Windows 2000 Server

Chapter 19 - Network Load Balancing

Network Load Balancing is one of the Windows Clustering features of Microsoft® Windows® 2000 Advanced Server. Network Load Balancing can enhance the availability and scalability of Internet server programs such as those used on Web servers, FTP servers, and other mission-critical servers. The material in this chapter will help you configure Network Load Balancing clusters to deliver the reliability and performance that high-volume Web servers and other mission-critical servers need.

In This Chapter

Network Load Balancing Overview
Implementing Network Load Balancing
Network Load Balancing with Network Hardware Switches
Scenarios
Default Handling of Client Requests
Wlbs Display Command
Changing Network Load Balancing Resource Limits in the Registry
Additional Resources

Network Load Balancing Overview

Network Load Balancing clusters distribute client connections over multiple servers, providing scalability and high availability for client requests for TCP/IP-based services and applications.

The heart of Network Load Balancing is the driver Wlbs.sys, which is loaded into each member server, or *host*, in the cluster. Wlbs.sys includes the statistical mapping algorithm that the cluster hosts collectively use to determine which host handles each incoming request.

Load-balanced applications need to carefully manage state on the server. State that is persistent across multiple requests or that is shared among clients needs to be shared in locations that are transparently accessible from all cluster hosts. Updates to state that is shared among the hosts needs to be synchronized — for example, by using a back-end database server.

Network Load Balancing provides the following:

- Load balancing

  Load balancing is static (unless the cluster set changes). The port rules that you create for the cluster's hosts specify the division of the incoming client connections.

- High availability

  If a host fails, the cluster detects the failure and redistributes subsequent client requests to hosts that are still viable members of the cluster. Any client connections that were open when the host failed are ended. On retry, the client connection is routed to a viable host. The downtime for client connections is less than 10 seconds.

- Scalability

  Network Load Balancing accomplishes scalability by allowing you to add as many as 32 hosts to the cluster. You can add hosts without shutting down the cluster.

How Network Load Balancing Works

Briefly, when Network Load Balancing is installed as a network driver on each of the cluster hosts, the cluster presents a virtual IP address to client requests. The client requests go to all the hosts in the cluster, but only the host to which a given client request is mapped accepts and handles the request. All the other hosts drop the request. Depending on configuration of port rules and on affinity, the statistical mapping algorithm, which is present on all the cluster hosts, maps the client requests to particular hosts for processing.

The hosts exchange heartbeat messages to maintain consistent data about the cluster's membership. If a host fails to send or does not respond to heartbeat messages, the remaining hosts perform convergence, a process in which they determine which hosts are still active members of the cluster. If a new host attempts to join the cluster, it sends heartbeat messages that trigger convergence. After all cluster hosts agree on the current cluster membership, the client load is repartitioned, and convergence completes.

Discussion of Network Load Balancing clusters requires clarification of two kinds of client states, application data state and session state:

- In terms of application data, you must consider whether the server application makes changes to the data store and whether the changes are synchronized across instances of the application (the instances that are running on the Network Load Balancing cluster hosts). An example of an application that does not make changes to the data store is a static Web page supported by an IIS server.

  Means must be provided to synchronize updates to data state that need to be shared across servers. One such means is use of a back-end database server that is shared by all instances of the application. An example would be an Active Server Pages (ASP) page that is supported by an IIS server and that can access a shared back-end database server, such as a SQL Server.

- Session state (or intraclient state) refers to client data that is visible to a particular client for the duration of a session. Session state can span multiple TCP connections, which can be either simultaneous or sequential. Network Load Balancing assists in preserving session state through client affinity settings. These settings direct all TCP connections from a given client address or class of client addresses to the same cluster host. This allows session state to be maintained by the server application in the host memory.

  Client/server applications that embed session state within "cookies" or push it to a back-end database do not need client affinity to be maintained.

  An example of an application that requires maintaining session state is an e-commerce application that maintains a shopping cart for each client.

By setting port rules, cluster parameters, and host parameters, you gain great flexibility in configuring the cluster, which enables you to customize the cluster according to the various hosts' capacities and sources of client requests. You can:

- Divide the load of incoming client requests among the hosts according to a given load partitioning, expressed as percentages of the incoming client connections. You can optionally route all requests of a given client to the host that handled the client's first request (single affinity).

  Network Load Balancing normalizes the load percentage based on the sum of assigned load percentages for all active hosts. In other words, if one host fails, the remaining hosts increase the number of client requests they handle, proportionally to their original load percentages. For example, assume each host in a four-host cluster is assigned 25 percent of the load. If one of these hosts fails, the three remaining active hosts would each handle 33 percent of the load.

- Specify that one host handle all client requests, with the others serving as failover alternatives.

