55 The postscripts table*

```
[root@xcat2mgmt scripts]# tabdump postscripts
#node,postscripts,comments,disable
"hs20b04","remoteshell",,
"hs21b10","remoteshell",,
"hs21b11","remoteshell",,
[root@xcat2mgmt scripts]#
```

## 4.2.17 Installing compute nodes

Before you proceed to compute node installation, check the boot sequence and the installation template, which you might want to alter.

### Select the network boot

Make sure that you have selected the network boot first. Otherwise, change the boot sequence by using **rbootseq** command, as shown Example 4-56.

*Example 4-56 Checking the boot sequence*

```
[root@hs20b05 ~]#  rbootseq hs20b04 list
hs20b04: cdrom,hd0,floppy,net
[root@hs20b05 ~]#  rbootseq hs20b04 n,h,c,f
hs20b04: net,hd0,cdrom,floppy
[root@hs20b05 ~]#
```

### Prepare the template

Create the kickstart file for node installation. Run the **nodeset** command to prepare the template file, as shown in Example 4-57.

*Example 4-57 Create the kickstart file*

```
[root@hs20b05 ~]# nodeset hs20b04 install
hs20b04: install rhels5-x86-compute
```

**Note:** You can see the kickstart installation files in /install/autoinst and you can modify the files according to your cluster requirements.

### Start the installation.

To start the installation, use the **rinstall** command, as shown in Example 4-58.

*Example 4-58 start the installation*

```
[root@hs20b05 ~]# rinstall hs20b04
hs20b04: install rhels5-x86-compute
hs20b04: on reset
```

**Note:** The **rinstall** command edits the kickstart file and then boots the node for installation. If you want to use your customized kickstart file, boot the node using the **rpower** command after running **nodeset** command.

## Monitor the installation

You can view the node's installation status by running the **nodestat** command, as shown in Example 4-59.

*Example 4-59 Checking the node status*

```
[root@hs20b05 ~]# nodestat hs20b04
hs20b04: installing prep
```

Also check the /var/log/messages, as shown in Example 4-60, and see whether tftp is transferring the image to node.

*Example 4-60 /var/log/messages*

```
[root@hs20b05 ~]tail -f /var/log/messages
.......
May 31 14:35:39 hs20b05 dhcpd: DHCPDISCOVER from 00:0d:60:9c:21:df via eth1
May 31 14:35:39 hs20b05 dhcpd: DHCPDISCOVER from 00:0d:60:9c:21:df via eth1
May 31 14:35:39 hs20b05 dhcpd: DHCPOFFER on 192.168.101.164 to
00:0d:60:9c:21:df via eth1
May 31 14:35:39 hs20b05 dhcpd: DHCPOFFER on 192.168.101.164 to
00:0d:60:9c:21:df via eth1
May 31 14:35:47 hs20b05 dhcpd: DHCPREQUEST for 192.168.101.164
(192.168.101.165) from 00:0d:60:9c:21:df via eth1
May 31 14:35:47 hs20b05 dhcpd: DHCPREQUEST for 192.168.101.164
(192.168.101.165) from 00:0d:60:9c:21:df via eth1
May 31 14:35:47 hs20b05 dhcpd: DHCPACK on 192.168.101.164 to 00:0d:60:9c:21:df
via eth1
May 31 14:35:47 hs20b05 dhcpd: DHCPACK on 192.168.101.164 to 00:0d:60:9c:21:df
via eth1
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.0 to
192.168.101.164:2070
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.0 to
192.168.101.164:2070
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.0 to
192.168.101.164:2071
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.0 to
192.168.101.164:2071
May 31 14:35:47 hs20b05 atftpd[12685]: Serving
pxelinux.cfg/01-00-0d-60-9c-21-df to 192.168.101.164:57089
May 31 14:35:47 hs20b05 atftpd[12685]: Serving
pxelinux.cfg/01-00-0d-60-9c-21-df to 192.168.101.164:57089
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.cfg/C0A865A4 to
192.168.101.164:57090
May 31 14:35:47 hs20b05 atftpd[12685]: Serving pxelinux.cfg/C0A865A4 to
192.168.101.164:57090
May 31 14:35:47 hs20b05 atftpd[12685]: Serving xcat/rhels5/x86/vmlinuz to
192.168.101.164:57091
```

```
May 31 14:35:47 hs20b05 atftpd[12685]: Serving xcat/rhels5/x86/vmlinuz to
192.168.101.164:57091
May 31 14:35:48 hs20b05 atftpd[12685]: Serving xcat/rhels5/x86/initrd.img to
192.168.101.164:57092
May 31 14:35:48 hs20b05 atftpd[12685]: Serving xcat/rhels5/x86/initrd.img to
192.168.101.164:57092
.......... <<<<< Omitted lines >>>>> ..........
```

