

Building and Measuring the Performance of Windows HPC Server 2008–Based Clusters for TOP500 Runs

Microsoft Corporation

Published: November 2008

Abstract

The TOP500 project ranks and describes the 500 most powerful publicly known computer systems in the world. The list has been compiled twice a year since 1993 and aims to provide a reliable basis for tracking and detecting trends in high-performance computing. Rankings are based on High-Performance LINPACK (HPL), a portable implementation of the LINPACK benchmark for distributed-memory computers.

This white paper provides guidance for deploying and optimizing a Windows® HPC Server 2008-based cluster. It also describes how to run the HPL benchmark to both verify that the cluster is operating well and to tune the cluster for optimal performance. Finally, the paper explains how to submit a cluster to the TOP500 List.

The paper focuses on the best practices for setting up and tuning a large Windows HPC Server 2008–based cluster; it also documents the considerations that are unique to running HPL on large-scale Windows®-based compute clusters.

mslogo_DS_300

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, Excel, SQL Server, Visual Studio, Windows, the Windows logo, and Windows Server are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Before You Begin 5](#_Toc214096080)

[Overview 9](#_Toc214096081)

[The TOP500 List 10](#_Toc214096082)

[What Is the Purpose of the TOP500 List? 10](#_Toc214096083)

[The Growth of Clusters on the TOP500 List 11](#_Toc214096084)

[A Brief History of Clusters Running Windows Clustering Software on the TOP500 List 11](#_Toc214096085)

[June 2006 List: First Cluster Running Windows Clustering Software to Make TOP500 List 12](#_Toc214096086)

[June and November 2007 Lists: Additional Windows High-Performance Clusters Make the List 12](#_Toc214096087)

[June 2008 List: First Cluster Running Windows Clustering Software in Top 25 of TOP500 List 12](#_Toc214096088)

[Why Make a TOP500 Run? 13](#_Toc214096089)

[Will a Cluster Achieve TOP500 Level Performance? 14](#_Toc214096090)

[The LINPACK Benchmark 14](#_Toc214096091)

[Build a Windows HPC Server 2008–Based Cluster 16](#_Toc214096092)

[Plan the Cluster 16](#_Toc214096093)

[Prepare for Deployment 17](#_Toc214096094)

[Review System Requirements 18](#_Toc214096095)

[Decide How to Add Compute Nodes to the Cluster 19](#_Toc214096096)

[Choose the Active Directory Domain 20](#_Toc214096097)

[Choose a User Account for Installation and Diagnostics 20](#_Toc214096098)

[Choose the Network Infrastructure 21](#_Toc214096099)

[Choose a Network Topology 21](#_Toc214096100)

[Size the Network 22](#_Toc214096101)

[Prepare for Multicast 22](#_Toc214096102)

[Prepare for Integration of Scripted Power Control Tools 22](#_Toc214096103)

[Prepare for High-Speed Interconnect Installation 23](#_Toc214096104)

[Deploy the Cluster 24](#_Toc214096105)

[Deploy the Head Node 24](#_Toc214096106)

[Configure the Head Node 27](#_Toc214096107)

[Build and Deploy on a Small Scale 30](#_Toc214096108)

[Add Compute Nodes to the Cluster 30](#_Toc214096109)

[Verify the Cluster 33](#_Toc214096110)

[Use the Built-in Diagnostics 34](#_Toc214096111)

[Use the uSane Toolkit 34](#_Toc214096112)

[Disable Performance Counter Collection and Event Forwarding 40](#_Toc214096113)

[Summary of Considerations Unique to Large-Scale Clusters 40](#_Toc214096114)

[Benchmark a Windows HPC Server 2008—Based Cluster 41](#_Toc214096115)

[Prepare for an HPL Run 41](#_Toc214096116)

[Download the HPL Benchmark 41](#_Toc214096117)

[Install the Necessary Software 42](#_Toc214096118)

[Create and Configure the Makefile and Build the Benchmark 42](#_Toc214096119)

[Run HPL on the Cluster 42](#_Toc214096120)

[Assign the HPL Parameters 43](#_Toc214096121)

[Run a Test Job Using the Excel LINPACK Sizing Sheet 45](#_Toc214096122)

[Review the Results 47](#_Toc214096123)

[Tune the HPL Parameters 47](#_Toc214096124)

[Turn Off Non-Essential Services 47](#_Toc214096125)

[Use the Excel Add-In 49](#_Toc214096126)

[Use the LINPACK Tuning Wizard 53](#_Toc214096127)

[Submit the Results 58](#_Toc214096128)

[Method of Solution 58](#_Toc214096129)

[Verification 58](#_Toc214096130)

[Restrictions 58](#_Toc214096131)

[Additional Resources 60](#_Toc214096132)

[Appendix A: Sample Node XML File 61](#_Toc214096133)

[Appendix B: The HPL Parameters 62](#_Toc214096134)

[Appendix C: Additional Information about the HPL Parameters 64](#_Toc214096135)

[HPL Processes Distribution 64](#_Toc214096136)

# Before You Begin

There are many tasks required to build and measure the performance of a large-scale Windows HPC Server 2008–based cluster. These tasks are outlined below; additional detail is provided in the white paper.

Please review the checklists carefully before you begin. Careful preparation can save time and help ensure the success of your Windows HPC Server 2008 cluster.

* Checklist 1: Prepare for deployment
* Checklist 2: Deploy the cluster
* Checklist 3: Verify the cluster
* Checklist 4: Run the HPL benchmark

**Checklist 1:** **Prepare for deployment**

* [**Prepare for deployment**](#_Prepare_for_Deployment)  
  Before you start deploying your HPC cluster, review the list of prerequisites and initial considerations. Large-scale clusters, in particular, require careful preparation.
  1. [*Review system requirements*](#_Review_System_Requirements)  
     Review the list of initial considerations and system requirements to ensure that you have all the necessary hardware and software components to deploy an HPC cluster. Also, ensure that the boot order on the compute nodes’ BIOS is set to PXE boot first.
     1. *Hardware requirements*
     2. *Software Requirements*
  2. [*Decide how to add compute nodes to your cluster*](#_Decide_How_to)  
     Recommended practice is to provision the compute nodes via an image and a template, and then add the compute nodes to your cluster with an XML file.
  3. [*Choose the Active Directory® Domain Services (AD DS) domain for your cluster*](#_Choose_the_Active)  
     Choose the Active Directory domain to which you will join the head node and compute nodes of your HPC cluster.
  4. [*Choose a user account for installation and diagnostics*](#_Choose_a_User)  
     Choose an existing domain account with enough privileges to perform installation and diagnostics tasks.
  5. [*Choose the network infrastructure*](#_Choose_the_Network)  
     Choose the high-speed interconnect and the DHCP/DNS configuration.
     1. *Choose the interconnect*
     2. *Choose a DHCP/DNS Configuration*
  6. [*Choose a network topology*](#_Choose_a_Network)  
     Choose how the nodes in your cluster will be connected.
  7. [*Size the network*](#_Size_the_Network)  
     Network sizing is an additional consideration for large-scale clusters.
  8. [*Prepare for multicast*](#_Prepare_for_Multicast)  
     If your private network switch supports multicast, use it to speed deployment.
  9. [*Prepare for the integration of scripted power control tools*](#_Prepare_for_Integration)   
     If you want to use your own power control tools to start (in addition to shut down and reboot) compute nodes remotely, obtain and test all the necessary components.
  10. [*Prepare for high-speed interconnect installation*](#_Prepare_for_High-Speed)  
      You need to ensure that your interconnect drivers are updated.

**Checklist 2:** **Deploy the cluster**

* [**Deploy the cluster**](#_Deploy_the_Cluster)Develop and test a compute node image, and create a template. Next, deploy and configure the head node and add nodes to the cluster by importing an XML file.
  1. [*Deploy the head node*](#_Deploy_the_Head)  
     Install Windows Server 2008 and the Microsoft HPC Pack 2008.
     1. *Install the operating system on the head node*Install Windows Server 2008 HPC Edition (or another 64-bit edition of Windows Server 2008) on the head node.
     2. *Install the high-speed interconnect drivers*  
        Install the interconnect drivers on the head node using the Windows Device Manager.
     3. *Join the head node to a domain*  
        Join the head node to an Active Directory domain.
     4. *Set up SQL Server on the head node*  
        Set up SQL Server Standard, not the included version of SQL Server Express.
     5. *Install Microsoft HPC Pack 2008 on the head node*   
        Install Microsoft HPC Pack 2008 on the head node, using the installation media.
  2. [*Configure the head node*](#_Configure_the_Head)  
     Configure the head node by following the steps in the configuration to-do list, and build a sample image to use in deployment.
     1. *Configure the HPC cluster network*  
        Verify that the private network has no other PXE or DHCP servers, and then configure the cluster network by using the Network Configuration Wizard.
     2. *Provide installation credentials*  
        Specify which credentials to use for system configuration, for adding new nodes to the cluster, and for running diagnostics.
     3. *Configure the naming of new nodes*  
        Specify the naming convention to use when generating names automatically for new compute nodes.
     4. *Create a node template*  
        Create a template that defines the steps to follow when configuring a compute node.
     5. *Add drivers for the operating system images*  
        Add drivers, including the InfiniBand drivers, for the operating system images that you created for your node template on the previous task.
     6. *Add or remove users (optional)*  
        If you will be giving access to the cluster to other members of your organization, add or remove users or administrators for your cluster.
  3. [*Build and test deploy on a small scale*](#_Build_and_Deploy)Test deploy manually on a small number of nodes.
  4. [*Add compute nodes to the cluster*](#_Adding_Compute_Nodes_1)  
     Add nodes to the cluster by importing an XML file.
     1. *Add the compute nodes*

Although you can use several methods to add compute nodes, the most efficient method for a large-scale cluster is to use the node.xml file template.

* + 1. *Monitor progress*

Monitor progress as you add compute nodes.

* + 1. *Power up the nodes*

Power up the compute nodes using the method you selected.

* + 1. *Flash the firmware*

Verify the firmware version on your networking host channel adapter (HCA) card and on your network switch using tools from your vendor.

**Checklist 3:** **Verify the cluster**

* [**Verify the cluster**](#_Verify_the_Cluster)  
  After you have configured your head node and added all compute nodes to the cluster, run diagnostic tests to validate cluster functionality and troubleshoot any configuration issues.
  1. [*Built-in diagnostics*](#_Built-in_Diagnostics_1)   
     Run the MPIPingPong test and output an XML file containing latency and throughout measurements and statistics
  2. [*uSane Toolkit*](#_uSane_Toolkit)   
     To identify any potential performance issues, the uSane toolkit runs a small C ping-pong test between nodes.
     1. *Pre-uSane Testing*

Start by ensuring that the compute nodes are visible, and that you can use a Run command and get results.

* + 1. *Set up uSane*

Copy uSane to a share that is accessible to your workstation, the head node, and the compute nodes

* + 1. *Run the uSane tests*
  1. [*Disable performance counter collection and event forwarding*](#_Disable_Performance_Counter)

**Checklist 4:** **Run the HPL benchmark**

* [**Prepare for the HPL run**](#_Prepare_for_an)Follow these steps to prepare for the HPL run—alternately, use the LINPACK Tuning Wizard.
  1. [*Download the HPL benchmark*](#_Download_the_HPL)  
     Download the benchmark from Netlib into a shared partition.
  2. [*Install the necessary software*](#_Install_the_Necessary)  
     Install the software that is necessary to run the benchmark.
  3. [*Create and configure the Makefile and build the benchmark*](#_Create_and_Configure)  
     Create the Makefile and edit it so that it reflects your particular cluster configuration.
* [**Run HPL on the cluster**](#_Run_HPL_on)  
  Follow these steps to run HPL
  1. [*Assign the HPL parameters*](#_Assign_the_HPL)   
     Become familiar with the HPL parameters. These will be tuned to enhance performance.
  2. [*Run a test job*](#_Run_a_Test)   
     Run a small test job on the cluster to verify operation using the Excel LINPACK sizing sheet.
  3. [*Review the results*](#_Review_the_Results)  
     Review the resulting output file.
  4. [*Tune the parameters*](#_Tune_the_HPL)  
     Tune the HPL parameters to optimize performance.
  5. [*Turn off non-essential services (optional)*](#_Turn_Off_Non-Essential)  
     Optimize performance further by turning off all non-essential services.
  6. [*Use the Excel add-in (optional)*](#_Use_the_Excel)Use the Excel add-in for tuning. You can also use the add-in as an interface to LINPACK.
* [**Use the LINPACK Tuning Wizard**](#_Use_the_LINPACK)

The LINPACK Tuning Wizard automates many of the procedures described in the previous sections, including validating the cluster and optimizing the LINPACK benchmark.

# Overview

The goal of this white paper is to encourage customers to build Windows® HPC Server 2008–based compute clusters, to attempt High-Performance LINPACK (HPL) benchmarking runs on these clusters, and to submit the benchmark results to the TOP500 List.

Windows HPC Server 2008, which is comprised of Microsoft® HPC Pack 2008 and the Windows Server® 2008 HPC Edition operating system, provides a productive, cost-effective high-performance computing (HPC) solution that runs on x64-bit hardware. Windows HPC Server 2008 can be deployed, managed, and extended using familiar tools and technologies.

Windows HPC Server 2008 enables broader adoption of HPC by providing a rich and integrated user experience. A wide range of software vendors, in various vertical markets, have designed their applications to work seamlessly with Windows HPC Server 2008, so that users can submit and monitor jobs from within familiar applications without having to learn new or complex user interfaces.

Developing parallel programs requires integrated development environments, along with support for distributed computing standards. The Microsoft® Visual Studio® 2008 development system provides a comprehensive parallel programming environment for Windows HPC Server 2008. In addition to supporting the message passing interface (MPI), OpenMP, and Web services, Windows HPC Server 2008 also supports third-party numerical library providers, performance optimizers, compilers, and a native parallel debugger for developing and troubleshooting parallel programs.

Windows HPC Server 2008 was designed primarily for clusters with up to 256 compute nodes; clusters that are larger than this have some unique requirements—and these are the focus of this white paper. The recommendations presented, for deployment and for benchmarking, are based on the real-world experience that Microsoft has had with large-scale clusters that have been ranked on the TOP500 List.

This white paper begins by introducing the TOP500 List and emphasizing its importance. This paper also provides a brief overview of the HPL benchmark that is used by the TOP500 List to rank computer systems. Because a benchmarking run requires a commitment of resources and time, this paper intends to provide the motivation customers may need to help them decide to attempt TOP500 runs, even if their clusters are unlikely to actually make the TOP500 List.

Next, this white paper provides guidance for deploying, verifying, and tuning a Windows HPC Server 2008–based cluster for best performance. The paper includes general, easy-to-follow steps with links for additional, more detailed information; considerations that are unique to a large-scale TOP500 run are discussed. The aim of this paper is to provide enough information so that an IT generalist is able to set up and optimize the performance of a large-scale Windows®-based cluster.

This white paper then discusses how to build and prepare the HPL benchmark. Planning and tuning tools are examined and recommended, including a Microsoft® Office Excel® add-in for sizing parameters, submitting jobs, and managing and analyzing the results; these tools are essential for verifying and optimizing the performance of large Windows-based clusters. The paper also highlights the LINPACK Tuning Wizard, a tool to automate the process of optimizing HPL for a large-scale cluster.

Finally, this white paper describes how to refine the parameter values to optimize performance, and most importantly, how to submit the results to the TOP500 List. References are also given for further information.

# The TOP500 List

The TOP500 List ranks and describes the 500 most powerful publicly-known computer systems in the world. The list has been compiled twice a year since 1993 with the help of high-performance computing experts, computational scientists, manufacturers, and the general Internet community. The first of the biannual updates coincides with the International Supercomputing Conference (ISC) held in Europe every year in June; the second update is presented each November at the Institute of Electrical and Electronics Engineers (IEEE) Supercomputing Conference (SC Conference) in the United States.

TOP500 rankings are based on HPL, a portable implementation of the LINPACK benchmark for distributed-memory computers. Computers are ordered by the maximal sustained LINPACK performance they achieved (Rmax value). In the case of equal performances, the computers are ordered by theoretical peak performance (Rpeak). TOP500 results, along with a wealth of related statistics, are published at <http://www.top500.org/>.

## What Is the Purpose of the TOP500 List?

*“After 15 years and 30 lists, we have managed to establish TOP500 among HPC users, manufacturers and the media as THE instrument for analyzing the HPC market.”*

*Prof. Dr. Hans Werner Meuer, University of Mannheim & Prometeus GmbH, Germany, January 20, 2008*

The TOP500 List is the most prestigious ranking of its kind. Vendors and HPC users can capitalize on favorable rankings to promote their work and their organizations. For the many government and academic sites that have traditionally dominated the list, a good TOP500 showing can help make a strong case when seeking funding for an HPC installation. For the organizations and suppliers whose computers make the TOP500 List, the rank they achieve provides publicity and bragging rights. A TOP500 ranking can be cited to great advantage in publications and presentations.

While the TOP500 List does lay down a competitive challenge, it also compiles information about the variety of approaches to advanced computing technologies. The statistics on high-performance computers provided by the TOP500 List—such as the number of systems installed, the physical location of the various supercomputers within the HPC community, and the applications for which these computer systems are being used—are of major interest to manufacturers and to potential HPC users. These statistics can provide an understanding of the high-performance computer market and can help foster communication and collaboration within the HPC community by disseminating information about the capabilities and technical specifications of the various large and powerful computer systems.

The TOP500 List provides a reliable basis for tracking and detecting trends in HPC. Over the years, the TOP500 List has served well as a tool for documenting and analyzing technological and architectural changes in the HPC arena, including the tremendous growth in performance levels and the increase in the number of compute clusters.

