

Integrating Hardware Management with Cluster Administration Toolkits

Jacob Liberman, Garima Kochhar, Arun Rajan, Munira Hussain, Onur Celebioglu

Dell, Inc.

{Jacob_Liberman, Garima_Kochhar, Arun_Rajan, Munira_Hussain, Onur_Celebioglu}@dell.com

Abstract— Cluster administration toolkits simplify the deployment and management of large commodity clusters by incorporating scalable installers with management and cataloguing interfaces. Although these toolkits facilitate operating system level cluster administration, they typically lack tightly integrated frameworks for hardware management. Administrators who wish to leverage the standards based hardware management capabilities commonly available in commodity clusters are left to their own devices. Most opt to write their own integration scripts which can be a costly and inefficient process. This paper presents a case study for resolving this problem by integrating standards based hardware management utilities into a typical open source deployment and administration framework.

I. INTRODUCTION

The TOP500 list provides a biannual ranking of the world's fastest supercomputers as measured by the High Performance Linpack (HPL) benchmark. Although the TOP500 does not present a comprehensive view of the technical computing landscape, it provides a reliable basis for identifying trends in the industry. According to the June 2007 TOP500 list clusters are the dominant architecture used in the world's fastest supercomputer sites with 373 of 500 entries. [1] Clusters of commodity servers have emerged as the dominant supercomputing architecture due to the phenomenal price for performance they can deliver.

Commodity clusters' rapid ascension through the ranks of elite supercomputing solutions has not been without growing pains. The technical computing community has become increasingly aware that extraneous costs contribute to the ongoing expense of maintaining a large cluster. The power consumed by running and cooling a large cluster can incur sizeable expense. [2] Their size also makes them difficult to manage. Tasks easily performed on a single server become impractical as the number of servers grows from hundreds to thousands, and the operational costs associated with hiring additional administrative staff can quickly eat away cost savings. [3]

The inherent complexity in administering many servers is further compounded by the number of specialized networks – called fabrics – required to operate a cluster. Each fabric provides a specific function. The cluster fabric is used for intra-process communication as it transports messages related to the application running on the cluster. This fabric is typically comprised of low latency, high bandwidth host channel adapters (HCAs) and switches for communication intensive applications. InfiniBand is a common interconnect

choice for cluster fabrics. [1] The administration fabric is used to deploy and manage the cluster nodes. Tasks performed over the administration fabric include installing a node's operating system, accessing user home directories, and applying security patches. Finally, the out-of-band (OOB) management fabric affords operating system independent hardware management via dedicated controllers. Tasks performed over the OOB management fabric include changing a node's BIOS configuration or powering on systems that are currently powered off.

Each specialized fabric grants additional capability at the cost of added complexity. Instead of managing the cluster via one set of tools, administrators must learn, execute, and monitor an additional set of utilities for each fabric. Figure 1 highlights the complexity introduced by multiple management fabrics in a typical commodity cluster.

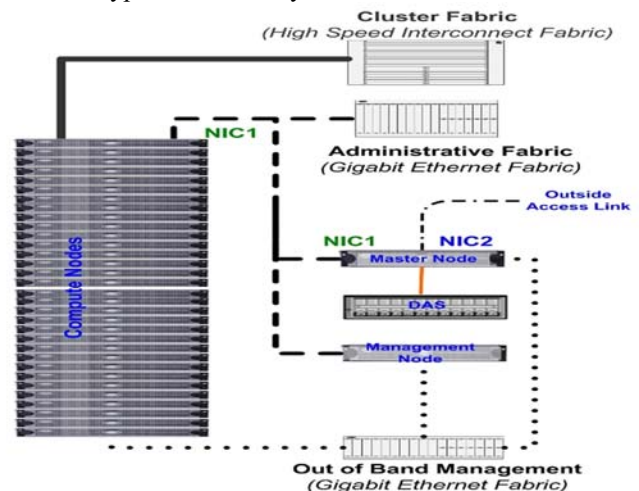


Fig. 1 This diagram depicts cluster, administrative, and out of band management fabrics.