To see the output on the remote console, open a remote console by using the **rcons** command.

If terminal server is not working, the other option is to open the remote console from the Web browser. The remote console from management module works if the system supports SOL.

## 4.2.18 Updating the node root user's known_hosts file

After the node state changes to sshd, gather SSH public keys of the node by using **ssh-keyscan**, and then update it in /root/.ssh/known_hosts of the management node, as shown in the Example 4-61.

*Example 4-61 Update the SSH public key*

```
[root@xcat2mgmt scripts]# ssh-keyscan -t rsa hs20b04
>>/root/.ssh/known_hosts
# hs20b04 SSH-2.0-OpenSSH_4.3
```

After this step, you can log in to the node without entering the password.

A

# Additional material

This paper refers to additional material that can be downloaded from the Internet, as described in the following sections.

## Locating the Web material

The Web material associated with this paper is available in softcopy on the Internet from the IBM Redbooks Web server:

ftp://www.redbooks.ibm.com/redbooks/REDP4437

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional Materials** and open the directory that corresponds with the IBM Redbooks form number, REDP4437.

# Using the Web material

Additional Web material that accompanies this paper includes the following file:

**xCAT_ITSO_tools.tar** This is a tarball that contains sample scripts for CSM to xCAT database conversion and incorporating GPFS into diskless node image

## System requirements for downloading the Web material

The following system configuration are recommended:

**Hard disk space**: 1 MB minimum
**Operating system**: Red Hat Enterprise Linux 5.1 x86 32/64 bit and ppc4

## How to use the Web material

Create a subdirectory (folder) on your xCAT management node, and then unpack the contents of the Web material tarball file into this folder.

# Related publications

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

## IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 123. Note that the following document might be available in softcopy only:

*“IBM Eserver xSeries and BladeCenter Server Management”*, SG24-6495

## Online resources

The following Web sites are also relevant as further information sources:

- xCAT documentation Web site:

  http://xcat.svn.sourceforge.net/svnroot/xcat/xcat-core/trunk/xCAT-client/share/doc

- CSM documentation page

  http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=/com.ibm.cluster.infocenter.doc/library.html

## How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications, and Additional materials, as well as order hardcopy Redbooks, at this Web site:

**ibm.com**/redbooks



 

# Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

# xCAT 2 Guide for the CSM System Administrator

Redpaper™

**xCAT architecture overview**

**xCAT 2 quick deployment example**

**CSM to xCAT transition scenarios**

This IBM Redbooks publication positions the new Extreme Cluster Administration Toolkit 2.x (xCAT 2) against the IBM Cluster Systems Management (CSM) for IBM Power Systems and IBM System x in a High Performance Computing (HPC) environment.

This paper provides information to help you:

- Understand, from a broad perspective, a new clustering management architecture. The paper emphasizes the benefits of this new solution for deploying HPC clusters of large numbers of nodes.
- Install and customize the new xCAT cluster management in various configurations.
- Design and create a solution to migrate from existing CSM configurations to xCAT-managed clusters for various IBM hardware platforms.

.

**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

REDP-4437-00