The HPC market is very dynamic by nature, and this is clearly reflected in the TOP500 List. The average replacement rate has been about 160 systems every six months—thus more than half of the list is new every year. While the list was originally dominated by specialized HPC companies, it is now composed primarily of mainstream manufacturers from the workstation and PC market segments.

It is important to remember, however, that a system’s TOP500 rank does not necessarily reflect the performance users will obtain from their applications. The list reflects the ability of a system to solve a predetermined, linear set of equations; it does not indicate how a system may perform with other workloads. The TOP500 List, therefore, should not be used as a primary tool for selecting a supercomputing system for an organization.

## The Growth of Clusters on the TOP500 List

The number of clusters on the TOP500 List has increased dramatically over the 15 years that the list has been compiled. The first cluster to make the TOP500 List was the Berkeley Network of Workstations (NOW) in June of 1997. By the June 2008 TOP500 List, 400 systems were identified as clusters, as shown in Figure 1, taken from the TOP500 Web site. The figure also shows the changes over time in massively parallel processing (MPP) systems, symmetric multiprocessing (SMP) systems, constellations (commodity clusters with more microprocessors in a node than nodes in the cluster), and single processor systems.

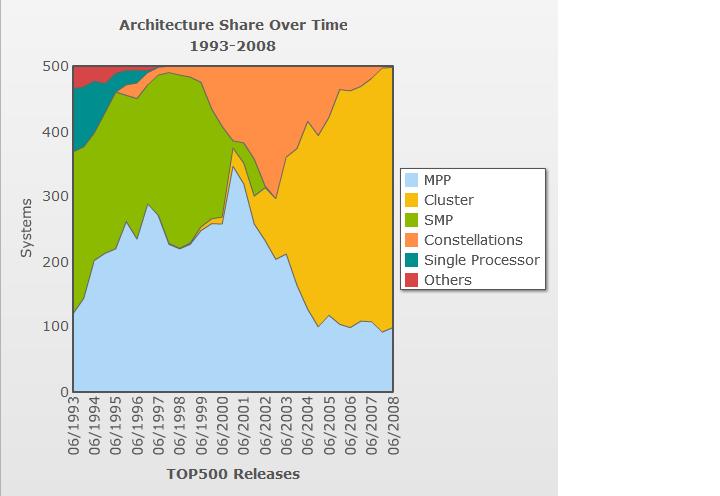


Figure 1 Architecture Share over Time

## A Brief History of Clusters Running Windows Clustering Software on the TOP500 List

The number and performance clusters running the Windows operating system on the TOP500 List have increased dramatically as well—from a single entry with a rank of 130 on the June 2006 list (with a performance of 4.1 teraflops, or 1012 floating point operations per second), to three entries on the June 2008 list (with a top performance of 68.5 teraflops).

### June 2006 List: First Cluster Running Windows Clustering Software to Make TOP500 List

The first Windows HPC cluster to enter the TOP500 List was the National Center for Supercomputing Applications (NCSA) cluster “Lincoln,” which achieved a rank of 130 in June of 2006. The Lincoln cluster ran Windows® Compute Cluster Server 2003 on a 3.2 GHz Dell PowerEdge 1855 with a Cisco InfiniBand (IB) interconnect. The system had 900 cores on Intel EM64T Xeon (3200 MHz, 6.4 gigaflops) processors. The Rmax was 4,106 gigaflops, and the Rpeak was 5,760 gigaflops.

### June and November 2007 Lists: Additional Windows High-Performance Clusters Make the List

Two clusters running Windows Compute Cluster Server 2003 made it to the TOP500 List in June of 2007.

A 448-node cluster, which was designed to run risk management simulations at Mitsubishi UFJ Securities, achieved a rank of 193. This system is based on an IBM BladeCenter HS21 cluster (3 GHz Xeon dual-core with Gigabit Ethernet interconnect) with 1,792 processors, generating 6.52 teraflops (Rmax).

The second Windows-based cluster added to the list in June of 2007 was “Rainier,” a 256-node, 2,048-core cluster of Dell PowerEdge 1955 blades running Intel Xeon 5300 quad-core processors (1866 MHz, 7.464 gigaflops) and Cisco InfiniBand switches, achieving 8.99 teraflops. This cluster, which is owned by Microsoft and installed at its data center at Tukwila, Washington, earned a ranking of 106 on the list.

By November of 2007, the Rainier cluster achieved 11.75 teraflops using the same hardware, but with an upgrade to a pre-release version of the Windows HPC software. Despite the increased performance, the cluster achieved a rank of 116 compared to the previous 106 rank, emphasizing just how quickly performance capabilities are increasing. It is interesting to note that the entry ranked last on the June 2008 TOP500 List achieved just under 9.0 teraflops—the same performance achieved by the 106th-ranked Rainier cluster just six months earlier!

### June 2008 List: First Cluster Running Windows Clustering Software in Top 25 of TOP500 List

At the June 2008 International Supercomputing Conference, three clusters running Windows HPC Server 2008 made the TOP500 List.

Microsoft debuted in the top 25 of the list for the first time with the NCSA cluster “Abe,” ranking at 23. The NCSA cluster used a beta version of Windows HPC Server 2008 to achieve its 68.5 teraflops and 77.7 percent efficiency on 9,472 cores (2.33 GHz Dell PowerEdge 1955, InfiniBand SDR interconnect), making this facility one of the most powerful supercomputing systems in the world and the fastest Windows cluster to date.

Computer scientists at Umeå University in northern Sweden, also working with a beta version of Windows HPC Server 2008, achieved 46 teraflops and 85.5 percent efficiency on 5,376 cores, making their system the second-largest Windows cluster ever deployed and the fastest academic cluster in Sweden. The university’s cluster employs 672 IBM blade servers (BladeCenter HS21, 2.5 GHz Intel Xeon QC HT, InfiniBand), and also marks the first time that Windows HPC Server 2008 has been run publicly on IBM hardware. The Umeå system achieved a rank of 39 on the TOP500 List. It also achieved a rank of 27 on the Green500 List, which provides a ranking of the most energy-efficient supercomputers in the world and serves as a complementary view to the TOP500 List. Green500 results are published at <http://www.green500.org/home.php>.

The Center for Computing and Communication (CCC) at RWTH Aachen University in Germany deployed a 256-node cluster with Windows HPC Server 2008 and an InfiniBand network specifically to perform a TOP500 run (Fujitsu PRIMERGY RX200 S4, 3000 MHz Intel EM64T Xeon, 12 gigaflops) in April of 2008. The cluster reached position 100 in the June 2008 list, achieving 18.81 teraflops and 77 percent efficiency with only minimal performance tuning.

The table that follows summarizes the history of Windows-based clusters in the TOP500 List.

Table 1 TOP500 List Clusters Running Windows HPC Software

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Date | Cluster Location | TOP500 Rank Achieved | Number of Cores | Performance | Efficiency | Software |
| June 2006 | NCSA | 130 | 896 | 4.1 teraflops | Not Available | Windows Compute Cluster 2003 |
| June 2007 | Microsoft | 106 | 2,048 | 9.0 teraflops | 58.8% | Windows Compute Cluster 2003 |
| November 2007 | Microsoft | 116 | 2,048 | 11.8 teraflops | 77.0% | Windows HPC Server 2008 |
| June 2008 | Aachen (Germany) | 100 | 2,096 | 18.8 teraflops | 76.5% | Windows HPC Server 2008 |
| June 2008 | Umeå (Sweden) | 40 | 5,376 | 46.0 teraflops | 85.5% | Windows HPC Server 2008 |
| June 2008 | NCSA | 23 | 9,472 | 68.5 teraflops | 77.7% | Windows HPC Server 2008 |

## Why Make a TOP500 Run?

With HPC gradually becoming part of the general fabric of enterprise computing, the publicity that a Windows-based TOP500 run generates can encourage industries that may not have utilized HPC in the past to consider HPC for production. As companies look to migrate from their current systems to an HPC environment, the ability to use the familiar Windows operating system becomes very enticing.

Attempting a TOP500 run can also be beneficial to smaller systems that may never approach the performance levels needed to make the TOP500 List. Though there is much debate about how relevant the HPL benchmark is to industry, it remains an excellent “burn in” test for very complex HPC systems. HPL is a good tool for validating a system: it provides a self-check to make sure all is working well, and it stresses the system more than typical applications would.

It makes sense then to have the HPL run be the first test of a newly installed cluster; HPL can thus be used as the final diagnostic before putting a cluster into production. Note that it is important to ensure that the cluster is returning correct results in the benchmarking test—not just fast results.

### Will a Cluster Achieve TOP500 Level Performance?

It can be difficult to predict whether an HPL run will achieve the performance required for a TOP500 listing. The TOP500 List moves very quickly, and the level of performance required to make the list is increasing at an incredible rate, as can be seen in Figure 2, a graph from the TOP500 Web site. Note that Figure 2 is a semi-log graph—the vertical axis increases exponentially. Thus the positive linear increase in projected performance indicates exponential growth.

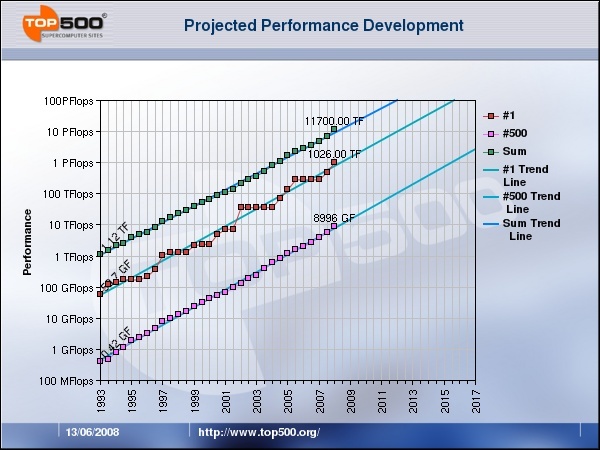


Figure 2 Projected TOP500 performance

The performance a cluster achieves is based on both the number of cores and the speed of the interconnect—the faster the interconnect, the fewer cores are needed to achieve a certain level of performance. A high-performance interconnect is therefore essential; at a minimum, Gigabit Ethernet (GigE) is required, but for even higher performance, the cluster requires a high-speed application interconnect such as 10 Gigabit Ethernet (10 GigE), InfiniBand, or Myrinet. Though it is difficult to predict accurately, a cluster in 2008 or early 2009 using InfiniBand is likely to need at least 3,000 cores to achieve TOP500List performance.

## The LINPACK Benchmark

The originators of the TOP500 List selected LINPACK as a basis for ranking because it is widely used and freely available, and because performance numbers are available for most relevant systems. **HPL**, the portable implementation of the LINPACK benchmark for distributed-computing systems, provides a testing and timing program to quantify the **accuracy** of the obtained solution, as well as the time it took to compute it. The HPL algorithm and its attached implementation are scalable: though the best **performance** achievable on a system depends on many factors (with some restrictive assumptions concerning the interconnection network), solving larger and larger matrices can maximize the parallel efficiency with respect to the per-processor memory usage.

Again, note that the result of the LINPACK test is not intended to reflect the overall performance of a system; instead, it provides a measure of the performance of a dedicated system for solving a specific set of linear equations. Since the LINPACK problem is very regular, the performance achieved is quite high; the performance numbers give a good indication of peak performance and provide an idea of how much floating-point performance is theoretically possible from a given system. Most HPC applications exhibit much more complex behavior than LINPACK does, so the benchmark does not necessarily give a good indication of real-world performance. In other words, production performance will not be as high as the LINPACK results.

The HPL software package **requires** the Message Passing Interface (**MPI**), as well as an implementation of **either** the Basic Linear Algebra Subprograms (**BLAS) or** the Vector Signal Image Processing Library (**VSIPL)**. Machine-specific, in addition to generic implementations of MPI, BLAS, and VSIPL, are available for a wide variety of systems.

For frequently asked questions about the LINPACK benchmark, go to <http://www.netlib.org/utk/people/JackDongarra/faq-linpack.html>

The white paper “The LINPACK Benchmark: Past, Present, and Future” provides detailed information; go to <http://www.netlib.org/utk/people/JackDongarra/PAPERS/hpl.pdf>.

# Build a Windows HPC Server 2008–Based Cluster

Cluster deployment has changed dramatically since the early days of compute clusters, when all cluster installation was manual. In those days, a careful system administrator kept detailed records of the configuration of each computer so that in the very likely event that a node failed, it would be relatively easy to install and configure a new cluster member to take its place.

With Windows HPC Server 2008, administrators no longer need to do the record-keeping: an automated cluster deployment package records the data on a computer designated as the head node of the cluster. Detailed records are now stored in a Microsoft® SQL Server® database, so they can be used to automatically generate the configuration files needed to keep the cluster organized. Once a cluster is set up, the maintenance tools included in Windows HPC Server 2008 let the user maintain both the cluster software and hardware.

Because Windows HPC Server 2008 was designed for clusters with up to 256 compute nodes, there are a few additional factors to consider that are specific to larger clusters. These are detailed in the sections that follow and summarized in the section [Summary of Considerations Unique for Large-Scale Clusters](#_Summary_of_Considerations). The basic conceptual information and general procedures that an IT generalist needs for installing a large-scale Windows-based HPC cluster are also provided; links are given throughout for more detailed information and troubleshooting.

For step-by-step deployment procedures, see the Design and Deployment Guide for Windows HPC Server 2008 at <http://go.microsoft.com/fwlink/?LinkID=117921>.

**Special note to Linux administrators**Though similar in many ways, Linux and Windows take a slightly different approach to clustering. Linux distributions are generally heavily modified if they are to be used for clustering. Windows clustering software, on the other hand, takes an off-the-shelf version of the operating system and adds the clustering pack to it—rather than customizing the operating system, Windows uses the standard operating system and adds a clustering layer to it.

## Plan the Cluster

The following checklist describes the overall process of deploying a Windows HPC Server 2008–based cluster. Most of the procedures for a large cluster are the same as those for a standard-size cluster; the differences are detailed in the sections that follow.

**Note:** Tasks in the checklists are linked to the sections in this document that provide more detail about the tasks.

Table 2 Checklist for Deploying a Windows HPC Server 2008 Cluster

|  |  |
| --- | --- |
| Task | Description |
| 1. [Prepare for deployment](#_Preparing_for_Deployment) | Before you start deploying your HPC cluster, review the list of prerequisites and initial considerations. Large-scale clusters, in particular, require careful preparation. |
| 2. [Deploy the cluster](#_Deploying_the_Cluster) | Develop and test a compute node image, and create a template. Next, deploy and configure the head node and add nodes to the cluster by importing an XML file. |
| 3. [Verify the cluster](#_Verifying_the_Cluster_1) | Run the built-in MPIPingPong and the uSane diagnostic tests to verify that the deployment of the cluster was successful. |

### Prepare for Deployment

In order for a TOP500 run to be successful, it is critical to plan carefully. The first step in the deployment of a large-scale HPC cluster is to make important decisions, such as deciding how you will add nodes to your cluster and choosing a network topology for your cluster.

**Special note to Linux administrators**  
The role of the head node in a Linux-based cluster differs from that in a Windows-based cluster. Windows-based clusters rely on robust head nodes that support many services. Linux-based clusters, on the other hand, may have spread many of these services to multiple infrastructure nodes or other servers that are not part of the cluster proper.

The following checklist describes the steps involved in preparing for your deployment.

Table 3 Checklist for Preparing for Deployment

|  |  |
| --- | --- |
| Task | Description |
| 1. [Review system requirements](#_Reviewing_System_Requirements) | Review the list of initial considerations and system requirements to ensure that you have all the necessary hardware and software components to deploy an HPC cluster. Also, ensure that the boot order on the compute nodes’ BIOS is set to PXE boot first. |
| 2. [Decide how to add compute nodes to your cluster](#_Deciding_How_to) | Recommended practice is to provision the compute nodes via an image and a template, and then add the compute nodes to your cluster with an XML file. |
| 3. [Choose the Active Directory domain for your cluster](#_Choosing_the_Active) | Choose the Active Directory domain to which you will join the head node and compute nodes of your HPC cluster. |
| 4. [Choose a user account for installation and diagnostics](#_Choosing_a_User) | Choose an existing domain account with enough privileges to perform installation and diagnostics tasks. |
| 5. [Choose the network infrastructure](#_Choosing_the_Network) | Choose the high-speed interconnect and the DHCP/DNS configuration. |
| 6. [Choose a network topology](#_Choosing_a_Network_1) | Choose how the nodes in your cluster will be connected. |
| 7. [Size the network](#_Preparing_for_Multicast) | Network sizing is an additional consideration for large-scale clusters. |
| 8. [Prepare for multicast](#_Preparing_for_Multicast_1) | If your private network switch supports multicast, use it to speed deployment. |
| 9. [Prepare for the integration of scripted power control tools](#_Preparing_for_Integration) | If you want to use your own power control tools to start (as well as shut down and reboot) compute nodes remotely, obtain and test all the necessary components. |
| 10. [Prepare for high-speed interconnect installation](#_Preparing_for_InfiniBand) | You need to ensure that your interconnect drivers are updated. |

### Review System Requirements

First, take the time to review the system hardware and software requirements.

#### Hardware Requirements

Hardware requirements for Windows HPC Server 2008 are identical to those for the 64-bit editions of the Windows Sever 2008 operating system; for more information, go to <http://go.microsoft.com/fwlink/?LinkID=119578>.

The number of network adapters on the head node and on the compute nodes depends on the network topology that you choose for your cluster. For more information about the different HPC cluster network topologies, see the section [Choose a Network Topology](#_Choosing_a_Network) later in this document, or go to <http://go.microsoft.com/fwlink/?LinkId=124377>.