There are many cluster management utilities which mitigate the administrative burden associated with managing large clusters. While the majority of these utilities are excellent choices for operating system level management, most do not provide robust support for the OOB management controllers. Administrators who wish to leverage these controllers end up either writing scripts which integrate the OOB management tools into their existing management framework, employing nonintegrated tools in an unsystematic way, or giving up on OOB management altogether. None of these alternatives is optimal. Custom scripts are costly to develop and maintain and likely to break when changes are made to the underlying

technology. Nonintegrated tools are also expensive in terms of the time it takes administrators to master yet another set of commands which may be very different from the tools provided by their current management utility. Ignoring the tools altogether might prove to be the most costly alternative. OOB hardware management controllers can trivialize time consuming tasks -- such as manually power cycling two thousand nodes following a power outage -- which would otherwise cost dearly in terms of lost productivity.

This paper provides the model for a complete cluster management framework which seamlessly integrates OOB hardware management with operating system level cluster management. The first half of this paper describes the technologies used to demonstrate the integration: a standards-based Baseboard Management Controller (BMC) for hardware management and Platform Open Cluster Stack (OCS), an open source cluster management package. Platform OCS includes software developed by the Rocks Cluster Group at the San Diego Supercomputer Center. This paper also details the extent to which hardware management and OCS are currently integrated. The second half of the paper presents the results of a use case for the functional integration of the BMC with OCS. The paper concludes with directions for future research.

II. CLUSTER PACKAGE OVERVIEW

There are many proprietary and open source software packages that provide cluster management capabilities. [4] Popular open source cluster packages include Rocks, OSCAR, OpenMOSIX and Warewulf. [5, 6, 7, 8] Each has its own set of features. Platform Open Cluster Stack, a commercially supported cluster software package developed by Platform Computing Inc, is based on the Rocks framework. [9, 10] It is designed to make clusters easy to deploy, manage, maintain, and scale, and it is built on standard and open source components. The Rocks framework takes a modular approach to software management where optional software is bundled into packages called 'rolls' which can be flexibly installed into the distribution as needed. [11] Rocks has been used to deploy approximately 900 registered clusters with a combined CPU count of over 55,000. [5] Platform OCS offers most of the features present in Rocks as well as several add-on scripts and utilities. Although this study used OCS, the lessons learned can be easily adapted to any cluster management package.

At the heart of OCS is a SQL database maintained on the master node of the cluster. The database contains global cluster configuration data such as appliance types, node names, network addresses, and partitioning information. These database records are used to generate critical host configuration files every time a node is installed or re-installed, ensuring cluster-wide consistency of host configuration data.

OCS also provides an extensible mechanism for customizing cluster node installations via policy based inheritance. XML based 'node' and 'graph' files define inheritance relationships. Each node file specifies a service and its configuration. The graph files define the relationships

between the node files. By creating new nodes and defining edges, the definition of a compute node can be extended to meet the unique requirements of the cluster. Each custom definition contains a complete set of software packages and configuration details for the node. The definition is then used to generate a custom Red Hat Linux kickstart utilized by the Anaconda based installer. The installer creates host-specific kickstart files by traversing the graph. Figure 2 depicts a kickstart graph highlighting the relationships between node files.

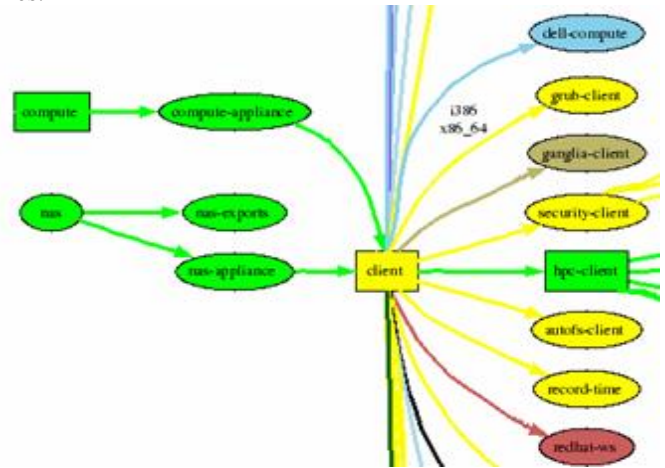


Fig. 2 Node/graph relationships

In addition to the database and installer, OCS provides operating system level management tools. For example, 'insert-ethers' populates the database with new records as hosts are added to the cluster. The 'cluster-fork' utility simultaneously executes a specified command on all or some subset of cluster nodes. OCS includes several utilities -- such as 'dbreport' -- for querying and updating the central database. The combination of the installer, database, and administrative tools simplifies the tasks associated with managing large clusters.

