

Advertisement: Support LinuxWorld, click here!



March 2000

#### Navigate

[Home](#)  
[Topical Index](#)  
[Archive](#)


#### Subscribe

It's FREE!  
And you'll get  
regular e-mail  
updates

#### Contact Us

[Masthead](#)  
[Advertising Info](#)  
[Writer Guidelines](#)  
[Link to LinuxWorld](#)  
[Copyright](#)

#### Search

  
Sponsored by:

## Cluster Management

# Linux clustering cornucopia

## Which cluster is for you?

### Summary

Rawn Shah serves as your expert guide through the maze of both open- and closed-source clustering solutions available for Linux today. (5,000 words)

### By Rawn Shah

Trying to count clustering projects in Linux is like trying to count the number of startup companies in Silicon Valley. Unlike Windows NT, which has been hindered by its closed environment, Linux has a large selection of clustering systems available for different uses and needs. That doesn't make it any easier for those trying to figure out what clustering system they should use.

Part of the problem is the fact that the term *clustering* is used in different contexts. While an IT manager might be concerned with keeping maximum uptime of their servers or making the applications run faster, a mathematician may be more interested in performing large scale numerical calculations on their servers. Both need a cluster, but each needs a cluster of different properties.

This article surveys different forms of clustering and some of the many implementations that are available commercially and as freeware. Although not all the solutions listed here are open source,

### Other clustering projects

This article surveys many of the Linux clustering projects that are available commercially or as freeware. It also examines the different types of clustering available and how those are implemented under Linux.

If you know about or are working on another Linux clustering project, please send it to us -- its formal name, a one-sentence description, and an URL -- and we'll be happy to list it here.

The solutions profiled in this article include:

1. Beowulf
2. Giganet cLAN
3. Legion

much of the software follows the common practices of distributing Linux source code, especially since those who implement clustering often also want to tweak the performance of the system to their needs.

## Hardware

Clustering always involves hardware connections between machines. In most cases today, that is simply a Fast Ethernet card and a hub. But at the high-end scientific, there are a variety of network interface cards designed specifically for clustering.

Those include Myricom's Myrinet, Gigaset's cLAN and the IEEE 1596 standard Scalable Coherent Interface (SCI). Those cards' function is not only to provide high bandwidth between the nodes of the cluster but also to reduce the latency (the time it takes to send messages). Those latencies are crucial to exchanging state information between the nodes to keep their operations synchronized.

## Myricom

Myricom offers cards and switches that interconnect at speeds of up to 1.28 Gbps in each direction. The cards come in two different forms, copper-based and optical. The copper version for LANs can communicate at full speed at a distance of 10 feet but can operate at half that speed at distances of up to 60 feet. Myrinet on fiber can operate at full speed up to 6.25 miles on single-mode fiber, or about 340 feet on multimode fiber. Myrinet offers only direct point to point, hub-based, or switch-based network configurations, but it is not limited in the number of switch fabrics that can be connected together. Adding switch fabrics simply increases the latency between nodes. The average latency between two directly connected nodes is 5 to 18 microseconds, a magnitude or more faster than Ethernet.

## Gigaset

Gigaset is the first vendor of Virtual Interface (VI) architecture cards for the Linux platform, in their cLAN cards and switches. The VI architecture is a platform-neutral software and hardware system that Intel has been promoting to create clusters. It uses its own network communications protocol rather than IP to exchange data directly between the servers, and it is not intended to be a WAN routable system. The future of VI now lies in the ongoing work of the System I/O Group, which in itself is a merger of the Next-Generation I/O group led by Intel, and the Future I/O Group led by IBM and Compaq. Gigaset's products can currently offer 1 Gbps unidirectional communications between the nodes at minimum latencies of 7 microseconds.

## IEEE SCI

The IEEE standard SCI has even lower latencies (under 2.5 microseconds), and it can run at 400 MB per second (3.2 Gbps) in each direction. SCI

4. Cplant
5. JESSICA 2
6. PARIS
7. Linux Virtual Server
8. TurboLinux TurboCluster and enFuzion
9. Platform Computing's LSF Batch
10. Resonate Dispatch series
11. MOSIX
12. Linux-HA Project

## Types of clustering

The three most common types of clusters include high-performance scientific clusters, load-balancing clusters, and high-availability clusters.