It is a best practice for the computer designated as the head node of a large-scale cluster to meet the following requirements:

* At least four CPU cores
* At least two disks
* At least 8 GB of RAM

Because the head node may perform many different functions, it may have very different requirements from the rest of the compute nodes.

#### Software Requirements

The following list outlines the software requirements for the head node and the compute nodes in a Windows HPC Server 2008 cluster:

* Windows Server 2008 HPC Edition, or another 64-bit edition of the Windows Server 2008 operating system
* Microsoft HPC Pack 2008
* Microsoft® SQL Server® Standard

For large-scale clusters with more than 128 nodes, it is necessary to use Microsoft SQL Server Standard instead of Microsoft® SQL Server® 2005 Express Edition, the version included in Windows HPC Server 2008.

Though SQL Server Express is identical to the retail versions of SQL Server in many respects, there are some key benefits provided by the full versions of SQL Server:

* Unlimited database size (SQL Server Express databases are limited to 4 GB in size)
* SQL Server Management Studio for configuring and maintaining your SQL Server database
* Support for high-availability configurations
* Multi-threading to take advantage of multiple processors on the head node
* Unlimited RAM usage for SQL Server caching

For the high data loads used in a TOP500 HPL run, it is also important to add a couple of disks to the head node in order to split the SQL Server log file from the SQL Server database; ideally, during the installation sequence, place the system partition, data, and logs on separate spindles. Alternately, high-performance Redundant Array of Independent Disks (RAID) can be used instead of the three spindles.

**Note:** For smaller clusters, placing both the log file and the data on the C: drive is acceptable.

To enable users to submit jobs to your HPC cluster, you can install the utilities included with Microsoft HPC Pack 2008 on client computers. Software requirements for the client computers can be found at <http://technet.microsoft.com/en-us/library/cc947680.aspx>.

Additional software is needed for running HPL; this is described in the [Install the Necessary Software](#_Installing_the_Necessary) section later in this document.

### Decide How to Add Compute Nodes to the Cluster

After provisioning the compute nodes with an imaging template, you add the compute nodes to the cluster. There are three options for adding compute nodes:

* **Add preconfigured compute nodes**. The compute nodes are already running Windows Server 2008 HPC Edition (or another 64-bit edition of Windows Server 2008) and Microsoft HPC Pack 2008 is manually installed on each node. This is the option for users who wish to use a deployment technology other than Windows® Deployment Services; however, note that this option is not recommended for large-scale clusters.
* **From bare metal**. The operating system and all the necessary HPC cluster components are automatically installed on each compute node as it is added to the cluster. No manual installation of the operating system or other software is required. With this option, the head node acts as the Windows Deployment Services provider and automatically discovers the compute nodes when they are booted from the Pre-Boot Execution Environment (PXE). The administrator then selects the nodes and deploys to them. This can be cumbersome with large numbers of nodes.
* **Import a node XML file**. An XML file that contains a list of all the nodes to be deployed is used. This XML file can be used to add nodes from bare metal or from preconfigured nodes.

For more information about node XML files, go to <http://technet.microsoft.com/en-us/library/cc707389.aspx>.

For a large-scale cluster, a best practice is to use the third option, importing a node XML file. The preconfigured node.xml file can then also be used for naming the compute nodes. This is described in the section [Configure the Naming of New Nodes](#_Configuring_the_Naming).

### Choose the Active Directory Domain

The head node and the compute nodes in your HPC cluster must be members of an Active Directory domain. Before deploying your cluster, you must choose the Active Directory domain to use for your HPC cluster. A single Active Directory server is sufficient, but for large-scale deployments, it must be a separate server.

A best practice in a large-scale cluster is to use a single dedicated server for Active Directory Domain Services; it is important not to combine head node and Active Directory roles because of the large amount of processing power required for deployment.

If the cluster is a standalone domain, a best practice is to use domainname.local.

For more information about installing the Active Directory Domain Services role on a computer that is running Windows Server 2008, go to <http://go.microsoft.com/fwlink/?LinkID=119580>.

### Choose a User Account for Installation and Diagnostics

During the configuration process of your HPC cluster, you must provide credentials for a domain user account to be used for installation and diagnostics. You can choose an existing account or create a new account before starting the cluster deployment. The user account that you choose must be a domain account with enough privileges to create Active Directory computer accounts for the compute nodes.

Note that if part of your deployment requires access to resources on the enterprise network, the user account must have the necessary permissions to access those resources. Note also that if you want to be able to restart nodes remotely from the cluster administration console, the account must be a member of the local Administrators group on the head node (necessary only if you do not have scripted power control tools that you can use to remotely restart the compute nodes).

### Choose the Network Infrastructure

Setting up the network is key to a large-scale TOP500 installation.

#### Choose the Interconnect

You need to use a high-speed interconnect, preferably one that supports NetworkDirect, for running applications on a large-scale cluster. In this guide, we describe procedures for InfiniBand; procedures for other high-speed interconnects are similar.

**Note:** Compute nodes are not deployed over the high-speed interconnect.

#### Choose a DHCP/DNS Configuration

A best practice is to use a separate infrastructure server(s) running the Windows operating system for all network services.

With a large number of nodes, you need to keep close control over naming; it is best not to rely on basic input/output system (BIOS), globally unique identifiers (GUIDs), or Media Access Control (MAC) addresses.

If your application servers need a fixed IP address on the network, you can use DHCP reservations (a permanent IP address assignment) to assign IP addresses. For example, you can create DHCP reservations based on MAC addresses (mapped to a physical location), obtaining predetermined static addresses for the compute nodes and setting static addresses for the MPI based on the private IP.

For large-scale clusters, a best practice is to run DHCP remotely instead of managing it though the Windows HPC Server 2008 user interface.

**Note:** For large-scale deployments, create reverse DNS zones, as they do not get set up automatically. Omitting this step may delay name resolution. For more information, go to <http://support.microsoft.com/kb/174419>.

### Choose a Network Topology

Next, choose a network topology, taking into consideration your existing network infrastructure. Windows HPC Server 2008 supports five cluster topologies, which are distinguished by the way the compute nodes in the cluster are connected to each other and to the enterprise network. Choosing one of the supported cluster topologies simplifies all aspects of cluster setup and management.

For a detailed description of the supported topologies, go to <http://technet.microsoft.com/en-us/library/cc719008.aspx> or <http://go.microsoft.com/fwlink/?LinkId=124377>.

Large-scale clusters require a high-speed application network. Therefore, a best practice for large-scale clusters is to use a topology with the compute nodes isolated on private and application networks (topology 3) or a topology with all nodes on enterprise, private, and application networks (topology 4).

An isolated private network can increase the reliability and can be critical to the success of TOP500 clusters; the private network should be truly private (on a separate subnet), and not on the Internet or Intranet(s). There should be one DHCP server and no other PXE servers on the private network.

**Note:** The application network should include all applicable management software. For example, you need a dedicated subnet manager (SM) server for InfiniBand; you can use OpenSM, which provides an implementation of an InfiniBand Subnet Manager and Administration, running with a high priority.

### Size the Network

A best practice is to use a private, non-routed network. For large-scale clusters, an additional consideration is network sizing. A typical 192.168.x.0 network, for example, holds up to 254 nodes (the head node reserves one address for itself). IT administrators must plan ahead for “network space” when more than 255 nodes are required.

Sizing the network can be complex, and may require consultation with a networking expert. You can also consult the “Networking and Network Access Protection (NAP)” section of the *Windows Server 2008 Resource Kit* (Microsoft Press® 2008).

Plan to configure your subnet mask on the cluster network to allow as many hosts as needed. For example, if you have 255.255.252.0 for 1,021 nodes**,** statically define the head node cluster interface, and then configure the DHCP scope starting and ending IP address. For 10.10.(0-3).(1-254), you would have a starting address of 10.10.0.1 and an ending address of 10.10.3.254. You can repeat the process if additional IP networks are needed, but alter the third octet to select the next free network. (For example, a second network would use 10.10.[4-7].x for its IP range.)

### Prepare for Multicast

Deployment is significantly faster with multicast. Therefore, it is a best practice is to use multicast for large-scale clusters if the switches support it.

To prepare for multicast:

* Enable Internet Group Management Protocol (IGMP) snooping on your network switches, if this feature is available. IGMP snooping makes it possible for the switch to "listen in" on the IGMP conversation between hosts and routers by processing the layer 3 IGMP packets sent in a multicast network. IGMP snooping can effectively reduce multicast traffic from streaming and from other bandwidth-intensive IP applications.
* Enable “portfast” mode on ports, which lets the Spanning Tree Protocol (STP) run on your network switches without interfering with the PXE boot process.

**Note:** If you are having any problems with multicast, refer to the documentation for your switches (frequently they are the source of the problem).

### Prepare for Integration of Scripted Power Control Tools

The cluster administration console includes actions to manage compute nodes remotely, including powering on the compute nodes after deployment. These actions are linked to a script file (CcpPower.bat) that performs these power control operations using operating system commands. You can replace the default operating system commands in that script file with your own power control scripts, such as Intelligent Platform Management Interface (IPMI) scripts, if desired.

Following is a script for the sample tool powersw.exe:

@setlocal

@echo off

if L%1 == Lon goto on

if L%1 == Loff goto off

if L%1 == Lcycle goto cycle

echo "usage:CcpPower.cmd [on|off|cycle] nodename [ipaddress]"

goto done

:on

rem exit /b 1

powersw.exe %3 private on %2

goto done

:off

rem shutdown /s /t 0 /f /m \\%2

powersw.exe %3 private off %2

goto done

:cycle

rem shutdown /r /t 0 /f /m \\%2

powersw.exe %3 private reboot %2

goto done

:done

exit /b %ERRORLEVEL%

endlocal

The arguments to be customized in the script are:

%1 – on/off/recycle  
%2 – NodeName  
%3 – ManagementIPAddress

For information about modifying CcpPower.bat to integrate your own scripted power control tools, go to <http://go.microsoft.com/fwlink/?LinkId=124378>.

### Prepare for High-Speed Interconnect Installation

To prepare for the application network deployment, install the high-speed interconnect hardware onto the head node and compute nodes.

## Deploy the Cluster

When you have finished the planning phase, you are ready to begin building the cluster and deploying the cluster nodes.

Table 4 Checklist for Deploying the Cluster

|  |  |
| --- | --- |
| Task | Description |
| 1. [Deploy the head node](#_Deploying_the_Head) | Install Windows Server 2008 and the Microsoft HPC Pack 2008. |
| 2. [Configure the head node](#_Configuring_the_Head_1) | Configure the head node by following the steps in the configuration To-Do List, and build a sample image to use in deployment. |
| 3. [Build and test deploy on a small scale](#_Adding_Compute_Nodes) | Test deploy manually on a small number of nodes. |
| 4. [Add compute nodes to the cluster](#_Adding_Compute_Nodes_1) | Add nodes to the cluster by importing an XML file. |

### Deploy the Head Node

The head node is deployed first. As noted, a high-speed interconnect (such as InfiniBand) is required on the head node in order to run the MPI diagnostics.

The following checklist describes the steps involved in deploying the head node.

Table 5 Checklist for Deploying the Head Node

|  |  |
| --- | --- |
| Task | Description |
| 1. [Install the operating system on the head node](#_Installing_the_Operating_1) | Install Windows Server 2008 HPC Edition (or another 64-bit edition of Windows Server 2008) on the head node. |
| 2. [Install the high-speed interconnect drivers](#_Installing_the_InfiniBand) | Install the interconnect drivers on the head node using the Windows Device Manager. |
| 3. [Join the head node to a domain](#_Joining_the_Head_1) | Join the head node to an Active Directory domain. |
| 4. [Set up SQL Server on the head node](#_Installing_Microsoft_HPC) | Set up SQL Server Standard, not the included version of SQL Server Express. |
| 5. [Install Microsoft HPC Pack 2008 on the head node](#_Installing_Microsoft_HPC_1) | Install Microsoft HPC Pack 2008 on the head node, using the installation media. |

#### Install the Operating System on the Head Node

To deploy the head node of your HPC cluster, install Windows Server 2008 HPC Edition (or another 64-bit edition of Windows Server 2008) on the computer that will act as the head node. It is recommended that you obtain the latest device drivers for your head node computer from the hardware vendor’s Web site.

For more information about installing Windows Server 2008, including system requirements, go to <http://go.microsoft.com/fwlink/?LinkID=119578>.

#### Install the High-Speed Interconnect Drivers on the Head Node

You first need to obtain the drivers for your high-speed interconnect.

The 10 GigE iWARP Remote Direct Memory Access (RDMA) adapters supporting NetworkDirect are available to enable Windows HPC Server 2008 to run over 10 GigE or standard Ethernet infrastructures. For more information, go to <www.microsoft.com/hpc>.

You can obtain InfiniBand NetworkDirect-enabled drivers from your InfiniBand vendor, the Mellanox Web site (<http://www.mellanox.com/products/MLNX_WinOF.php>), or the OpenFabrics Alliance (<http://www.openfabrics.org/>).

##### Perform the Preliminary Steps for InfiniBand

In this guide, we describe procedures for InfiniBand.

You first need to copy the InfiniBand (IB) drivers onto the head node.

If you are updating an older driver set on the head node, you need to first deregister the NetworkDirect provider by running the following command from an elevated (administrator) command prompt:

C:\Windows\System32\ndinstall –r

If the older driver set was installed with a Windows Installer or via the Windows Device Manager, uninstall the driver set from the command line using msiexec. For more information about using msiexec, go to <http://technet.microsoft.com/en-us/library/cc759262.aspx>.

Next, install the IB drivers on the head node using the Windows Device Manager.

From the head node,register the NetworkDirect provider using the tool provided by your network hardware vendor. Typically, this tool is ndinstall.exe; it is run from an elevated (administrator) command prompt:

C:\Windows\System32\ndinstall –i

For more information about customizing driver installation, go to <http://msdn.microsoft.com/en-us/library/ms791073.aspx>.

Verify that the provider is registered using:

C:\Windows\System32\ndinstall –l

##### Start an InfiniBand Subnet Manager (SM)

For large-scale clusters (over approximately 500 nodes), run OpenSM on a dedicated (separate) computer; do not run OpenSM on the head node. OpenSM provides an implementation of an InfiniBand Subnet Manager (SM) and administration, which is required to initialize the InfiniBand hardware.

A best practice for a large-scale cluster is to use a PC-based subnet manager instead of a switch-based manager.

You need to copy the subnet manager to system32 and register the service. You can then start the service.

copy /Y opensm.exe %SystemRoot%\system32

sc create opensm binPath= "%SystemRoot%\system32\opensm.exe --service" start= auto DisplayName= "IB Subnet Manager"

sc start opensm

If the SM is running as a service on the head node and if the head node is also serving as a DHCP server for the application network, there may be a timing issue in which the DHCPserver service starts before the SM. The SM must be up and running before the DHCPserver service starts, otherwise the DHCP server will not create a binding for the application network. (This is because the application network appears as “disconnected” until the SM is up and running.)

To ensure that the SM is running before the DHCPserver service starts, add an OpenSM dependency for the DHCPserver service as follows:

sc stop dhcpserver

sc config dhcpserver depend= RpcSs/Tcpip/SamSs/EventLog/EventSystem/opensm

sc start dhcpserver

#### Join the Head Node to a Domain

The head node must be a member of an Active Directory domain. After you have installed Windows Server 2008 on the head node, manually join the head node to an existing Active Directory domain.

#### Set Up SQL Server on the Head Node

You need to set up Microsoft® SQL Server® 2005 x64. For more information about SQL Server, go to <http://www.microsoft.com/sql/prodinfo/overview/productguide.mspx>.

#### Install Microsoft HPC Pack on the Head Node

Installing the Microsoft HPC Pack 2008 is an easy, wizard-driven process. Note that if you did not perform a clean installation of the operating system on the head node and are installing Microsoft HPC Pack 2008 on an existing installation of Windows Server 2008, you need to remove all server roles first.

The installation of HPC Pack 2008 adds the following server roles to the head node:

* DHCP server, to provide IP addresses and related information for compute nodes
* Windows Deployment Services, to deploy compute nodes remotely
* File Services, to manage shared folders
* Network Policy and Access Services, to enable the Routing and Remote Access Service (RRAS) so that network address translation (NAT) services can be provided to the cluster nodes

**Note:** Unless you have a high-performance RAID array, separate the SQL Server data, log, and system on three separate spindles.

### Configure the Head Node

The head node is configured by following the configuration To-Do List in Windows HPC Cluster Manager.

The following checklist covers the items that you need to complete.

Table 6 Checklist for Configuring the Head Node

|  |  |
| --- | --- |
| Task | Description |
| 1. [Configure the HPC cluster network](#_Configuring_the_HPC) | Verify that the private network has no other PXE or DHCP servers, and then configure the cluster network by using the Network Configuration Wizard. |
| 2. [Provide installation credentials](#_Providing_Installation_Credentials) | Specify which credentials to use for system configuration, for adding new nodes to the cluster, and for running diagnostics. |
| 3. [Configure the naming of new nodes](#_Configuring_the_Naming) | Specify the naming convention to use when generating names automatically for new compute nodes. |
| 4. [Create a node template](#_Creating_a_Node) | Create a template that defines the steps to follow when configuring a compute node. |
| 5. [Add drivers for the operating system images](#_Adding_Drivers_for) | Add drivers, including the InfiniBand drivers, for the operating system images that you created for your node template on the previous task. |
| 6. [Add or remove users (optional)](#_Adding_or_Removing) | If you will be giving access to the cluster to other members of your organization, add or remove users or administrators for your cluster. |

#### Configure the HPC Cluster Network

Verify that there is no other PXE server or DHCP server on your private network.