III. IPMI OVERVIEW

In the past hardware management was primarily achieved via vendor-specific proprietary solutions. Contemporary clusters built from commodity components utilize an open-standard hardware manageability interface based on the Intelligent Platform Management Interface (IPMI) specification. [12] IPMI was developed and continues to be jointly promoted by various hardware vendors including Dell, HP, Intel, and NEC. IPMI provides a common interface for accessing system hardware such as environmental sensors, chassis power control, BIOS configuration, and system event logs. With cross-platform support, standard IPMI utilities can be used to manage servers from different vendors.

IPMI specification 1.0 -- announced in September of 1998 at the Intel Developer Forum -- provided the foundation for Intelligent Platform Management. It defined operating system and processor independent feature support for monitoring, logging, autonomous access, and control. Taken together these features provided an interoperable framework for systems management and asset tracking intended to drive down the

total cost of managing a heterogeneous data centre. IPMI 1.5 added support for remote management interfaces and extensions for additional standards while retaining backward compatibility with IPMI 1.0. As IPMI systems management proliferated security became an increasing concern. IPMI 2.0 - the latest version of the IPMI specification - added enhanced authentication, encryption, and remote access capabilities while retaining backward compatibility with IPMI 1.5.

The IPMI specification was used to design the interface to the Baseboard Management Controller (BMC). The BMC is an on-board micro-controller that provides comprehensive out of band management features including power management, console redirection, and system event monitoring. Administrators can use the BMC to monitor hardware components and environmental conditions inside the server chassis such as fans, temperatures, and voltage. It also enables out-of-band access to the server, allowing administrators to control the server even when it is powered down or the operating system is not responding. The BMC is accessible over the network via either a dedicated or shared network interface. The diagram below describes the BMC subsystem.

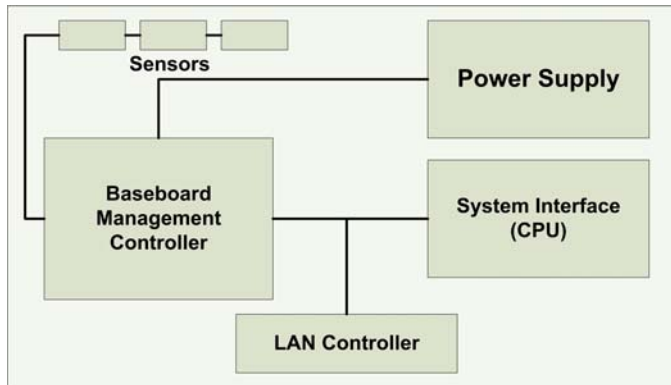


Fig. 3 High level overview of IPMI 2.0 architecture

There are several open source tools which leverage the BMC including IPMITool and OpenIPMI. IPMITool is a command line interface which provides an authenticated session-based means of controlling the BMC across a Local Area Network (LAN). [13] The OpenIPMI project includes Linux kernel modules and libraries that expose an open operating system level interface to the local server's BMC. The OpenIPMI kernel module is a fully functional Linux device driver. It enhances the functionality of the BMC to include support for watchdog timers, interrupts, multiple users, and multiple interfaces. [14] Versions of both IPMITool and OpenIPMI are available with most enterprise Linux distributions.

IV. DESIGN SPECIFICATIONS

Most cluster management packages do not include robust support for hardware management. The research team first defined the following specifications for integrating hardware management with a cluster toolkit. These specifications were then used to implement a robust integration of IPMI hardware management with the OCS cluster toolkit.

A. No-touch installation mechanism

The cluster toolkit should remove the administrative burden associated with implementing hardware management by installing all required software and automatically configuring the hardware management interfaces.

OCS installs OpenIPMI by default and provides limited configuration of the BMC network parameters. However it lacks a mechanism for logically associating the BMC interface with its host system and configuring secure remote authentication remains a manual process.