You can combine the preceding capabilities by setting cluster and host parameters and creating port rules for your particular scenario. For guidelines on setting parameters and port rules for various scenarios, see "Scenarios" later in this chapter.

Before specific scenarios are discussed, the following sections explore the basic concepts of Network Load Balancing:

- System Requirements

  This section includes caveats and recommendations.

- Components

- Network Load Balancing Design

  This section covers basic concepts, such as the parameters and port rules, heartbeats and convergence, how Network Load Balancing maps client requests to hosts, and maintaining client connections.

- Implementing Network Load Balancing

  This section discusses the cluster and host parameters and the port rules in more depth.

System Requirements

The following are Network Load Balancing requirements:

- Windows 2000 Advanced Server
- TCP/IP protocol
- FDDI, Ethernet, or Gigabit Ethernet
- Cluster hosts that reside on the same physical subnet
- 1.5-megabyte (MB) hard disk space
- Between 250 KB and 4 MB of RAM, using the default parameters and depending on the network load

The following are some additional considerations in creating an environment for Network Load Balancing:

- It is not supported for a given server to be a member of both a Network Load Balancing cluster and a server cluster.
- Network Load Balancing does not support load-balancing on a token ring network.
- Network Load Balancing can operate on a mixed-version cluster — that is, on a cluster in which some hosts run Windows Load Balancing Service under Microsoft® Windows® NT version 4.0 and some run Network Load Balancing under Microsoft® Windows® 2000.
- Although only one network adapter is necessary per host, an additional network adapter is recommended for separating, on each host, client requests from other network traffic that is not related to Network Load Balancing, such as content replication or access to a back-end database.

For information about installing or upgrading Network Load Balancing, including rolling upgrades, see Windows 2000 Network Load Balancing Help.

Components

The following are the principal Network Load Balancing components. They are installed to each Network Load Balancing cluster host ("Wlbs" remains from a previous version of the software):

- Wlbs.sys

  The Network Load Balancing networking device driver.

- Wlbs.exe

  The Network Load Balancing control program. Except for changing registry parameters, you can use Wlbs.exe from the command line to start, stop, and administer Network Load Balancing, as well as to enable and disable ports and to query cluster status.

  For information about command-line syntax and arguments that Wlbs.exe carries out, see Windows 2000 Network Load Balancing Help.

- Wlbs.chm

  Network Load Balancing Help.

Network Load Balancing Design

Rather than routing incoming client requests through a central host for redistribution, every Network Load Balancing cluster host receives each client request. A statistical mapping algorithm determines which host processes each incoming client request. The distribution is affected by host priorities, whether the cluster is in multicast or unicast mode, port rules, and the affinity set.

This design has the following advantages:

- Because filtering packets is faster than modifying them in one host and then retransmitting them to their destination hosts, which then must receive them, Network Load Balancing provides significantly higher throughput than do load-balancing solutions that route packets through a central host.
- Network Load Balancing avoids a single point of failure and provides redundancy equal to the number of servers in a cluster.

- Because Network Load Balancing is a software solution, it scales with the technology of the servers where it is installed.

The trade-off for these advantages is that sending all the client traffic to all the hosts means that the network adapter(s) in each host must handle all the incoming client traffic (which is usually a small percentage of overall traffic).

Requests That Require Synchronized Change in Data State

When a Network Load Balancing host processes a client request that requires changing state information that is visible to all application instances, the change in data must be synchronized across all the hosts in the cluster. To accomplish this synchronization, the application can maintain shared state information in a back-end database and generate an update to the back-end database server. If the target application is managed as a server-cluster resource, the back-end servers can be members of a server cluster. The application can also provide other methods of its own design, such as cookies, for managing shared state information.

Heartbeats and Convergence

Network Load Balancing hosts maintain membership in the cluster through heartbeats. By default, when a host fails to send out heartbeat messages within about five seconds, it is deemed to have failed, and the remaining hosts in the cluster perform convergence, in order to do the following:

- Establish which hosts are still active members of the cluster.
- Elect the host with the highest priority as the new default host.

  Note that the lowest value for the Priority ID host parameter indicates the highest priority among hosts.
- Redistribute the failed host's client requests to the surviving hosts.

In convergence, surviving hosts look for consistent heartbeats; if the host that failed to send heartbeats once again provides heartbeats consistently, it rejoins the cluster in the course of convergence. The other consistency that active hosts establish during convergence is that all the hosts have a consistent view of which hosts are active.

Convergence generally takes less than 10 seconds, so interruption in client service by the cluster is minimal.

By editing the registry, you can change both the number of missed messages required to start convergence and the period between heartbeats. However, making the period between heartbeats too short increases network overhead on the system.