Run the command ipconfig/renew. This re-establishes TCP/IP connections on all network adapters. If the computer is unable to find a DHCP server, the automatic private IP address (APIPA) service assigns itself a 169.x.x.x IP address. If it returns anything other than an address in the range 169.x.x.x, then you need to investigate and troubleshoot.

To check if there are other PXE servers, put a server on the network and attempt to PXE boot it. If there is a PXE server on the private network, it takes over and begins to boot. Stop the process and troubleshoot.

The HPC cluster network configuration is the first step in the configuration process of your head node. When configuring the HPC cluster network, choose the network topology that you have selected for your cluster, as described in [Prepare for Your Deployment](#_Preparing_for_Deployment).

Before you start the process, ensure that you are able to identify to which network each one of the network adapters in the head node is connected.

Also ensure that PXE is enabled and that the boot order on all compute nodes is PXE first.

**Note:** As discussed earlier, you must choose a topology that includes an application network with an active subnet manager.

#### Provide Installation Credentials

Installation credentials must be provided in order to configure new compute nodes. These credentials are used when installing the operating system and applications, and when adding nodes to the Active Directory domain. These same credentials are used when running diagnostic tests on the compute nodes. Installation credentials are provided from the To-Do List.

#### Configure the Naming of the New Nodes

A best practice for large-scale clusters is to create an XML file containing the GUID and/or MAC addresses of each node, and to use a naming convention that follows the physical location of the node (for example, BC01CN010 for BladeCenter 01 and Compute Node 10). [Appendix A](#_Appendix_A:_Sample) provides a sample script for a node.xml file.

You can create the node XML file in any XML editor or text editor, but it must follow the node XML file schema. For more information about node XML files, go to <http://technet.microsoft.com/en-us/library/cc707389.aspx>.

You can also create a node XML file from an HPC cluster that is already configured by exporting it from the Windows HPC Cluster Manager.

When creating a node XML file for a deployment from bare metal, you need a hardware identification parameter for each compute node. This parameter can be the SMBIOS GUID or the MAC address of the computer., and can be obtained from the OEM.

When creating a node XML file for deployment from bare metal:

* You must specify a node template for each compute node listed, and that node template must include a step to deploy an operating system image. If you do not specify a node template, or if you specify a node template that does not include a step to deploy an operating system image, the deployment will fail.
* Ensure that the node template names that are specified in the node XML file match the names of the node templates listed on the head node.
* Specify any location information that you want to be attached to the node.
* If you want nodes to be automatically added to specific node groups during deployment, specify the Tag attribute with the name of the node group for each compute node.
* If your integration of scriptable power control tools requires a Base Management Controller (BMC) IP address for each compute node, it can be added to the node XML file.

**Note:** The Node.xml file can also be generated with Excel by using the XML Map feature.

If you are deploying compute nodes from bare metal and you are not using a node XML file to import nodes to the cluster, Windows HPC Server 2008 automatically generates computer names for the new nodes that are being deployed. You need to specify how those names are generated by defining a naming series; this series is defined by selecting a root name and the starting number, enclosed in percentage signs (%), that accompany that name (for example, ClusterNode%1000%). The nodes are named in sequence, as they become available. The compute node naming series is specified from the To-Do List.

#### Create a Node Template

Node templates can help simplify the deployment process. They define the necessary tasks for configuring and adding compute nodes to your cluster and let you deploy an operating system image, add specific drivers and software to compute nodes, or simply add a preconfigured node to your cluster. Because you might have more than one type of compute node, or you may be adding compute nodes to your cluster in different ways, you can create different templates that apply to different nodes or situations.

You can create two types of node templates:

* **With an operating system image**This type of template includes a step to deploy an operating system on the compute nodes. Use this type of template when adding compute nodes from bare metal. For this type of node template, you either need the installation media for Windows Server 2008 HPC Edition (or another 64-bit edition of Windows Server 2008) or you must have the installation files available on a network location that is accessible from the head node computer. You can add additional drivers, such as the high-speed interconnect drivers to the operating system image, as described in the section that follows.
* Without an operating system image  
  This type of template is used to add preconfigured compute nodes to the cluster, or to update existing nodes.

The type of template that you create for the initial deployment of your HPC cluster depends on how you decided to add compute nodes to your cluster. Node templates are created from the To-Do List.

For more information about creating node templates, go to <http://technet.microsoft.com/en-us/library/cc719017.aspx>.

#### Add Drivers for the Operating Systems Images

If you are deploying compute nodes from bare metal and those nodes require special device drivers such as the InfiniBand drivers, you need to add those drivers during the configuration process of your head node. Drivers must be in the .inf format, and must be available to all operating system images in the image store. Drivers are added from the To-Do List.

As with the head node,register the NetworkDirect provider using the tool provided by your network hardware vendor; typically, this tool is ndinstall.exe. Add a Deployment Task on each compute node with the command:

C:\Windows\System32\ndinstall -i

**Note:** Add this task in **Deployment**, not the **Maintenance,** part of the node template.

#### Add or Remove Users

If you will be giving access to the cluster to other members of your organization, you need to add them as HPC cluster users or HPC cluster administrators using the To-Do List. Also, you can remove users or administrators that were added by default during installation.

### Build and Deploy on a Small Scale

You now have your operating system image and template, and you are ready to deploy the compute nodes.

Before beginning a large-scale deployment, however, a best practice is to test deploy a small number of compute nodes (first two or three compute nodes, then a rack) manually before you use the XML file.

### Add Compute Nodes to the Cluster

You are now ready to add the compute nodes to the cluster; this process must be paced for large-scale clusters because DHCP is broadcast and susceptible to errors if a large number of nodes are booted at once. (The Trivial File Transfer Protocol [TFTP] uses User Datagram Protocol [UDP], and is also susceptible errors from heavy loads.)

Table 7 Checklist for Adding Compute Nodes

|  |  |
| --- | --- |
| Task | Description |
| 1. [Add the compute nodes](#_Add_Compute_Nodes) | Although you can use several methods to add compute nodes, the most efficient method for a large-scale cluster is to use the node.xml file template. |
| 2. [Monitor progress](#_Monitor_Progress) | Monitor progress as you add compute nodes. |
| 3. [Power up the nodes](#_Power_Up_the) | Power up the compute nodes using the method you selected. |
| 4. [Flash the firmware](#_Flash_the_Firmware) | Verify the firmware version on your networking host channel adapter (HCA) card and on your network switch using tools from your vendor. |

#### Add Compute Nodes

Although you can use several methods to add compute nodes, the most efficient method for a large-scale cluster is to use the node.xml file template you created in [Configure the Naming of the New Nodes](#_Configuring_the_Naming), which contains a list of compute nodes that you want to add to your cluster.

A best practice is to deploy in batches of no more than approximately 256 nodes. Do not overlap the processes; deploy a batch only when the previous batch is Offline.

**Note:** With large numbers of nodes, it is possible that some nodes will fail. If this occurs, examine the logs. If there are no obvious issues, try to add the nodes again in the next batch.

The default in Windows HPC Server 2008 is to respond only to PXE requests that come from existing compute nodes. This default setting is automatically changed when you use the Add Node Wizard to add nodes from bare metal. Also, you can manually change this setting in the Options menu, under Deployment Settings.

Use the **Add Node** wizard: On the Select Deployment Method page, select Import compute nodes from a node XML file. Type or browse to the location of the node XML file on the Select Node XML File page, and then click Import. The nodes appear in an Offline state. Next, assign the node template you created in [Creating a Node Template](#_Creating_a_Node). Start with a small number of nodes for the first batch—between 10 and 100 nodes—and then add 256 nodes at a time.

For more information, see the “Design and Deployment Guide for Windows HPC Server 2008” (<http://go.microsoft.com/fwlink/?LinkID=117921>).

Step-by-step procedures for adding compute nodes to your cluster are available in the Windows HPC Cluster Manager Help documentation. To see these procedures, click Adding Nodes to a Cluster on the Select Deployment Method page of the Add Node Wizard.

#### Monitor Progress

During the deployment process of a compute node, its state is set to Provisioning. After the deployment process is complete, the state changes to Offline. You must bring compute nodes online before they can process jobs.

For more information about monitoring progress from the Windows HPC Cluster Manager, review the Design and Deployment Guide for Windows HPC Server 2008 published at <http://go.microsoft.com/fwlink/?LinkID=117921>.

You can monitor Windows Deployment Services progress, as shown in the figures that follow.

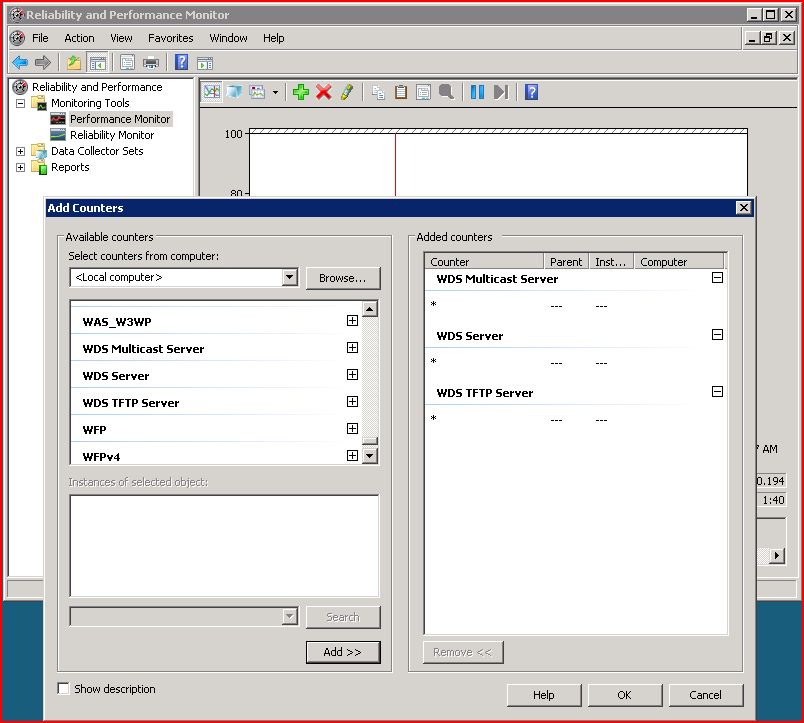


Figure 3 Monitoring Windows Deployment Services progress

In the WDS TFTP Server, selecting **Active Requests** shows the number of nodes PXE booting at the same time. In the WDS Multicast Server, selecting **Active Clients** gives the number of nodes that are multicasting.

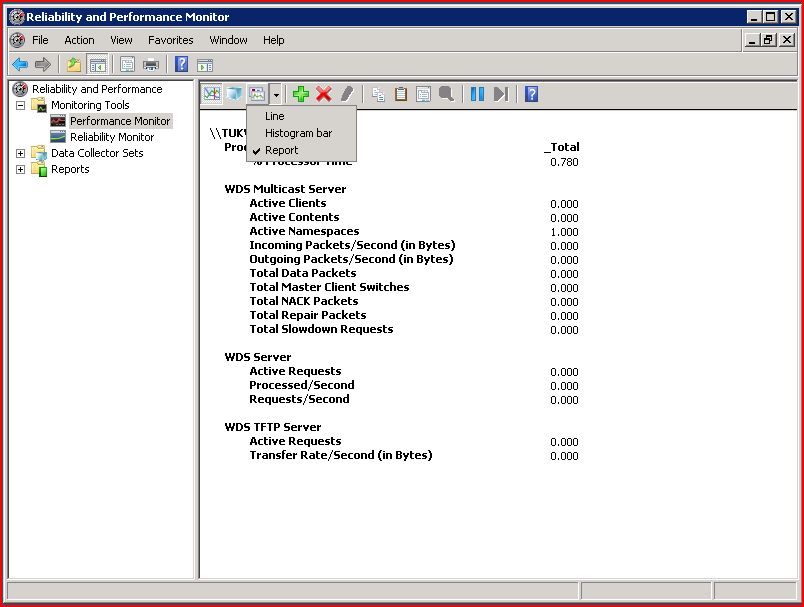


Figure 4 Report of Windows Deployment Services progress

For more information about Windows Deployment Services, go to <http://technet.microsoft.com/en-us/library/cc771670.aspx>.

#### Power Up the Nodes

Next, you must power up the nodes, using the method you selected in [Prepare for Integration of Scripted Power Tools](#_Preparing_for_Integration) (using a script such as an IPMI script).

#### Flash the Firmware

Verify the firmware version on your networking host channel adapter (HCA) card and on your network switch using tools from your vendor. For example, you can use the Mellanox Firmware Tools (MFT) for Windows. For more information, go to <http://www.mellanox.com/support/HCA_FW_update.php> and <https://docs.mellanox.com/dm/WinIB/docs/Mellanox_Firmware_Tools_for_Windows_release_notes_1_0_3.pdf>.

## Verify the Cluster

After you have configured your head node and added all compute nodes to the cluster, run diagnostic tests to validate cluster functionality and troubleshoot any configuration issues.

Because Windows HPC Server 2008 was designed for clusters with up to 256 nodes, clusters with larger numbers of nodes should run the uSane toolkit, which includes a heat map view and has been optimized for clusters of this size, in conjunction with some of the built-in diagnostics for cluster verification. See <http://windowshpc.net/Resources/Pages/Programs.aspx> for more information.

Table Checklist for Verifying the Cluster

|  |  |
| --- | --- |
| Task | Description |
| 1. [Use the built-in diagnostics](#_Built-in_Diagnostics_1) | Run the MPIPingPong test and output an XML file containing latency and throughout measurements and statistics. |
| 2. [Use the uSane Toolkit](#_uSane_Toolkit) | To identify any potential performance issues, the uSane toolkit runs a small C ping-pong test between nodes.   1. Pre-uSane Testing Start by ensuring that the compute nodes are visible, and that you can use a Run command and get results. 2. Set up uSane Copy uSane to a share that is accessible to your workstation, the head node, and the compute nodes 3. Run the uSane tests |
| 3. [Disable performance counter collection and event forwarding](#_Disable_Performance_Counter) | Disable performance counter collection and event forwarding—these capabilities can put a large performance tax on the Windows HPC Management Service and on the SQL Server database. For these clusters, a best practice is to these, starting after deploying about 500 nodes. |

### Use the Built-in Diagnostics

For large clusters, use the built-in diagnostics from the command line rather than from the Administrator Console GUI. MPIPingPong is installed on the cluster by default. Run the MPIPingPong test and output an XML file containing latency and throughout measurements and statistics with:

mpipingpong [options]  < output-filename >

For more information about MPIPingPong, go to <http://technet.microsoft.com/en-us/library/cc972824.aspx>.

### Use the uSane Toolkit

For high HPL performance, you need to have the lowest possible latency and the highest possible bandwidth. To identify any potential performance issues, the uSane toolkit runs a small C ping-pong test between nodes.

The uSane toolkit, derived from a larger suite of tests called Sanity, tests the performance of your interconnect and indicates whether the cluster is ready for benchmarking. It lets you know if the cluster has been set up correctly, and if the file share is accessible to the compute nodes. It also tests the individual links and finds the slow interconnect links, so that you can be sure that all links are running as fast as they should. While the HPL benchmark stresses all of the cores, uSane focuses on the application network.

**Note:** Because diagnostic tests on large-scale clusters can take several hours, it is a best practice to run multiple smaller bandwidth and latency tests, rather than a single large test. You can use the batch\_cband.cmd and the batch\_clate.cmd scripts to launch multiple small bandwidth and latency tests with a single user action.

#### Pre-uSane Testing

Start by ensuring that the compute nodes are visible, and that you can use a Run command and get results. Use the Windows HPC Cluster Manager GUI to run a quick connectivity test with 15–20 of the compute nodes.

**Note:** If you are using an application network that was not configured during setup in the Admin To-Do List, confirm that the MPI network interface on each node has the appropriate IP/netmask. Jobs that do not use MPI (such as running ipconfig on all the nodes, in this example) will run successfully, but this does not ensure that MPI jobs will also be successful. Manually set up the MPI NICs on the head node via the clusrun command, and then use cluscfg setenvs CCP\_MPI\_NETMASK=xxx.xxx.xxx.xxx/yyy.yyy.yyy.yyy to set the MPI traffic to go over the MPI NICs. Note that xxx.xxx.xxx.xxx/yyy.yyy.yyy.yyy is a netmask of the same form as that used by the mpiexec environment variable MPICH\_NETMASK. For more information about using environment variables, go to <http://technet.microsoft.com/en-us/library/cc720153.aspx>.

#### Set Up uSane

Next, define a share that is accessible to your workstation, the head node, and the compute nodes; you will copy the uSane directory there. All of the test scripts assume that you have a global share that can be used as a working directory—it must appear to all nodes as the same path.

Though the command shell does not support uniform naming convention (UNC) paths as working directories, you can use pushd to the UNC path; the pushd command saves the current working directory in memory (so it can be returned to at any time), maps a drive to the host, and puts you directly into the directory.

The set-base.cmd command sets the base directory of the Sanity tests and the name of the job scheduler head node. Edit sb-yoursanity.cmd as follows, noting that <yourbase> should be the uSane folder:

\\<yourhost>\<yourshare>\<yourbase>

Execute the command using sb-yoursanity.cmd.