Rocks 4.3 - which is based on Red Hat Enterprise Linux (RHEL) 4.5 -- includes the OpenIPMI software package, but currently lacks a mechanism for integrating IPMI compatible hardware management controllers such as the BMC into its framework.

B. Integration with the cluster database

The OCS database inventories cluster hardware and acts as a data source for many of its administrative tools. Although the insert-ethers command includes a 'management' appliance type, OCS makes no attempt to logically associate the management appliance with its host system during installation. OCS should not only configure the BMC interfaces but also add them to the database in such a way that they are logically mapped to their host system. For example, bmc-0-0 resides in compute-0-0.

C. Full support for the cluster toolkit's existing administration utilities

Most cluster toolkits provide a robust set of administrative utilities for managing cluster nodes. One of the primary drawbacks of developing custom scripts for managing hardware controllers is that the administrator loses the opportunity to leverage the powerful tools included with their toolkit. This problem is compounded by the fact that custom tools which are not integrated into the overall management framework are likely to break when changes are made to the framework's underlying structure. Therefore, the final specification agreed to by the research team was that existing utilities - such as insert-ethers, cluster-fork, and dbreport in OCS -- should interface with the hardware management controllers wherever possible.

V. TEST CASE

The general premise that guided the research team's design decisions was that OCS should behave the same way toward the hardware management interfaces that it behaves toward cluster nodes. This meant that the management controllers should be deployed with the same utilities used to deploy the cluster nodes, the management controller configuration information should be stored in the same database as the cluster nodes' configuration information, and that the same tools used to manage the cluster nodes should be used to manage the hardware controllers.

A. Deployment

The research team began by creating a custom roll that bundled all the software and configuration commands required to configure the hardware management interfaces in the cluster nodes. The BMC roll extended the existing compute node configuration with additional commands for configuring and enabling the BMC. The additional commands – executed during the post install section of each compute node’s kickstart file – detected the existence of the BMC via a Desktop Management Interface (DMI) query, created an IPMI device, assigned the BMC IP address and additional network parameters such as the netmask and default gateway, and configured the BMC to allow remote access. [15]

The deployment process also created a logical association between the BMC and its host compute node by mirroring the naming convention used by OCS. For example, the BMC in node compute-0-0 was named bmc-0-0. Every BMC IP address, hostname, and MAC address was entered into the cluster database using standard MySQL insert statements. Integrating the BMC’s into the database also required the research team to create a custom appliance type named BMC. By following these steps, the research team successfully integrated the BMC controller into the OCS deployment framework and reproducibly deployed and configured the BMC interfaces during several test installations.

B. Simulate node failure

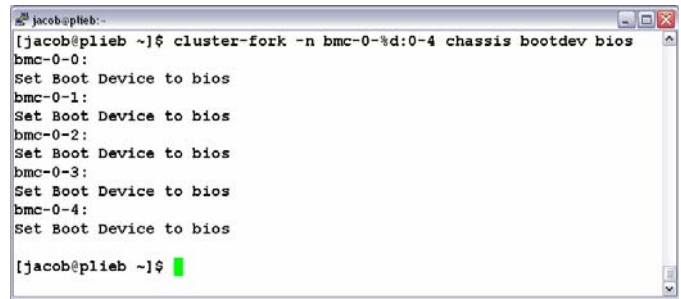
In the second phase of the test case the research team simulated a node hardware failure and replacement by physically replacing a cluster node with a like model. The existing OCS framework for replacing a failed node – using ‘insert-ethers –replace *nodename*’ --worked with only one slight modification. The research team had to remove the BMC interface from the database with ‘insert-ethers –remove’ prior to replacing the failed node. The BMC configuration was successfully reapplied during the replacement server’s reinstallation.

C. Modify BIOS configuration for one rack of cluster nodes

During the final use case the research team attempted to demonstrate the full integration of the BMC with both the database and OCS management utilities. This was accomplished by issuing an IPMI command to the BMCs with cluster-fork. The cluster-fork command queries the database for all hosts of the specified appliance type and issues the same command to them in parallel.

The research team modified the cluster-fork utility so that it could execute IPMI commands on the BMC interfaces over the network without authentication. Next the research team issued a command which would force the hosts to boot into their BIOS configuration menu on the subsequent reboot. The results of the command are illustrated in figure 4.