### ● Scientific clusters

The first type typically involves developing parallel programming applications for a cluster to solve complex scientific problems. That is the essence of parallel computing, although it does not use specialized parallel supercomputers that internally consist of between tens and tens of thousands of separate processors. Instead, it uses commodity systems such as a group of single- or dual-processor PCs linked via high-speed connections and communicating over a common messaging layer to run those parallel applications. Thus, every so often, you hear about another cheap Linux supercomputer coming out. But that is actually a cluster of computers with the equivalent processing power of a real supercomputer, and it usually runs over \$100,000 for a decent cluster configuration. That may seem high for the average person but is still cheap compared to a multimillion-dollar specialized supercomputer.

is a ring-topology-based networking system unlike the star topology of Ethernet. That makes it faster to communicate between the nodes on a larger scale. Even more useful is a torus topology network, with many rings between the nodes. A two-dimensional torus can be pictured as a grid of  $n$  by  $m$  nodes with a ring network at every row and every column. A three-dimensional torus is similar, with a 3D cubic grid of nodes that also has rings at every level. Supercomputing massively parallel systems use those to provide the relatively quickest path for communications between hundreds or thousands of nodes.

The limiting factor in most of those systems is not the operating system or the network interfaces but the server's internal PCI bus system. Basic 32-bit, 33-MHz PCI common in nearly all desktop PCs and most low-end servers offers only 133 MB per second (1 Gbps), stunting the power of those cards. Some costly high-end servers such as the Compaq Proliant 6500 and IBM Netfinity 7000 series have 64-bit, 66-MHz cards that run at four times that speed. Unfortunately, the paradox arises that more organizations use the systems on the low end, and thus most vendors end up building and selling more of the low-end PCI cards. Specialized network cards for 64-bit, 66-MHz PCI also exist, but they come at a much higher price. For example, Intel offers a Fast Ethernet card of that sort for about \$400 to \$500, almost five times the price of a regular PCI version.

### **Scientific clusters**

The reason some of the parallel clusters systems can achieve such high bandwidth and low latencies is that they usually skip the use of network protocols like TCP/IP. Although the Internet Protocol is great for wide area networking, it contains too much overhead that isn't necessary in a closed network cluster in which the nodes are known to each other. Instead, some of those systems can use direct memory access (DMA) between the nodes, which is similar to how some graphics cards and other peripherals work inside a single machine. Thus across the cluster, a form of distributed shared memory can be accessed directly by any processor on any node. They can also use a low-overhead messaging system to communicate between the nodes.

supercomputer.

Those are profiled on this page.

- **Load-balancing clusters**

Load-balancing clusters provide a more practical system for business needs. As the name implies, that system entails sharing the processing load as evenly as possible across a cluster of computers. That load could be in the form of an application processing load or a network traffic load that needs to be balanced. Such a system is perfectly suited for large numbers of users running the same set of applications. Each node can handle part of that load, and the load can be dynamically assigned between the nodes to balance it out. The same holds for network traffic. Often network server applications take in too much incoming traffic to be able to process it quickly enough and thus the traffic needs to be sent to network server applications running on other nodes. That can also be optimized according to the different resources available on each node or the particular environment of the network.

These are profiled on page 2.

- **High-availability clusters**

High-availability clusters exist to keep the overall services of the cluster available as much as possible. to take into account the fallibility of computing hardware and software. As the primary node in a high-availability cluster fails, it is replaced by a secondary node that has been waiting for that moment. That secondary node is usually a mirror image of the primary node, so that when it does replace the primary, it can completely take over its identity and thus keep the system environment consistent from the user's point of view.

These are profiled on page 3.

With each of those three basic types of clusters, hybrids and interbreeding often occur between them. Thus you can find a high-availability cluster that can also load-balance users across its nodes, while still attempting to maintain a degree of high-availability. Similarly, you can find a parallel cluster that can also perform load balancing between the nodes separately from what was programmed into the application. Although the clustering system itself is independent of what software or hardware is in use, hardware connections play a pivotal role when it comes to running the system efficiently.

The Message Passing Interface (MPI) is the most common implementation of a messaging layer between the parallel cluster systems. MPI exists in several variations but in all cases, it offers a common API for developers to parallel applications without having to manually figure out how code segments can be distributed across the nodes of the cluster. The Beowulf system, for one, uses MPI as the common programming interface.

It is difficult to decide on which high-performance clustering package to use. Many offer similar services, but the specifics of your computational needs are the determining factors. In many cases, the research work in those systems is only part-way to solving your needs, and using the software may require specific assistance and collaboration with the clustering package developers.