The uSane directory is the base for all of the tests. It contains a README file, a bin directory, and some sample usages of the set-base command sb-xxx.cmd. The tests include:

* **Cmd**: A test that runs a simple command on a single CPU on a compute node (command line).
* **MpiHi**: A more robust parallel Hello World program (C Code).
* **cNodeLate**: A test that checks the MPI latency on all pairs of nodes (C code).
* **cNodeBand**: A test that checks the MPI bandwidth on all pairs of nodes (C code).
* **cCheck** (if needed): An MPI ping-pong bandwidth test that verifies the data sent is returned without transmission errors. Handy for link verification or generating both network and CPU load.
* **cFlood** (if needed): A multiple ping-pong bandwidth test that floods a switch with traffic to see if it can handle a full load. (The cFlood test is the fastest of the tests, and quickly verifies InfiniBand connectivity and bandwidth.)

#### Run the uSane Tests

The recommended order for running the tests is as follows:

##### The cmd test

Begin by running cmd, as shown in the figure that follows. This test shows if uSane is working, and shows that the head node and compute nodes can access the share you set up. Note that this is a user-level version of the Administrator console node management Run command feature.

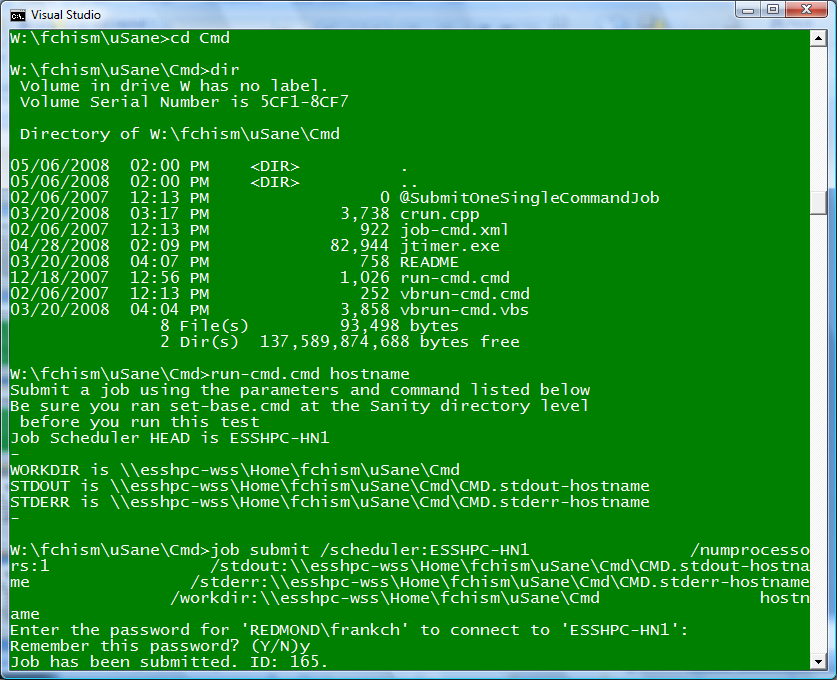


Figure 6 The CMD test

##### The MpiHi test

Next, run the MpiHi test, a parallel C Hello World test. This shows whether MPI is working. It requires the nodes to actually send messages, and tests the correctness of the MPI configuration and the ability of the cluster to compile. Note, however, that it does not test performance. Figure 6 shows the output of the test.

**Note:** One of the most common issues reported is an “eager message” error. This indicates a poor MPI network configuration or information left over from an earlier test. If this error occurs, check to make sure that the MPICH\_NETMASK is being set in the script\_MpiHi.cmd file.

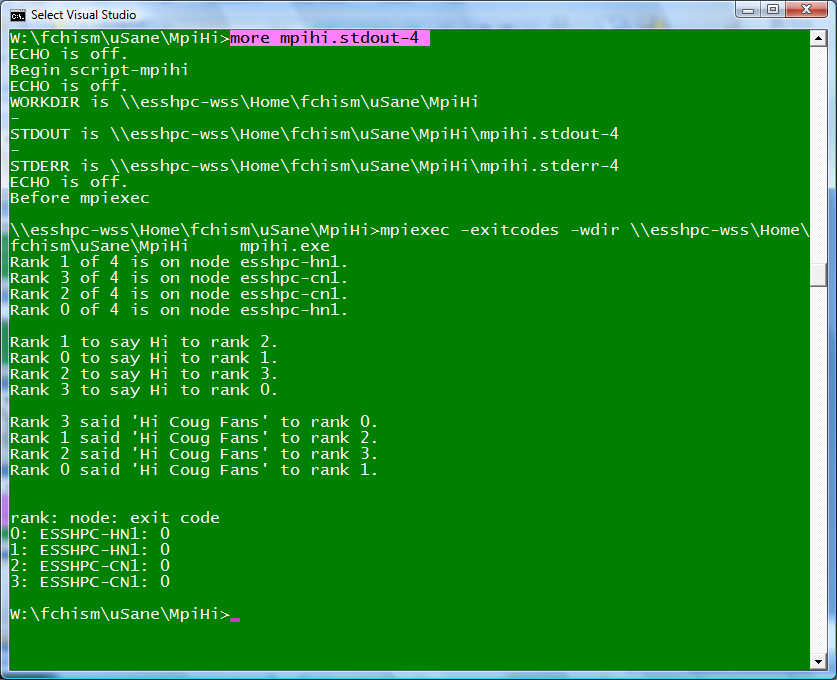


Figure 7 The MpiHi test

##### The cNodeBand test

Next, run the cNodeBand, or cband, test. This is a ping-pong bandwidth test that measures the point-to-point performance by nodes and the send/receive bandwidth (the default is 1 MB messages, which is big enough to get near peak). If necessary, compile the code from a Visual Studio command shell using nmake cband.exe.

##### The cNodeLate test

The next test to run is the cNodeLate, or cLate, test. This test measures the point-to-point node-to-node send/receive latency (using 4-byte messages). For this test, do not use SBUFFER 0. Compile the code from a Visual Studio command shell using nmake clate.exe.

##### The cFlood test

This test is a multiple ping-pong bandwidth test that floods a switch with traffic to see if it can handle full load. Run cFlood after cBand, which tests links one at a time.

The cFlood test measures the simultaneous send/receive bandwidth (using 1 MB messages) for pairs of nodes at the requested count. It runs much faster than cNodeBand and cNodeLate. Results should match cBand with a small reduction because of switch contention. Note that it is common practice is to have 2:1 blocking in layer 2; in this case there can be up to 60 percent slowdown from cBand rates.

Compile code from a Visual Studio command shell using nmake cflood.exe.

**Note:** For the cNodeBand, cNodeLate, and cFlood tests, you can get a faster turnaround with smaller repeat counts (r repeat\_count), though the smaller the count, the higher the operating system jitter. As a general rule, use 100 for InfiniBand and 10 for Gigabit Ethernet.

Note also that uSane currently includes cRank tests to measure the application-like performance. These tests have not been used as much and will likely be removed from the uSane file in the future.

##### Output Reports

Output reports can be obtained in comma separated value (CSV) format; it appears in the working directory as cBAND.Results.csv. You can copy the file to a computer running the 2007 Microsoft® Office system and summarize the results. If you view the table in Microsoft® Office Excel® 2007, you can select the numeric data and click on **Conditional Formatting**. Results should resemble the figure that follows.

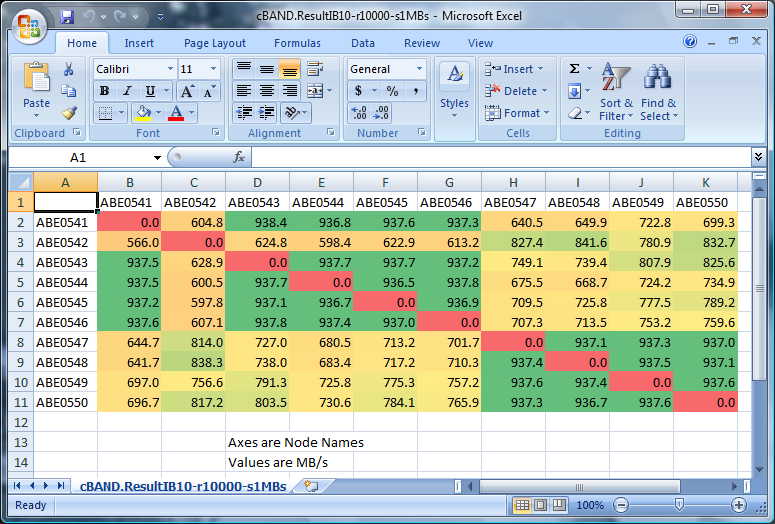


Figure 8 Output results for uSane tests

In the example, the red diagonal shows results between a node and itself. In this example, nodes ABE0541, ABE0543, ABE0544, ABE0545, and ABE0546 are on the same layer 1 switch, and nodes ABE0547, ABE0548, ABE0549, and ABE0550 are on the same layer 1 switch—these show as green. The two groups, however, are not on the same layer 1 switch—this shows as yellow. Node ABE0542, with no green at all, is a replacement node that is not on either of the above switches.

Table 9 shows the expected measured bandwidths, or throughput, and measured latencies. If your throughput is lower than the expected value, or if your measured latency is higher, further investigation is required.

Table 9 Expected Bandwidths and Latencies

|  |  |  |
| --- | --- | --- |
| Network | Measured Bandwidth (MB/Sec) | Measured Latency (µsec) |
| InfiniBand, ConnectX, Double Data Rate (DDR), NetworkDirect | 1500–1900 | 2–3 |
| InfiniBand, InfiniHost, DDR, NetworkDirect | 1250–1500 | 5 |
| InfiniBand, InfiniHost, Single Data rate (SDR), NetworkDirect | 850–1000 | 6–7 |
| InfiniBand, InfiniHost, TCP/IP over IB (IPoIB) | 150–180 | 25–35 |
| InfiniBand, WindsockDirect (WSD), SDR, | 800–950 | 10–14 |
| Gigabit Ethernet | 95–105 | 50–70 |

**Note:** With Gigabit Ethernet, if your measured bandwidth values are below 95 MB/sec and if your application is bandwidth sensitive, you may need to change the settings on your interface card (the performance features may not be turned on).

## Disable Performance Counter Collection and Event Forwarding

On very large clusters, performance counter and event collection can put a large performance tax on the Windows HPC Management Service and on the SQL Server database. For these clusters, a best practice is to disable these capabilities, starting after deploying about 500 nodes.

From the cluster pack, launch the HPC PowerShell window and turn off the performance counter collection and event forwarding:

Set-hpcclusterproperty –ForwardEvents 0

Set-hpcclusterproperty –CollectCounters 0

For more information on using HPC PowerShell, go to <http://technet.microsoft.com/en-us/library/cc947594.aspx>.

**Note:** You lose certain features when you disable the performance counter collection and event forwarding. Substitute the heat map found in the TOP500 toolpack for monitoring capabilities.

## Summary of Considerations Unique to Large-Scale Clusters

Following is a summary of the special provisions that must be made for clusters with more than 256 nodes.

* Set up separate infrastructure nodes—put as few services as you can on each one. These nodes comprise DNS and DHCP, in addition to the domain controller and the subnet manager. Both DNS and DHCP should be based on the Windows operating system.
* The private network should be truly private. Verify that there are no other PXE or DHCP servers on the network.
* For clusters with more than 128 compute nodes, run SQL Server Standard. The SQL Server database file should not be on the system drive, and the log files should be on separate spindles.
* Turn off Event Forwarding and disable Performance Counter collection.
* Use a Node.XML file to deploy the compute nodes.
* Deploy in batches of no more than approximately 256 nodes, and do not overlap the deployment processes (deploy a second batch only when the first one is **offline**).
* Use the MPIPingPong built-in diagnostic and the uSane toolkit (uSane has been optimized for larger-scale clusters and includes a heat map view).
* With uSane, run multiple smaller bandwidth and latency tests, rather than a single large test.

# Benchmark a Windows HPC Server 2008—Based Cluster

Running the HPL benchmark is a great “burn in” run for a cluster. Even if the results of an HPL run are not submitted to the TOP500 List, the run provides useful information about the cluster’s performance. Note that it is far preferable for a newly installed Windows-based cluster to be tested with Windows rather than with Linux, as is often the case; a Windows run can provide information that is more relevant in predicting how the cluster will perform as a Windows cluster in production.

The Interactive Computing Laboratory at the University of Tennessee has been distributing the HPL benchmark for many years; it has been used to benchmark supercomputing systems of all sizes and types. It can be used on large single-box symmetric multiprocessing (SMP) systems and on clusters of independent computers tied together with a high-speed interconnect such as Gigabit Ethernet, Myrinet, or InfiniBand.

You can use a stock distribution of the LINPACK benchmark and compile it for use as a native Win64 binary linked with the native MSMPI implementation that is part of the Windows HPC Server 2008 and the native ACML or MKL BLAS libraries.

Alternately, you can use the LINPACK Tuning Wizard; this is described in the section [Use the LINPACK Tuning Wizard](#_Using_the_LINPACK).

## Prepare for an HPL Run

The following checklist provides the steps for preparing for the HPL run.

Table 10 Checklist for Preparing for the HPL Run

|  |  |
| --- | --- |
| Task | Description |
| 1. [Download the HPL benchmark](#_Downloading_the_HPL) | Download the benchmark from Netlib into a shared partition. |
| 2. [Install the necessary software](#_Installing_the_Necessary) | Install the software that is necessary to run the benchmark. |
| 3. [Create and configure the Makefile and Build the Benchmark](#_Creating_and_Configuring) | Create the Makefile and edit it so that it reflects your particular cluster configuration. A single command builds the HPL benchmark. |

### Download the HPL Benchmark

Before beginning the benchmark, you want to locate a disk partition that is shared on all of the nodes of your cluster.

Download the benchmark (the file package is named **hpl.tar.gz**) into this shared partition. The benchmark is available at <http://www.netlib.org/benchmark/hpl>.

You need WinZip or a Windows version of gzip to open the files.

For more information about gzip for Windows, go to <http://gnuwin32.sourceforge.net/packages/gzip.htm>.

For more information about WinZip, go to <http://www.winzip.com/index.htm>.

### Install the Necessary Software

Start by installing the following software:

* **Microsoft Visual Studio 2008**  
  Ensure that the 64-bit versions of the SDK (tools and libraries) are installed.
* **Microsoft HPC Pack 2008 SDK**  
  This can be obtained from [http://go.microsoft.com/fwlink/?LinkId=133271](%20http://go.microsoft.com/fwlink/?LinkId=133271), and should be installed to C:\Program Files\Microsoft HPC Pack 2008 SDK\.
* **Machine-specific optimized Basic Linear Algebra Subprogram (BLAS) libraries**  
  Download from [http://www.netlib.org/blas/faq.html](http://www.netlib.org/blas/faq.html%20).
* **The AMD Core Math Library (ACML) or the Intel Math Kernel Library (MKL)**.

A complete list of the available libraries, including ATLAS and GOTO, is available at [http://web.njit.edu/all\_topics/HPC/basement/libraries/libraries.html#goto](http://web.njit.edu/all_topics/HPC/basement/libraries/libraries.html%23goto).

ACML can be obtained from <http://developer.amd.com/cpu/libraries/acml/Pages/default.aspx>.

MKL can be obtained from <http://support.intel.com/support/performancetools/libraries/mkl/sb/CS-028417.htm>.

**Note:** The MKL comes with pre-compiled HPL binaries for 32-bit and 64-bit platforms. These are also included in the Intel Cluster Toolkit. Therefore Visual Studio, the Microsoft HPC Pack SDK, and BLAS are not required if the MKL will be installed.

### Create and Configure the Makefile and Build the Benchmark

Next, create the Makefile and edit it so that it reflects your particular cluster configuration. You can then build the benchmark.

For more detailed instructions, see: <http://windowshpc.net/Resources/Pages/Programs.aspx>.

## Run HPL on the Cluster

Because executing a job on a large cluster can take a long time, a best practice is to run a small test job using a small problem size (N) first. This gives you a good estimation of the first set of HPL parameters. You can then tune the parameters—manually, with the Excel add-in, or using the LINPACK Tuning Wizard—to enhance the performance.

Note that there are four basic ways to increase LINPACK performance:

* Add more nodes
* Add more memory
* Add a faster interconnect
* Modify the HPL parameters PQ, N, and NB (described in the sections that follow)

There are several ways to tune the HPL parameters. The LINPACK sizing sheet lets you easily choose values, see the results, then re-run with different values. The Excel add-in provides a slightly more complex tuning tool. Another option, the LINPACK Tuning Wizard, automatically steps through the tuning procedure for you. When available, it provides the easiest optimizing method.

The following checklist describes the steps for running HPL on your cluster.

Table Checklist for Running HPL

|  |  |
| --- | --- |
| Task | Description |
| 1. [Assign the HPL parameters](#_Assigning_the_HPL) | Become familiar with the HPL parameters. These will be tuned to enhance performance. |
| 2. [Run a test job](#_Running_a_Test) | Run a small test job on the cluster to verify operation using the Excel LINPACK sizing sheet. |
| 3. [Review the results](#_Reviewing_the_Results) | Review the resulting output file. |
| 4. [Tune the parameters](#_Tuning_the_HPL) | Tune the HPL parameters to optimize performance. |
| 5. [Turn off non-essential services (optional)](#_Turning_Off_Non-Essential) | Optimize performance further by turning off all non-essential services. |
| 6. [Use the Excel add-in (optional)](#_Using_the_Excel_1) | Use the Excel add-in for tuning. You can also use the add-in as an interface to LINPACK. |

### Assign the HPL Parameters

There are 29 input parameters that you can use and tune for HPL; go to <http://www.netlib.org/benchmark/hpl/tuning.html> or [Appendix B](#_Appendix_B:_The) for more detail. [Appendix C](#_Appendix_C:_Additional) provides additional information about the HPL problem.

The four main parameters are: N, NB, P, and Q. You set and tune these parameters to optimize the HPL run.