The command executed successfully without interacting with the host system at the operating system level. At this point the IPMI operating system services were not loaded or running, reducing the overall amount of system memory consumed by the hosts’ operating systems.



```
jacob@plieb:~$ cluster-fork -n bmc-0-%d:0-4 chassis bootdev bios
bmc-0-0:
Set Boot Device to bios
bmc-0-1:
Set Boot Device to bios
bmc-0-2:
Set Boot Device to bios
bmc-0-3:
Set Boot Device to bios
bmc-0-4:
Set Boot Device to bios
[jacob@plieb ~]$
```

Fig. 4 Result of third use case.

VI. CONCLUSION

Cluster management utilities mitigate much of the administrative overhead associated with deploying and managing large scale clusters. However, many widely used cluster management utilities do not take advantage of the robust hardware management capabilities available in commodity servers. Administrators are left with the unenviable task of implementing individual solutions for this common problem. This use case demonstrated that there is much to be gained from a tighter integration between open source cluster toolkits and standards based hardware management interfaces like IPMI. Although OCS was used as the cluster software package to demonstrate how such integration might be accomplished, the specifications and rationale presented in this paper can inform the ongoing design decisions for any cluster management toolkit.

Directions for future research include leveraging IPMI support through the various cluster health monitoring utilities such as Ganglia and Clumon for temperature sensor and event log reporting. The solution would also benefit from incorporating the enhanced security features available in IPMI 2.0 such as public key encryption for remote authentication.

REFERENCES

- [1] (2007) General highlights from the Top 500 since the last edition. [Online]. Available: <http://www.top500.org/lists/2007/06/highlights/general>
- [2] R. Ali, B. Guler, R. Radhakrishnan, and V. Sahasrabudhe, “Evaluating Scalability and Power benefits of Ninth-Generation Dell PowerEdge Servers in an HPC Environment,” *Dell Power Solutions*, pp. 44-47, 2006.
- [3] P. Papadopoulos, M. Katz, and G. Bruno, “NPACI Rocks: Tools and Techniques for Easily Deploying Manageable Linux Clusters,” *cluster*, pp. 258, 3rd IEEE International Conference on Cluster Computing (CLUSTER’01), 2001.
- [4] R. Gupta, B. Guler, S. Iqbal, and M. Kashyap, “Felix, Scale, and Rocks: An Introduction to Cluster Computing Solutions,” *Dell Power Solutions*, pp. 68-64, 2004.
- [5] (2007) Rocks Cluster register. [Online]. Available: <http://www.rocksclusters.org/rocks-register/>
- [6] (2007) What is OSCAR. [Online]. Available: <http://oscar.openclustergroup.org/>
- [7] (2007). The OpenMOSIX Project. [Online]. Available: <http://openmosix.sourceforge.net/>
- [8] (2007) Introduction to Warewulf. [Online]. Available: <http://warewulf-cluster.org/warewulf.html>
- [9] R. Ali, R. Gupta, G. Kochhar, and B. Bryce, “Platform Rocks: A Cluster Software Package for Dell HPC Platforms,” *Dell Power Solutions*, pp. 29-34, 2005.

- [10] B. Bryce, G. Kochhar, R. Gupta, R. Ali, "Platform Open Cluster Stack: An Enhanced Cluster Software Package for Dell HPC Platforms," *Dell Power Solutions*, pp. 54-58, 2006.
- [11] G. Bruno, M. Katz, F. Sacerdoti, and P. Papadopoulos, "Rolls: Modifying a Standard System Installer to Support User-Customizable Cluster Frontend Appliances," *cluster*, pp. 421-430, 2004 IEEE International Conference on Cluster Computing (CLUSTER'04), 2004
- [12] (2007) Intelligent Platform Management Interface. [Online]. Available: <http://www.intel.com/design/servers/ipmi/spec.htm>
- [13] (2007) IPMITool. [Online]. Available: <http://ipmitool.sourceforge.net/>
- [14] (2007) OpenIPMI. [Online]. Available: <http://openipmi.sourceforge.net/>
- [15] G. Kochhar, R. Ali, A. Rajan, "Configuring the BMC and BIOS on Dell Platforms in HPC Cluster Environments," *Dell Power Solutions*, pp. 12-15, 2005.