### 1. **Beowulf**

When asked about Linux clustering, the most immediate response by many is Beowulf. That is the most well known of scientific software clustering systems for Linux. There really isn't a single package called Beowulf. Rather, it is the term applied to a common set of software tools that run over the Linux kernel. That includes popular software messaging APIs like the Message Passing Interface (MPI) or the Portable Virtual Machine (PVM), tweaks to the Linux kernel to allow bonding several Ethernet interfaces, high-performance network drivers, changes to the virtual memory manager, and distributed interprocess communication (DIPC) services. A common global process identifier space allows access to any process from any of the nodes using the DIPC mechanism. Beowulf also supports a wide range of hardware connectivity options between the nodes.

Beowulf is probably the first high-performance clustering system that you will look at when considering Linux, simply because of its wide use and support. There is a lot of documentation, and even several books on the subject. The difference between Beowulf and some of the following scientific clustering systems can be real or just a difference in product names. For example, Alta Technologies' AltaCluster is a Beowulf system despite the name difference. Some vendors such as ParTec AG, a German company, offer variations on the Beowulf model to include other management interfaces and communications protocols.

### 2. **Giganet cLAN**

Giganet offers a custom hardware-based solution that uses a non-IP protocol to communicate between nodes in a scientific cluster. As described earlier, the Virtual Interface protocol supports faster communications between servers by removing much of the overhead of protocols such as IP. Additionally, the hardware system can run at gigabit speeds at very low latencies, making it very suitable for building up to 256-node scientific clusters. The vendor does support MPI and thus many of the parallel applications that can run on similar systems such as Beowulf.

It shares the same disability with Beowulf, that is, it should not be used as a network load-sharing system, unless you want to write an application that monitors and distributes those network packets between the servers.

### 3. **Legion**

Legion is an attempt to build a true multicomputer system. That is a cluster whereby each node is

an independent system, but the overall system appears to the user as a single computer. Legion was designed to support a single worldwide computer consisting of millions of hosts and trillions of software objects. Within Legion, users can construct their own collaborative groups.

**Legion provides high-performance parallelism, load-balancing, distributed data management and fault-tolerance.**

Legion provides high-performance parallelism, load-balancing, distributed data management, and fault-tolerance. It supports high-availability through its fault-tolerance management and dynamic reconfiguration across the member nodes. It also has an extensible core that can be dynamically replaced or upgraded over time as new advancements or developments arise. The system is not under a single monolithic control but can be managed by any

number of organizations, each supporting an autonomous part of the whole. The Legion API provides high-performance computing through its built-in parallelism.

Legion does not require specifically written software to be able to use its API library. It sits on top of the user's computer operating system and negotiates between the local and the distributed resources. It handles resource scheduling and security automatically and also manages a context space to describe and access any object out of the trillions of possibilities in the entire system. However, it does not need to run under system administrator privileges on each node and can work within the confines of a nonprivileged user account. That increases the flexibility of the nodes and users that can join Legion.

#### 4. Cplant

The Computational Plant at Sandia National Labs is a large-scale massively parallel cluster to achieve TeraFLOP (trillions of floating point operations) computation and built on commodity components. The entire system consists of Scalable Units that are partitioned to serve a different purpose (computation, disk I/O, network I/O, services management). Each node in the cluster is a Linux box with custom-developed kernel-level modules used to provide the partition services. The function of each partition can be modified by loading and unloading the kernel level modules.

The project was done in three phases, starting with a prototype of 128 433-MHz DEC Alpha 21164 based systems, each with 192 MB of RAM and 2 GB drives, interconnected with Myrinet cards and 8-port SAN switches. Phase I expanded that to 400 21164-based workstations running at 500 MHz, and 192 MB of RAM, with no storage, connected with 16-port SAN switches in a cube of hypercubes, and running Red Hat 5.1. The current Phase II has 592 DEC 21264-based machines running at 500 MHz, with 256 MB of RAM and no drives. Each node has a 64-bit, 33-MHz PCI Myrinet card connected to a 16-port switch, again in cubes of hypercubes.

The applications running on Cplant include solving sparse linear systems, optimization for computational systems in fluid and structural dynamics, simulations of molecular dynamics, finite element modal analysis for linear structural dynamics, and a dynamic load-balancing library for parallel applications.