* **N**, the rank of the matrix (or the problem size), is the most important variable. The higher the number, the more 64-bit floating point arithmetic (adds and multiplies in particular) gets executed by the benchmark. N is limited by the amount of memory that the system can deliver to the HPL executable before paging (swapping to disk)—for Windows HPC Server 2008, somewhere between 0.83 and 0.86 of the physical memory is optimal (note that the operating system should occupy its minimal memory footprint). A value of 12,000 works well for 2-GB systems (26,000 for four nodes with 2 GB each).

Start by making N small to get a quicker run, and then increase the value. When the problem gets too big, it no longer fits into the physical memory and the system starts paging, drastically reducing performance.

Note that you can run multiple problem sizes with a single run by putting several numbers separated by spaces on the same line. You need to set the value on the previous line to the number of problem sizes (Ns) you want to explore.

Remember that N should be the total number of processors you want to use, and should be equal or greater than 𝑃∗𝑄 in the HPL.dat file.

* **P and Q** are factors of the total core count. Conventional wisdom dictates that they should be as close to “square” as possible, but occasionally a long and thin configuration gives better results. P and Q are linked: 𝑃∗𝑄=*Number of Processes*. You can explore different combinations of 𝑃∗𝑄 and set the value of the previous line to the number of combinations you are working with.

For example, you can set the value of P to the number of processors (cores) you have on each node, and the value of Q to the number of nodes. Note that hyperthreading should be disabled in the BIOS.

* **NB**, the most widely tuned parameter, represents the number of blocks used to split the problem. The choice of NB varies; some prefer to start with a small value like 32 or 40, while other respected authorities prefer a much higher value. NB and N are linked, so aim for 𝑁 𝑚𝑜𝑑 𝑁𝐵*=0* to avoid remainder computations.

NB controls the amount of data distributed on each node. From a data distribution perspective, the smaller the value for NB, the better the load balance between processors. However, if you pick too small of a value for NB, performance may decrease because of an increase in the number of messages between processors.

Following is an input file (HPL.dat):

HPLinpack benchmark input file

Innovative Computing Laboratory, University of Tennessee

HPL.out output file name (if any)

6 device out (6=stdout,7=stderr,file)

1 # of problems sizes (N)

268608 Ns

1 # of NBs

192 NBs

0 PMAP process mapping (0=Row-,1=Column-major)

1 # of process grids (P x Q) (1xN works best on Eth hub)

42 Ps

128 Qs

16.00 threshold

1 # of panel fact

1 PFACTs (0=left, 1=Crout, 2=Right)

1 # of recursive stopping criterium

4 NBMINs (>= 1)

1 # of panels in recursion

2 NDIVs

1 # of recursive panel fact.

2 RFACTs (0=left, 1=Crout, 2=Right)

1 # of broadcast

1 BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)

1 # of lookahead depth

0 DEPTHs (>=0)

2 SWAP (0=bin-exch,1=long,2=mix)

192 swapping threshold

0 L1 in (0=transposed,1=no-transposed) form

0 U in (0=transposed,1=no-transposed) form

1 Equilibration (0=no,1=yes)

16 memory alignment in double (> 0)

### Run a Test Job Using the Excel LINPACK Sizing Sheet

You can use the Excel LINPACK sizing sheet to simplify the benchmarking process; an example is shown in Figure 9. (See <http://windowshpc.net/Resources/Pages/Programs.aspx>.) Fill in the highlighted fields; they provide the spreadsheet with information about the cluster. The rest of the values are calculated for you, and you can use them for the HPL job.

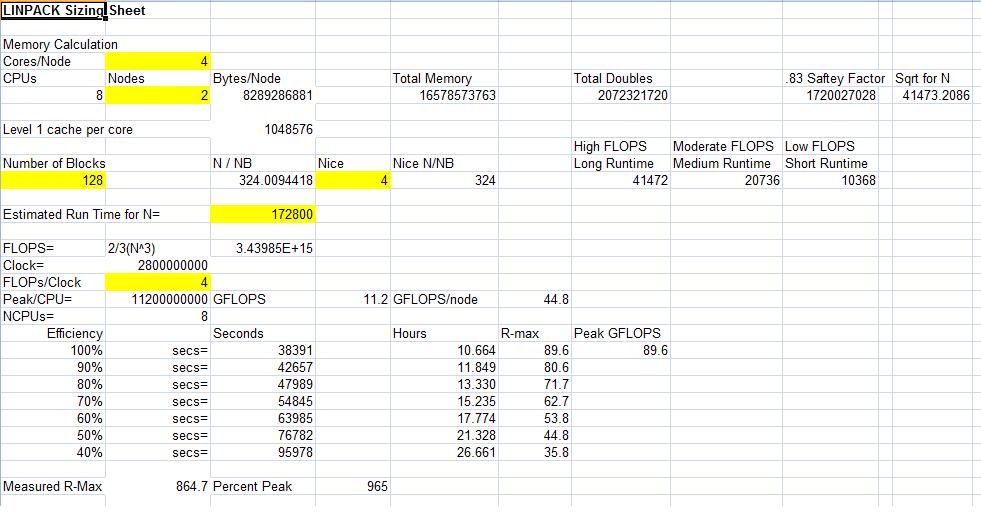


Figure 9 LINPACK sizing sheet example

To view the status of your submitted job, launch the **Job Manager** from the **Start** menu. After the GUI has loaded, find your job’s ID in the top middle pane. Click on the job to select it; this displays more detailed information in the lower middle pane of the GUI.

Once submitted, simply wait for the test to complete. This may take a couple of hours or minutes, depending on the problem size. If the job failed, you can select the job from the list and view any error messages in the lower window.

If the benchmark runs successfully, then you see results at the end of the file where the benchmark writes its output (\\headnode\NativeHPL\hpl), as in the figure that follows. For example:



Figure 10 Output results

### Review the Results

Below is a summary of pertinent information generated by the HPL benchmark and reported in the Top 500 List.

* **Rmax**: The performance in Gflop/s for the largest problem run on your system.
* **Nmax**: The size of the largest problem run on your system.
* **Rpeak**: The theoretical peak performance GFlop/s for the computer.
* **#Proc**: The number of processors used in the benchmark run.

The output results also show T/V (the wall clock time/encoded variant), time (time in seconds to solve the linear system), and gigaflops (the rate of execution for solving the linear system).

**Note:** It is *critical* that the equations are solved correctly. This is indicated by the “PASSED” result on the right of the output.

With large N values, it may take some time to see the result appear. For N=26,000, for example, the set of equations may take close to 10 minutes to solve. When running on a single processor (recommended for the first run to set a baseline), the resulting GFlop value should be roughly equal to or greater than the clock speed of a processor (3GHz → 3 GFlop).

When scaling across the compute nodes, there is a roughly linear drop-off in total performance from a theoretical max (N procs \* gigaflops per processor) because of the interconnect latency. For very rough calibration, on a GigE interconnect, there is approximately 85 percent of maximum value when running across two nodes each with two processors.

Next, you can then start refining your parameters to find the best result for your platform.

### Tune the HPL Parameters

Remember that the HPL benchmark solves a dense system of linear equations based on 64-bit floating point arithmetic. The HPL problem size is allowed to vary, and the best floating-point execution rate (measured in GFlop/s) is reported in the TOP500 List. Therefore, most tuning of the input data is related to the problem size. Optimizing Nmax and Rmax requires some experimenting with the problem size specified in the **HPL.dat** file, located in the executable directory.

### Turn Off Non-Essential Services

To ensure a minimal hardware footprint from the operating system and to further optimize the run, any unnecessary operating system services should be disabled; in particular, disable any desktop-oriented features that may have been enabled.

It is even possible to go a step further to “stunt-mode” Windows—turn off all services that could slow the cluster down, such as management, scheduled tasks, and others.

**Note:** The “stunt-mode” is an optional configuration used only for the benchmarking run, and great care must be taken so that you do not lose control of the compute nodes.

The following list shows the non-essential services that were turned off in some of the Microsoft TOP500 cluster installations:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | sc stop wuauserv | | sc stop WinRM | | sc stop WinHttpAutoProxySvc | | sc stop WAS | | sc stop W32Time | | sc stop TrkWks | | sc stop SstpSvc | | sc stop Spooler | | sc stop ShellHWDetection | | sc stop RemoteRegistry | | sc stop RasMan | | sc stop NlaSvc | | sc stop NetTcpActivator | | sc stop NetTcpPortSharing | | sc stop netprofm | | sc stop NetPipeActivator | | sc stop MSDTC | | sc stop KtmRm | | sc stop KeyIso | | rem sc stop gpsvc | | sc stop bfe | | sc stop CryptSvc | | sc stop BITS | | sc stop AudioSrv | | sc stop SharedAccess | | sc stop SENS | | sc stop EventSystem | | sc stop PolicyAgent | | sc stop AeLookupSvc | | sc stop WerSvc | | sc stop hkmsvc | | sc stop UmRdpService | | sc stop MpsSvc | | sc config wuauserv start= disabled | | sc config WinRM start= disabled | | sc config WinHttpAutoProxySvc start= disabled | | sc config WAS start= disabled | | sc config W32Time start= disabled | | sc config TrkWks start= disabled | | sc config SstpSvc start= disabled | | sc config Spooler start= disabled | | sc config ShellHWDetection start= disabled | | sc config RemoteRegistry start= disabled | | sc config RasMan start= disabled | | sc config NlaSvc start= disabled | | sc config NetTcpActivator start= disabled | | sc config NetTcpPortSharing start= disabled | | sc config netprofm start= disabled | | sc config NetPipeActivator start= disabled | | sc config MSDTC start= disabled | | sc config KtmRm start= disabled | | sc config KeyIso start= disabled | | rem sc config gpsvc start= disabled | | sc config bfe start= disabled | | sc config CryptSvc start= disabled | | sc config BITS start= disabled | | sc config AudioSrv start= disabled | | sc config SharedAccess start= disabled | | sc config SENS start= disabled | | sc config EventSystem start= disabled | | sc config PolicyAgent start= disabled | | sc config AeLookupSvc start= disabled | | sc config WerSvc start= disabled | | sc config hkmsvc start= disabled | | sc config UmRdpService start= disabled | | sc config MpsSvc start= disabled | |

### Use the Excel Add-In

A more complex planning and tuning tool is the Excel workbook companion (see <http://windowshpc.net/Resources/Pages/Programs.aspx>), which can help in computing and managing the HPL parameters. It contains several worksheets to help size and manage the HPL jobs.

Ensure that the Microsoft® .NET Framework programming extensibility for Microsoft Office Excel 2007 is set up when installing Excel (it is set up by default). Install the add-in by running setup.exe. Open Linpack.xlsx and accept the add-in to create a new document pane on the right side of the window, as shown in the following figure.

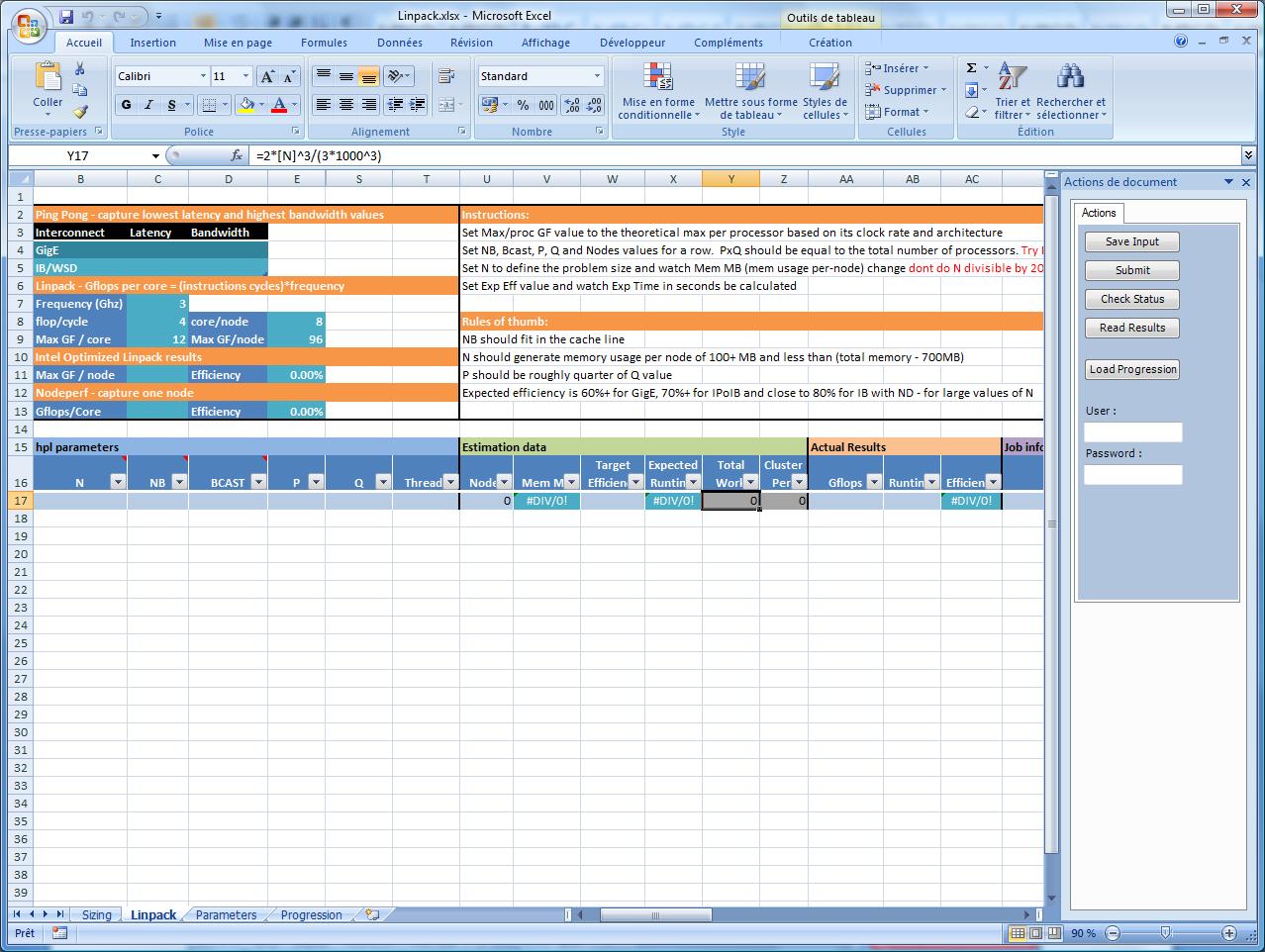


Figure 11 Add-In view

#### Directory Structure on Head Node

You need to set up a share on the head node to receive and store all of the files created by the add-in and the HPL jobs.

You may also use clusrun to deploy the \\headnode\scratch\Linpack\bin directory locally on each node in C:\Linpack\bin.

Table Directory Structure on the Head Node

|  |  |
| --- | --- |
| Directory | Description |
| \scratch\Linpack | The root directory for all tests. The scratch directory must be shared so that the compute nodes can access all of the LINPACK binaries and so the compute nodes can write their output to this location. |
| \scratch\Linpack\bin | The location of the HPL binaries. |
| \scratch\Linpack\input | The input files for the LINPACK parameters. These files are created by the Excel 2007 add-in, and follow the naming convention <hpl\_date\_time>.dat. |
| \scratch\Linpack\output | The LINPACK output files, named <jobid>.log and <jobid>.err. |
| \scratch\Linpack\jobmodels | These templates must be customized for the cluster. |

#### Use the Excel Add-In to Tune the Parameters

The Excel add-in contains several worksheets to help size and tune HPL jobs.

##### The Sizing Worksheet

The first worksheet is the Sizing Worksheet, shown in Figure 12. This worksheet lets you optimize the N and NB input parameters based on your cluster configuration.