#### 5. JESSICA 2

Hong Kong University's Systems Research Group has a Java-based cluster called the Java-Enabled Single System Image Computing Architecture (JESSICA) that acts as a middle-ware layer to achieve the illusion of a single-system image. That layer is a single global thread space of all the threads running on each of the nodes that communicate through a distributed shared memory (DSM) system. That project uses the ThreadMark DSM but will eventually replace it with one of their own creation known as the JiaJia Using Migrating-home Protocol (JUMP). They use their custom-built Java-based ClusterProbe software to manage the cluster's 50 nodes.

## 6. PARIS

The Programming pArallel and distRibuted systems for large scale numerical sImulation applicationS (PARIS) project at the IRISA research insititute in France provides several tools to create a cluster of Linux servers. The project constitutes three components: resource management software for clusters, runtime environments for parallel programming languages, and software tools for distributed numerical simulation.

The resource management software includes the Globelins distributed system for sharing memory, disk, and processor resources, and their Duplex and Mome distributed shared memory system.

## Load-balancing clusters

Load-balancing clusters distribute network or compute processing load across multiple nodes. The differentiating factor in that case is the lack of a single parallel program that runs across those nodes. Each node server in that type of cluster, in most cases, is an independent system running separate software. However, there is a common relationship between the nodes either in the form of direct communications between the node or through a central load-balancing server that controls each node's load. Usually, a specific algorithm is used to distribute that load.

Network traffic load-balancing is the process of examining the incoming traffic to a cluster and distributing the traffic to each of the nodes for processing as appropriate. That is best for heavy-duty network applications such as Web or FTP servers. Load-balancing networked application services requires the cluster software to examine the current load of each node and determine which nodes are able to take on new jobs. That is best for running serial and batch processing jobs such as data analysis. Those systems can also be configured to take into account the hardware or operating system features of specific nodes: thus, uniform nodes in a cluster are not necessary.

## 7. Linux Virtual Server

The Linux Virtual Server project has implemented a number of kernel patches that create a load-balancing system for incoming TCP/IP traffic. The LVS software examines incoming traffic, and based upon a load-balancing algorithm, redirects that to a set of servers acting as a cluster. That allows network applications such as Web servers to run on a cluster of nodes to support a greater number of users.

LVS supports cluster nodes that are directly attached to the same LAN as the load-balancing server, but it can also connect to remote servers by way of tunneling IP packets. That latter method involves encapsulating the balanced requests inside IP packets sent directly from the load-balancing server to the remote cluster node. Although LVS can support load-balancing to Websites remotely, the load-balancing algorithms it uses now are not efficient for widely spread

Web servers in a virtual cluster. Thus LVS works best when the Web servers are on the same LAN as the load-balancing server.

A number of hardware implementations of that system of load balancing can run much faster than on a general-purpose operating system such as Linux. They include those from Alteon and Foundry, and they have hardware logic and minimal operating systems that can perform the traffic management in hardware at much higher speeds than pure software. They also come at a hefty price tag, usually starting above \$10,000. If you need a simple and cheap solution, a mid-range Linux box with lots of memory (256 MB) makes for a good load-balancing system.

## 8. **TurboLinux TurboCluster and enFuzion**

TurboLinux has a product called TurboCluster that was originally based on the kernel patches developed by the Linux Virtual Server project. Thus it allows most of the same benefits, and it has the same drawbacks as the originating project. TurboLinux has also developed some tools for monitoring the behavior of the cluster that adds to the usefulness of the product. The commercial support from a leading vendor also makes it more attractive for large sites.

EnFuzion is an upcoming scientific clustering product from TurboLinux that isn't based on Beowulf. However, it does support hundreds of nodes and a number of different non-Linux platforms including Solaris, Windows NT, HP-UX, IBM AIX, SGI Irix, and Tru64. EnFuzion is interesting because it runs any existing software, and it doesn't need custom parallel applications written to the environment. It supports automated load balancing and resource sharing between the nodes and failed jobs can be automatically rescheduled.

**EnFuzion supports automated load balancing and resource sharing between the nodes, and failed jobs can be automatically rescheduled.**

## 9. **Platform Computing's LSF Batch**

Platform Computing, a veteran in the cluster computing area, now offers its Load-Sharing Facility (LSF) Batch software on the Linux platform. LSF Batch allows a central controller to schedule jobs to run on any number of nodes in a cluster. It is similar in concept to the TurboLinux enFuzion software and supports any type of application to be run on a node.

That method is very flexible to cluster size since you can specifically select the number of nodes or even the nodes themselves that should run the application. Thus you can split a 64-node cluster into smaller logical clusters, each running its own set of batch applications. Furthermore, it can reschedule the job on other servers should the application or node fail.

Platform's products run on major Unix systems as well as Windows NT. At that point, only their LSF Batch product has been ported to Linux. Eventually, the rest of LSF Suite's components will follow.

## 10. **Resonate Dispatch series**

Resonate has a software-based load-balancing approach similar to the Linux Virtual Server.

However, it supports more features and some better load-balancing algorithms. For example, using Resonate, you can load an agent on each of the cluster nodes that determines the current system load for that node. The load-balancing server then checks the agents on each of the nodes to determine which is the least loaded and sends new traffic to it. Additionally, Resonate can also support geographically distributed servers more efficiently with itsr Global Dispatch product.

Resonate has tested the software thoroughly on Red Hat Linux, but there is no real reason that it can't run on other distributions as well. Resonate's software also runs on various other platforms, including Solaris, AIX, Windows NT, and it can load-balance in a mixed environment as well.

## 11. MOSIX

MOSIX uses Linux kernel adaptations to implement a process load balancing clustering system. Within that cluster, any server or workstation can join or leave as assigned, thus adding to or removing from the total processing power of the cluster. According to its documentation, MOSIX uses adaptive process load-balancing and memory ushering algorithms to maximize overall performance. Application processes can be preemptively migrated between nodes to take advantage of the best resources, similar to the way a symmetric multiprocessor system can switch applications between the various processors.

MOSIX is completely transparent at the application level and requires no recompilation or relinking to new libraries, since everything occurs at the kernel level. It can be configured as a multiuser shared environment cluster in several ways. There can be a single pool of all servers and systems that are part of the cluster, or it can be dynamically partitioned into several sub-clusters, each for a different use. Linux workstations can also be part of the cluster on a full-time basis, part-time basis, or just as batch job submitters. As a part-time cluster node, the workstation can be used to increase the cluster processing capabilities during off-hours while sitting idle. A cluster can also be used only in batch mode where it is configured to accept batch-processing jobs through a queue. A daemon then takes the jobs and sends them to the cluster nodes to be processed.

MOSIX provides an interesting option for creating clustered environments in corporate settings, in addition to high-performance scientific computing. By using idle resources on servers and workstations, it can create and run applications much faster and more efficiently. Since it has access to multiple servers and can dynamically resize clusters and change load-balancing rules, it also offers a high degree of server availability. The downside of MOSIX is that it changes some of the core parts of the Linux kernel's behavior, and thus system-level applications may not function as expected. MOSIX is also limited currently when it comes to network applications that use socket connections based on a single server address. That means that when a network application starts running on a server node, it has to continue running on that node while the IP address is bound to the socket. Apparently MOSIX is working on migrating sockets as well, so that may soon become a moot point.

**The downside of MOSIX  
is that it changes  
some of the core parts  
of the Linux kernel's behavior  
and thus system-level applications  
may not function  
as expected.**



## High-availability clusters

High-availability (HA) clusters focus on keeping a server system running and responsive as much of the time as possible. They usually employ redundant nodes and services running on multiple machines to keep active track of each other. Should a node fail, its secondary will take over its responsibilities in a matter of seconds -- or even less. Thus, from the user's perspective, the cluster never goes down.

Some HA clusters can also maintain redundant applications across the nodes. Thus a user's application will continue to run even if the node he or she was working on fails. The running application is migrated to another node in a few seconds, and all the user perceives is a slight slowdown in responsiveness. That kind of application-level redundancy, however, requires the software to be designed as cluster-aware and to know what to do in case of a node failure. But that is mostly unavailable for Linux today, since there is no standard in HA clustering for Linux systems and no common API that application developers can use to build cluster-aware software.

HA clusters may perform load-balancing, but systems typically just keep the secondary servers idle while the primary server runs the jobs. The secondary server is usually a mirror of the operating system setup of the primary, even if the hardware itself is slightly different. The secondary nodes keep an active monitor or heartbeat watch on the primary to see if it is still running. Should the heartbeat timer go off without the primary responding, the secondary will take over the network and system identity (IP hostname and address in the case of Linux systems).

Unfortunately, Linux is still a little lax in that area. The good news is that the big-name vendors are working their best to bring about high-availability as quickly as they can, as it is a feature that's commonly demanded of their enterprise class servers.