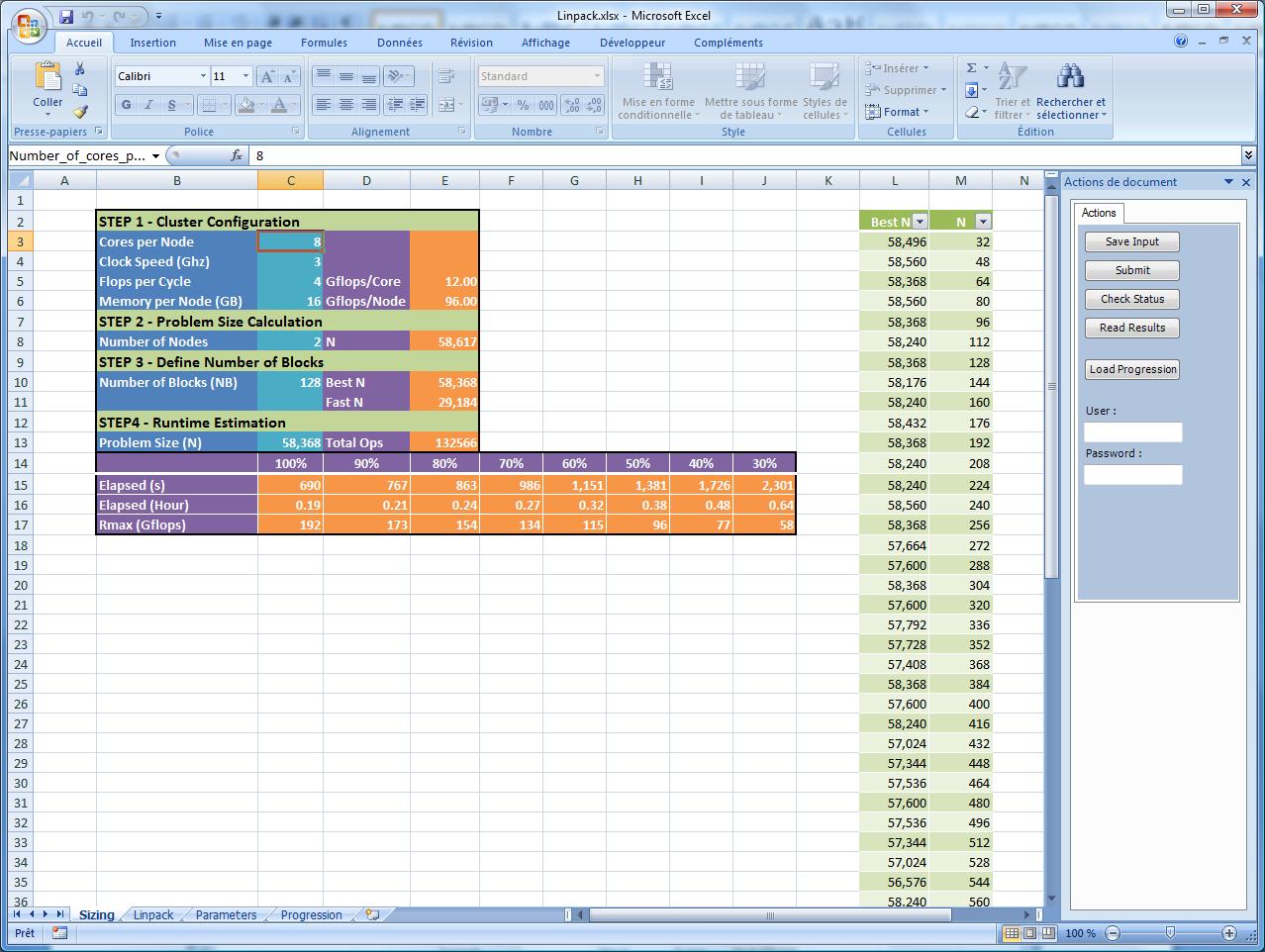


Figure 12 The Sizing Worksheet

Enter the following values for the cluster configuration in the worksheet:

* Number of cores per node
* Clock speed (in GHz)
* Memory per node (in GB)
* Number of flops per cycle delivered by the processor. The number of flops per clock is 2 for the AMD Opteron™ processor, 2 for the Xeon processor, 4 for Xeon dual-core processors, and 4 for Xeon quad-core processors. This first step computes the maximum gigaflops per core and per node.

Next, set the number of nodes for which you want to compute the N value. The formula used to compute N assumes that 80% of the node memory is allocated to the HPL processes.

Choose a value for the number of blocks used to split the problem size. The “Best N” value is 𝑁 𝑚𝑜𝑑 𝑁𝐵 *= 0*. “Fast N” is half of the “Best N.” To help you in choosing future NB values, the table on the right of the worksheet lists all the “Best N” values based on a range of NB from 32 to 1,024.

Finally, enter the value you have chosen for N. This computes the elapsed time in seconds and hours, in addition to the Rmax (the LINPACK result) value for a range of efficiencies from 100 percent to 30 percent.

##### The Linpack Worksheet

This worksheet keeps track of the different HPL input parameters and helps determine the best parameter set, as shown in Figure 13.

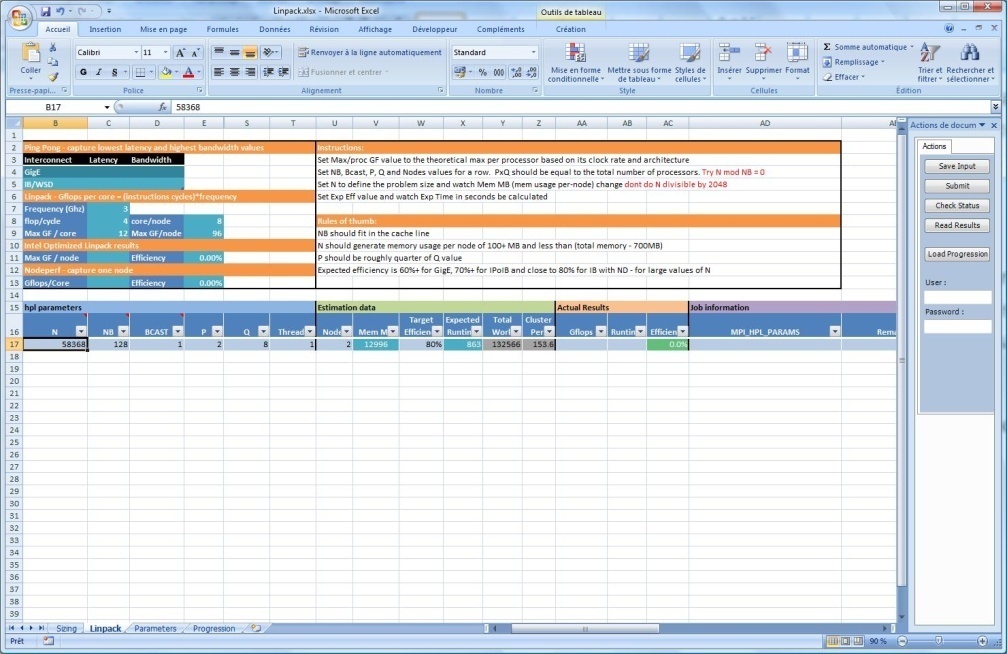


Figure 13 The LINPACK Worksheet

This sheet is organized in five main areas:

* **Cluster configuration** is a summary of the values entered in the Sizing Worksheet. You can use this area to store specific values related to the ping-pong test for latency and bandwidth, in addition to the results you can get with tools such as *nodeperf* from Intel.
* **HPL input parameters** contains the 29 input parameters. For clarity, only the five main parameters are displayed; the others are in hidden columns. Enter the N and NB values estimated in the Sizing worksheet. BCAST can be either 0 or 1 as a starting point, then enter P and Q with values that match the number of nodes (P\*Q=cores). Once you have entered P and Q, the (nodes) column is calculated for you.
* **Estimated results** computes the number of nodes that match P\*Q, in addition to the memory size required on each node. If you enter a target efficiency, then the expected runtime of the job is calculated. The Gigaflops and Efficiency columns are formatted with the new conditional formatting feature for clearer exposition (greener is better).
* **Actual results** is used to store the two values provided by the Linpack run, the gigaflops, and elapsed values. The efficiency is then calculated based on the cluster configuration.
* **Job information** is used to store specific job information, such as the job identifier, the MPI parameters used, or helpful remarks.

##### The Parameters Worksheet

Enter the name of the job template to be used, the cluster name, and the requested nodes (leave blank if you do not want specific nodes used). There are two job models: Linpack.xml and LinpackLocal.xml. See Figure 14 for an example.

* Linpack.xml must be used when running HPL from a share; it cannot be used when running multiple HPL jobs together.
* LinpackLocal.xml must be used when HPL is deployed locally on each node; it can be used simultaneously with other HPL jobs.

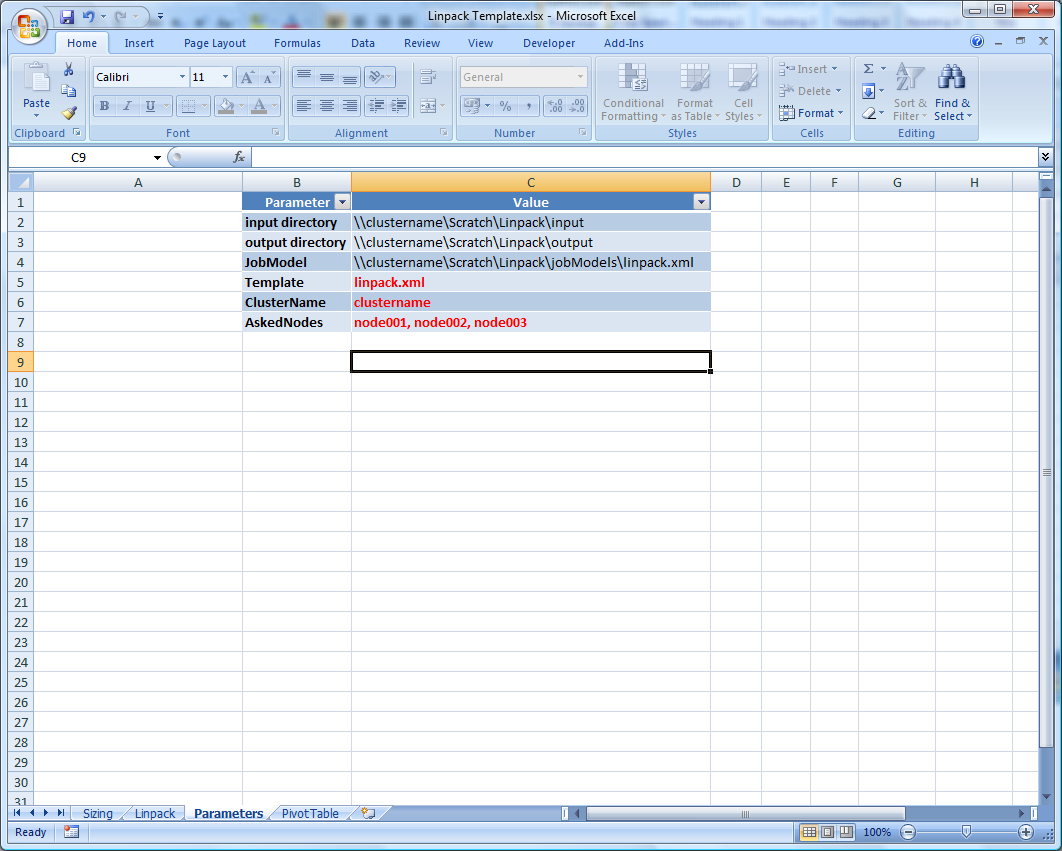


Figure 14 Parameters Spreadsheet

#### Use the Excel Add-In to Submit HPL

An alternate method for submitting the HPL job is through use of the Excel add-in.

From the sizing tools, you should be able to estimate a problem size to run for the first job. If you are using Excel to submit HPL, you can enter the parameter values in the spreadsheet and submit the job from there.

First, estimate the problem size. Next, enter the HPL parameters: N, NB, BCAST, P, Q, and Threads. In the right panel, click the **Submit** button; this creates a text file that uses the naming convention ***hpl\_DD-MM-YYYY\_HH\_MM\_ss.dat*** in the input directory and submits the job to the scheduler. In the **Target** **Efficiency** column, provide an estimated efficiency; the estimated job duration is then calculated. This value is not mandatory, but it can be useful for long jobs.

Once the job completes, the results are automatically read from the job output file and used to populate the **Gflops** and **Runtime** columns, as shown in the following figure. The conditional formatting can help you quickly identify the best results, as shown in Figure 15.

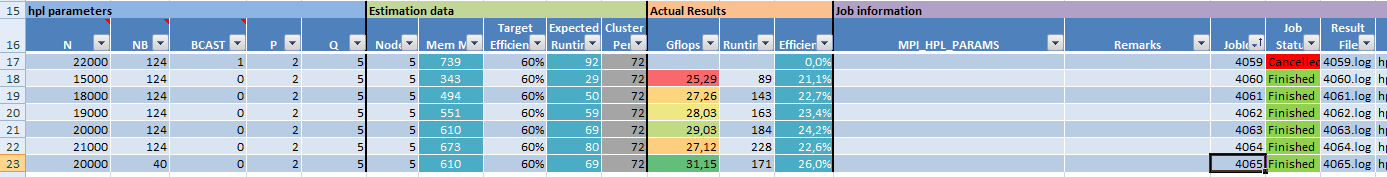


Figure 15 Results

## Use the LINPACK Tuning Wizard

The LINPACK Tuning Wizard automates many of the procedures described in the previous sections, including validating the cluster and optimizing the LINPACK benchmark (see <http://windowshpc.net/Resources/Pages/Programs.aspx>). This makes it possible for those unfamiliar with LINPACK to successfully run the benchmark on a cluster.

The LINPACK Tuning Wizard runs the LINPACK benchmark many times to tune it for optimal results. During the process, the system is stressed to the highest level allowed by the network interconnection, providing an excellent “burn-in” for the cluster.

**Note:** Because it runs multiple tests, the wizard may take several hours to complete the testing; it is thus recommended that you run the wizard overnight.

The LINPACK Tuning Wizard comprises three main sections: Configuration (which includes Network Setup, Node Selection, Initial Parameters, and Tuning Configuration), Tuning Process, and Results.

In the Configuration section, begin by choosing to use either express tuning or advanced tuning. With the express tuning option, the wizard handles all of the choices (runs tests on all of the compute nodes, runs all of the tests, defaults to the MPI network, and looks at the current directory for the LINPACK test), and then initiates the benchmarking runs.

The advanced tuning option lets you adjust all settings from the default values; it should be used by those familiar with LINPACK tuning. The screenshots that follow illustrate this tuning mode.

The first choice is the network setup, as shown in Figure 16. Specify the head node address, the specific network, and the version of LINPACK you want to run; this LINPACK binary build must be accessible to the compute nodes by means of a network share.

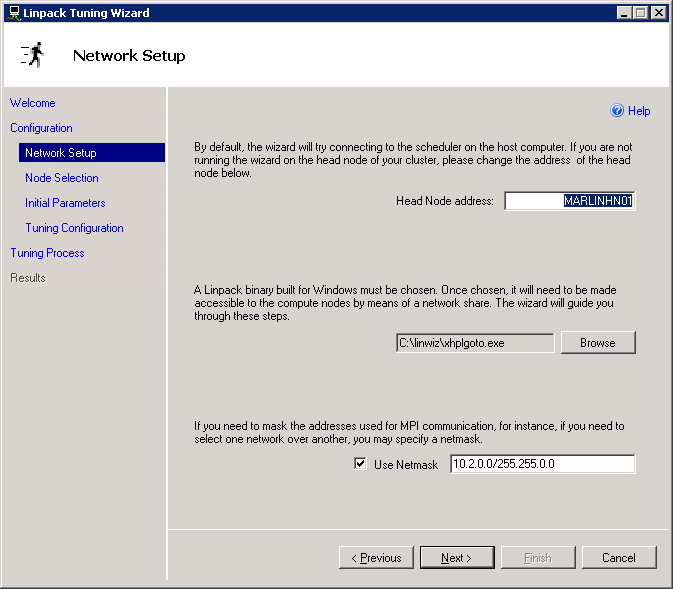


Figure 16 Network Setup page

Next, choose the compute nodes you want to use from the list the wizard compiles. In general, the more compute nodes you select, the higher the performance you can obtain from the cluster. However, if you have non-performing or weak nodes, opt to deselect them on this page.

The next page lets you select the initial LINPACK parameters or import an existing HPL.dat file to seed the system (this option is useful if you want to perform only minor tuning around a known good input). The wizard selects default values, as shown in Figure 17. The default N value is based on the number of nodes and the amount of memory in the system. The default P and Q values are based on the number of cores. Other parameter values are based on actual TOP500 runs performed by Microsoft.

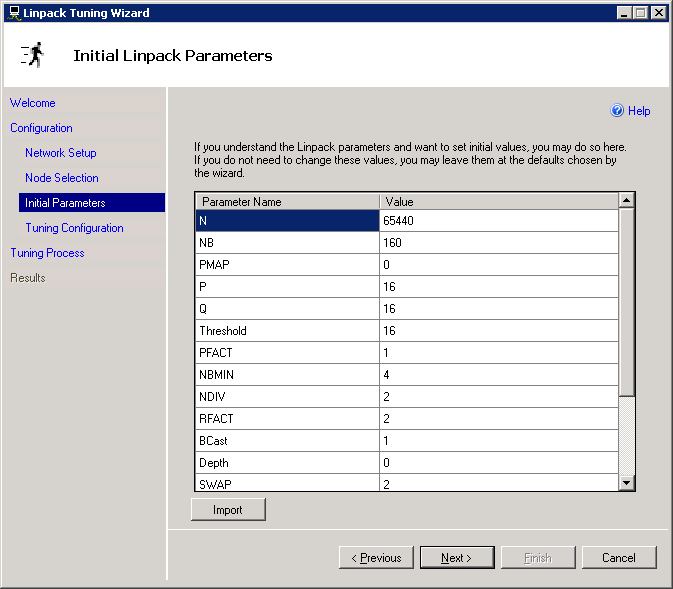


Figure 17 Initial LINPACK parameters

The final page in the **Configuration** section of the wizard lets you select which tuning processes to run, as shown in Figure 18. A best practice is to start with an integrity test and a consistency check—these tests validate the cluster. Other tests optimize the parameter values for the benchmarking run; a description is provided for each test.

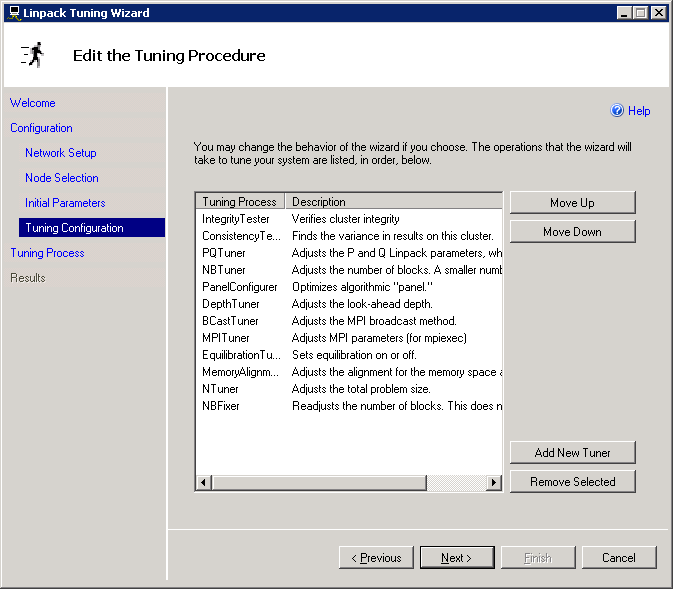


Figure 18 Tuning procedures

The integrity tester is designed to check for underperforming compute nodes that may affect cluster performance. The test runs two parallel jobs, each on half of the nodes, and then compares the results. If there is inconsistency, the test is repeated on smaller groups of nodes until the weak nodes are located. Figure 19 shows the results of an integrity test; a heat map clearly shows which nodes require investigation.

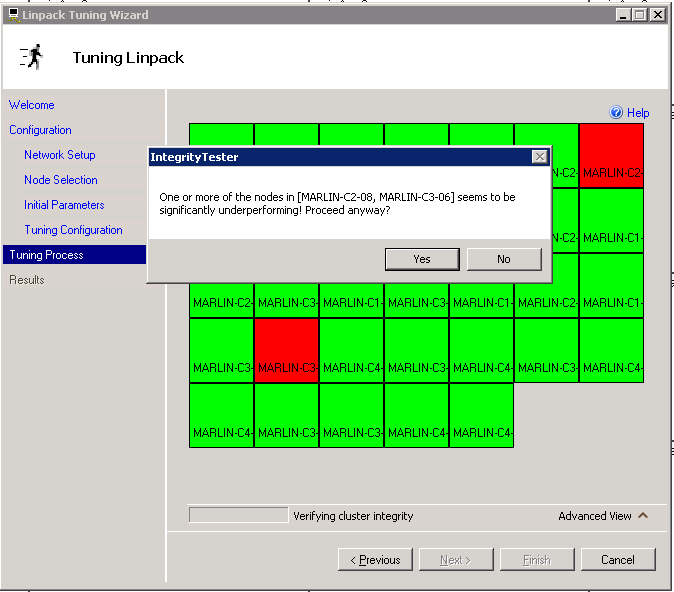


Figure 19 Integrity tester

Next, the consistency tester runs five small LINPACK tests to confirm that the cluster is performing consistently, as shown in Figure 20. The blue squares indicate that the test was completed; a pink square shows that the test is currently running. The green bar on the right shows that the results are acceptable for the completed tests.

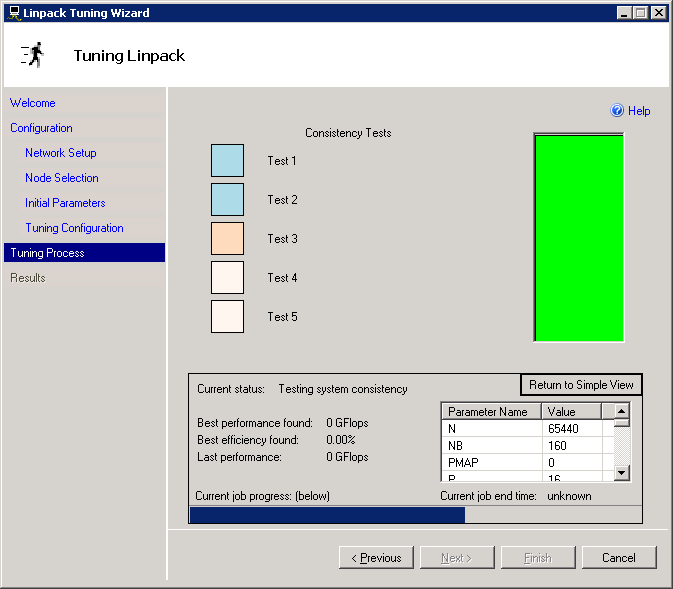


Figure 20 Consistency test

The Tuning Process page shows the tuning progress. If desired, enter a goal line (right-click and selecting a value) and observe how closely the results conform (this line appears as green in   
Figure 21). This page also keeps track of the best run results and efficiency, and of the current parameter choices.

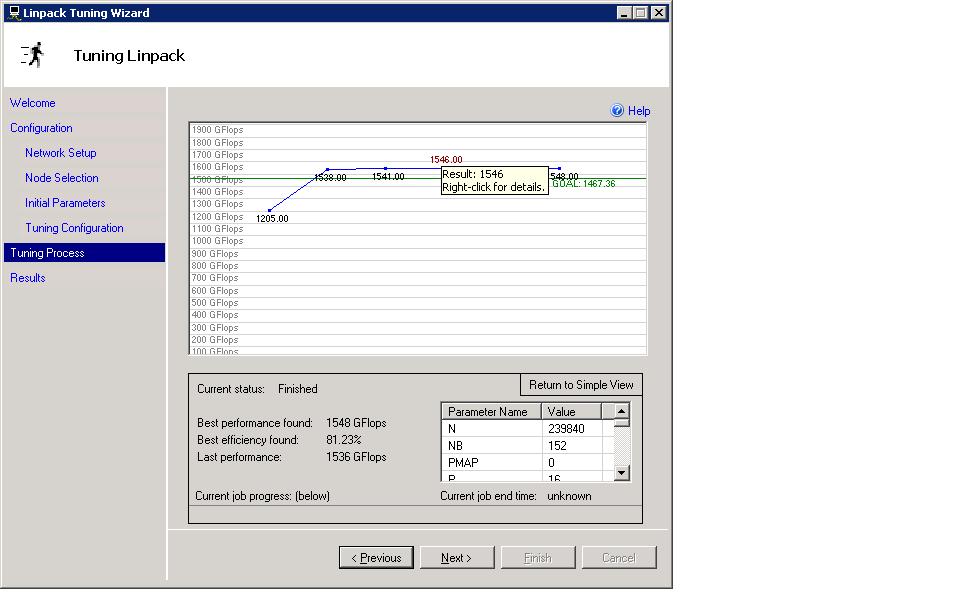


Figure 21 Tuning Process page

The final Results page is displayed when the wizard has completed the tuning procedure, as shown in Figure 22.

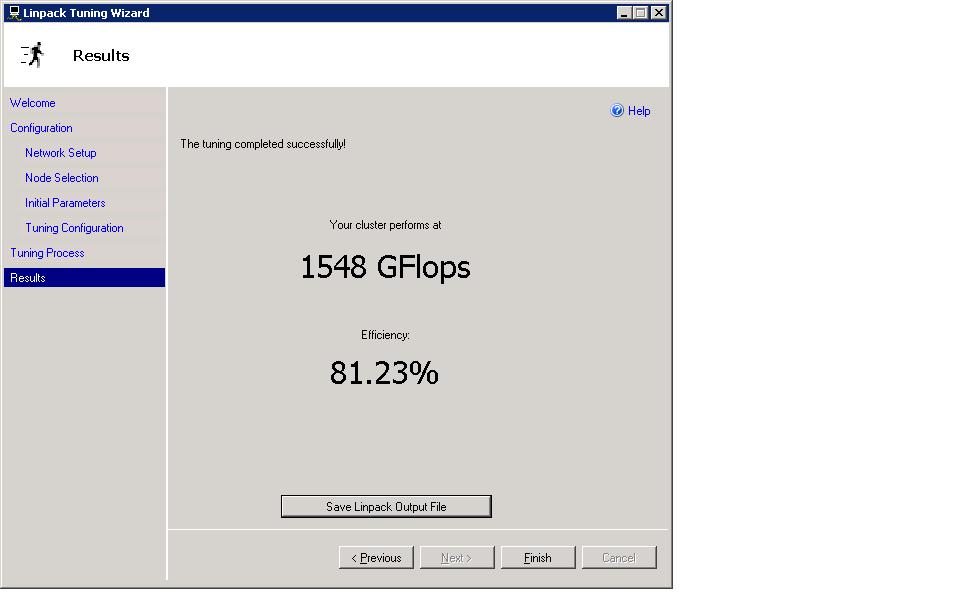


Figure 22 Results page

# Submit the Results

The first TOP500 List is released each year in June at the International Supercomputer Conference, in Dresden, Germany. The deadline for submitting entries is April 15th, and all reported systems have to be installed by May 15th. The second TOP500 List is released in November, in time for the IEEE Supercomputer Conference in the United States. The deadline for submitting entries is October 1st, and all systems reported have to be installed by November 1st. It is very important to meet the deadlines to be considered for the list. Note that rankings are not carried from list to list; you must resubmit for every list.

Note that the first submission is likely to be the highest a system will achieve, without an upgrade of hardware or software.

Companies or individuals may submit entries using the Web interface at [submissions.top500.org](http://submissions.top500.org). Submitters must also notify the maintainers of the TOP500 List by e-mail after their submission is complete.

Submissions should contain the following information:

* **Manufacturer:** Manufacturer or vendor.
* **Computer system:** Type indicated by manufacturer or vendor.
* **Installation Site:** Customer—full name of company or institution.
* **Location:** Location and country.
* **Year:** Year of installation/last major update.
* **Field of Application:** Type of customer (such as university, government, or industry) in addition to typical application (for example, geophysics, automotive, chemical, or benchmarking).
* **Processors:** Number of processors (as well as type, if necessary).
* **Memory:** Main memory configuration of the system.

## Method of Solution

To provide a comparable set of performance numbers across all computers, the algorithm used to solve the system of equations in the LINPACK benchmark procedure must conform to LU factorization with partial pivoting (more detail can be found at [http://www.netlib.org/benchmark/hpl/](http://www.netlib.org/benchmark/hpl/%20)). This excludes the use of a fast matrix-multiply algorithm or algorithms that compute a solution with a lower-than-full precision and then refine the solution using an iterative approach. You can obtain an implementation of the appropriate software at <http://www.netlib.org/benchmark/hpl/>.

## Verification

To verify entries, compilers of the TOP500 List cross-check the different sources of information. They randomly select a statistically representative sample of the first 500 systems in the database, and for these systems, they ask for direct verification of the given information.

## Restrictions

Because the TOP500 List aims to provide reliable information about the state of HPC, compilers of the list limit the number of entries of systems installed at any single vendor site. The accumulated performance of systems at vendor sites is limited to a maximum of 5% of the total accumulated installed performance of the vendor. (Rounding favors the vendor.)

Eligibility for the TOP500 List is limited to general-purpose systems that are in common use for high-end applications. A general-purpose system is used to solve a range of scientific problems. Any system designed specifically to solve the LINPACK benchmark problem, or any system that has a TOP500 ranking as its main goal, will be disqualified.

Detailed submission instructions can be found at <http://www.top500.org/project/call_for_participation>.

# Additional Resources

**General HPC information**

* <http://www.microsoft.com/hpc>
* [http://www.windowshpc.net](http://www.windowshpc.net/)
* <http://technet.microsoft.com/en-us/library/cc510343.aspx>

**Windows HPC Server 2008 resources (uSane, the LINPACK Tuning Wizard, the Excel add-in, and the Excel Sizing Sheet)**

* <http://windowshpc.net/Resources/Pages/Programs.aspx>

**Windows Server x64 information**

* <http://www.microsoft.com/x64/>

**Windows Server system information**

* <http://www.microsoft.com/windowsserver>
* Microsoft MVPs with the Windows Server team.2008. *Windows Server® 2008 Resource Kit*. Microsoft Press. <http://www.microsoft.com/mspress/books/10345.aspx>.

* **Get the Facts Web site**  
  <http://www.microsoft.com/getthefacts>

# Appendix A: Sample Node XML File

Following is a sample node XML file. The node XML file is based on an XML Schema Definition (XSD) language file: NodeConfigurationFile.xsd. This XSD file is available on the head node, in the Bin folder of the installation path for HPC Pack 2008. For example, if you are using the default installation path, the XSD file is available at:

C:\Program Files\Microsoft HPC Pack\Bin\NodeConfigurationFile.xsd

For a table that lists and describes the attributes and elements that are defined in the node XML file schema, see the “Design and Deployment Guide for Windows HPC Server 2008.” (<http://www.microsoft.com/Downloads/details.aspx?FamilyID=6e20fba5-ce39-44b1-8b3d-76cb31c01a70&displaylang=en>).

The most important fields in the file are the node name and the MAC address. Note that the administrator can optionally add one or more “tag” entries to designate node groups (for example, Tag=row1, Tag=rack1, Tag=chassis3). This enables the administrator to select nodes by their tags (also called node groups in the user interface), from which actions can be performed on groups.

<?xml version="1.0" encoding="utf-8" standalone="yes" ?>

<Nodes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.microsoft.com/HpcNodeConfigurationFile/2007/12">

<Node

Name="ComputeNodeName1"

Domain="CONTOSO"

MachineGuid="{4c4c4544-0038-5710-804b-c6c04f464331}">

<Location

DataCenter="Data Center 1"

Rack="2"

Chassis="1" />

<Template

Name="Default ComputeNode Template" Provisioned="True" />

<MacAddress>00301B445F02</MacAddress>

<MacAddress>001B2104EDF5</MacAddress>

<Tag>ComputeNodes</Tag>

<Tag>Rack2</Tag>

</Node>

</Nodes>

# Appendix B: The HPL Parameters

Following is a description of the HPL parameters, along with the default values and tips for large-scale clusters.

|  |  |  |  |
| --- | --- | --- | --- |
| Line | Value | Description | Comments |
| 3 | HPL.out | The output is written to this file (depending on the device output choice). |  |
| 4 | 6 | Device out (6=stdout, 7=stderr,file) |  |
| 5 | 1 | # of problem sizes (N) | Number of different problem sizes that you will test. |
| 6 | 10000 | N | The problem size (refer to the spreadsheet for its calculation). |
| 7 | 1 | # of NBs | Number of different block numbers that you will test. |
| 8 | 64 | NB | Number of blocks used to split the problem size (refer to the spreadsheet for its calculation). |
| 9 | 0 | PMAP process mapping (0=Row-, 1=Column-major) | 0 for best results on clusters. |
| 10 | 1 | # of process grids (P x Q) (1xN works best on Ethernet hub) | Number of different process grids that you will test. |
| 11 | 2 | P | P\*Q = total HPL processes. |
| 12 | 2 | Q | P\*Q = total HPL processes. |
| 13 | 16 | Threshold | The threshold to which the residuals should be compared with. Use a negative value to save time and skip the final check—the run will be marked as failed. |
| 14 | 1 | # of panel factorizations (PFACTs) |  |
| 15 | 1 | PFACTs (0=left, 1=Crout, 2=Right) |  |
| 16 | 1 | # of recursive stopping criteria |  |
| 17 | 4 | NBMINs (>= 1) |  |
| 18 | 1 | # of panels in recursion |  |
| 19 | 2 | NDIVs |  |
| 20 | 1 | # of recursive panel factorizations (RFACTs) |  |
| 21 | 2 | RFACTs (0=left, 1=Crout, 2=Right) |  |
| 22 | 1 | # of broadcasts (BCASTs) |  |
| 23 | 1 | BCASTs (0=1rg, 1=1rM, 2=2rg, 3=2rM, 4=Lng, 5=LnM) | BCAST=1 is best for clusters. |
| 24 | 1 | # of lookahead depth |  |
| 25 | 0 | DEPTHs (>= 0) |  |
| 26 | 2 | SWAP (0=bin-exch, 1=long, 2=mix) |  |
| 27 | 64 | Swapping threshold | Set equal to NB. |
| 28 | 0 | L1 in (0=transposed, 1=no-transposed) form | Keep 0. |
| 29 | 0 | U in (0=transposed, 1=no-transposed) form | Keep 0. |
| 30 | 1 | Equilibration (0=no, 1=yes) | Always 1. |
| 31 | 8 | Memory alignment in double (> 0) | Can be 4, 8, or 16. |

# Appendix C: Additional Information about the HPL Parameters

The Linpack NxN benchmark computes the solution of a linear system of equations **A.x = b**, where **A** is a dense NxN random matrix, and **x** and **b** are vectors of size N. Matrix **A** is first factored in place as the product **A= L.U**, where **L** and **U** represent lower and upper triangular matrices. For numerical stability, the factorization algorithm uses partial row pivoting. The solution **x** is then found in two successive triangular solve steps, **L.z=b** and finally **U.x=z**. The factorization requires (2/3)\*N3 floating point operations, and the two triangular solves account for N2 floating point operations each; as N increases, the factorization phase dominates the computation time.

N is driven by the maximum amount of memory that your system can deliver to the Linpack executable before you start to paginate. This implies that you must minimize the operating system (smallest memory footprint).

Assuming that the size of the Linpack matrix is 8\*N\*N bytes and that you leave approximately 20 percent of the memory for the system, N can be chosen with the following formula:

If N is too high, the system starts to paginate and the performance drops, while if N is too low, then the system is not optimized.

## HPL Processes Distribution

The Linpack matrix NxN is distributed onto a two-dimensional PxQ grid of processes and then divided into subgroups of NBxNB tiles. LU factorization is done by a loop of iterations. Data related to the ith iteration are presented in Figure 23: D is the ith block of the main diagonal; L, U and T are the current part of the lower, upper, and trailer matrices.



Figure 23 Data Pattern

The D matrix is divided in P\*Q processes, with eight processes on each node. On 256 nodes with P=32 and Q=64, the processes map becomes:



Figure 24 Processes map

The communication pattern when using the Linpack BCAST parameter 1 (Increasing Ring Modified) is described in the following figure.

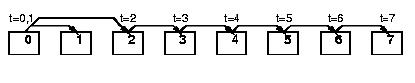


Figure 25 Increasing ring modified panel broadcast

When you combine the processes map with the panel, broadcast goes through rank 0 to rank 2048, and that panel factorization jumps from 0 → 64, 1 → 65. This implies that within the same system, as long as the MPI rank order matches the order of Linpack processes, the main communication line will be rank-to-rank.