### 12. Linux-HA Project

The High-Availability Linux project, according to its goal statement, aims to provide a high-availability solution for Linux that promotes reliability, availability, and serviceability through a community development effort. That is an attempt to give Linux the same competitive features as leading Unix systems such as Solaris, AIX and HP/UX when it comes to high-availability clustering. Thus the project's goal is to achieve analyst group D. H. Brown's specified level of functionality in its Unix clustering comparison report (<http://www.sun.com/clusters/dh.brown.pdf>) by 2001.

The project has software that can maintain heartbeats between nodes and take over IP addresses of failed nodes. Should a node fail, it uses the Fake Redundant IP software package to add the address of the failed node onto a working node to assume its responsibilities. Thus the failed node can be replaced automatically in a matter of milliseconds. For practical use, that heartbeat is usually kept in the several seconds range, unless you have a dedicated network link between the nodes. At that point, the user applications on the failed system still need to be restarted on the new node.

## Clustering everywhere

A wide choice of clustering systems are available for Linux. At the same time, several of those projects are noncommercial and even experimental. Although that does not pose a problem for academic situations and some organizations, big business typically prefers a commercially supported platform from a known vendor. Vendors such as IBM, SGI, HP, and Sun offer products or services to build

scientific clusters under Linux, due to its popularity and the possibility of selling a large quantity of server equipment. Once the other forms of clustering are perceived as reliable by commercial organizations, those same server vendors will likely create their own products around open source clustering solutions.

The importance of Linux as a server platform hinges on the ability to support large servers and clusters of servers. That gives it a leverage point to compete on a higher plane with Unix servers from Sun, HP, IBM, and others. Although Windows NT and 2000 do not support the range of clustering that Linux can, the availability of an official method of HA clustering and an API for building cluster-aware applications puts it in the running as well.

If you are considering building a cluster, you should definitely examine some of those possibilities and compare them against your needs. You may find out that what you wish to accomplish is not yet available as a complete solution, or you may find a ready solution. Either way, be assured that many existing organizations trust their applications to clusters of Linux systems doing deep calculations or serving large numbers of Webpages. Clustering is one enterprise system service that has been tested under Linux successfully. Although new ones will emerge, the variety of choices is an advantage for Linux over other systems such as Windows NT. ■

**Discuss this article in the LinuxWorld forums** (1 postings)  
(Read our forums FAQ to learn more.)

#### **About the author**

Rawn Shah is an independent consultant in Tucson, Ariz. He has worked with and written about multiplatform issues for years, and he is constantly surprised at how few people know about useful system tools. Check out his Linux Reviews Central discussion, hosted by ITworld.com.

---

Advertisement: Support LinuxWorld, click here!



---

(c) 2000 LinuxWorld, published by ITworld.com, Inc., an IDG Communications company

#### **Resources**

- Giganet's cLAN products:  
<http://www.giganet.com/products/indexlinux.htm>
- Fake: Redundant Server Switch:  
<http://linux.zipworld.com.au/fake/>
- ParTec AG's ParaStation Cluster:  
<http://www.par-tec.com/>

#### **Clustering hardware**

- Myricom:  
<http://www.myricom.com>
- Giganet:  
<http://www.giganet.com>

### **Scientific clusters**

- Beowulf:  
<http://www.beowulf.org>
- Legion:  
<http://legion.virginia.edu>
- Cplant:  
<http://www.cs.sandia.gov/cplant/>
- JESSICA 2:  
<http://www.srg.csis.hku.hk/>
- PARIS:  
<http://www.irisa.fr/paris/>

### **Load-balancing clusters**

- Linux Virtual Server Project:  
<http://www.linuxvirtualserver.org>
- TurboLinux TurboCluster and enFuzion:  
<http://www.turbolinux.com/products/>
- Platform Computing's LSF Suite:  
<http://www.platform.com/platform/platform.nsf/webpage/LSFDataSheets?OpenDocument>
- Resonate Dispatch series:  
<http://www.resonate.com/products/index.php3>
- MOSIX:  
<http://www.mosix.cs.huji.ac.il/>

### **High-availability clusters**

- High-Availability Linux Project:  
<http://www.linux-ha.org>

Feedback: [lweditors@linuxworld.com](mailto:lweditors@linuxworld.com)

Technical difficulties: [webmaster@linuxworld.com](mailto:webmaster@linuxworld.com)

URL: <http://www.linuxworld.com/lw-2000-03/lw-03-clustering.html>

Last modified: Thursday, March 30, 2000