During convergence, hosts that are still up continue handling client requests.

Statistical Mapping Algorithm

The assignment of a given client request to a server occurs on all the hosts; there is not a single host that centrally distributes the requests among the hosts. The hosts jointly use a statistical algorithm that maps incoming client requests to active hosts in the cluster.

Apart from the influence of cluster and host parameter settings, it is possible for two successive client requests to be assigned to the same host during normal operation. However, as more client requests come into the cluster, distribution of client requests by the algorithm statistically approaches the load division specified by the Load Weight parameter of the relevant port rule.

The distribution of client requests that the statistical mapping function effects is influenced by the following:

- Host priorities
- Multicast or unicast mode
- Port rules
- Affinity
- Load percentage distribution
- Client IP address
- Client port number

- Other internal load information

The statistical mapping function does not change the existing distribution of requests unless the membership of the cluster changes or you adjust the load percentage.

Affinity

Affinity defines a relationship between client requests from a single client address or from a Class C network of clients and one of the cluster hosts. Affinity ensures that requests from the specified clients are always handled by the same host. The relationship lasts until convergence occurs (namely, until the membership of the cluster changes) or until you change the affinity setting. There is no time-out — the relationship is based only on the client IP address.

There are three types of affinity, which you choose with the Affinity setting. The Affinity setting determines which bits of the source IP and IP port number affect the choice of a host to handle traffic for a particular client's request. The Affinity settings are as follows:

- None

  Setting Affinity to None distributes client requests more evenly; when maintaining session state is not an issue, you can use this setting to speed up response time to requests. For example, because multiple requests from a particular client can go to more than one cluster host, clients that access Web pages can get different parts of a page or different pages from different hosts.

  With Affinity set to None, the Network Load Balancing statistical mapping algorithm uses both the port number and entire IP address of the client to influence the distribution of client requests.

  In certain circumstances, setting Affinity to None is suitable when the Network Load Balancing cluster sits behind a reverse proxy server. All the client requests have the same source IP address, so the port number creates an even distribution of requests among the cluster hosts.

- Single

  When Affinity is set to Single, the entire source IP address (but not the port number) is used to determine the distribution of client requests.

  You typically set Affinity to Single for intranet sites that need to maintain session state. Single Affinity always returns each client's traffic to the same server, thus assisting the application in maintaining client sessions and their associated session state.

  Note that client sessions that span multiple TCP connections (such as ASP sessions) are maintained as long as the Network Load Balancing cluster membership does not change. If the membership changes by adding a new host, the distribution of client requests is recomputed, and you cannot depend on new TCP connections from existing client sessions ending up at the same server. If a host leaves the cluster, its clients are partitioned among the remaining cluster hosts when convergence completes, and other clients are unaffected.

- Class C

  When Affinity is set to Class C, only the upper 24 bits of the client's IP address are used by the statistical-mapping algorithm. This option is appropriate for server farms that serve the Internet. Client requests coming over the Internet might come from clients sitting behind proxy farms. In this case, during a single client session, client requests can come into the Network Load Balancing cluster from several source IP addresses during a session.

  Class C Affinity addresses this issue by directing all the client requests from a particular Class C network to a single Network Load Balancing host.

  There is no guarantee, however, that all of the servers in a proxy farm are on the same Class C network. If the client's proxy servers are on different Class C networks, then the affinity relationship between a client and the server ends when the client sends successive requests from different Class C network addresses.

Implementing Network Load Balancing

The following are required for applications to work with Network Load Balancing:

- They must use TCP connections or UDP data streams.
- If client data changes, applications must provide a means of synchronizing updates to client data that is shared on multiple instances across the cluster.
- If session state is an issue, applications must use single or Class C affinity or provide a means (such as a client cookie or reference to a back-end database) of

maintaining session state in order to be uniformly accessible across the cluster.

Applications that are not compatible with Network Load Balancing have one or more of the following characteristics:

- They bind to actual computer names (examples of such applications are Exchange Server and Distributed file system).
- They have files that must be continuously open for writing (examples of such applications are Exchange Server and SMTP servers).

  In a Network Load Balancing cluster, multiple instances of an application (on different cluster hosts) should not have a file simultaneously opened for writing unless the application was designed to synchronize file updates. This is generally not the case.

Configuring Network Load Balancing

You define how the cluster load-balances client requests (and its other behaviors) by using the following Network Load Balancing parameters:

- Cluster parameters (primary IP address, subnet mask, full Internet name, multicast support, remote control password, and confirm password), which specify the behavior of the cluster.
- Host parameters (host priority, initial value, dedicated IP address, and subnet mask parameters), which define how each host functions within the cluster and in load-balancing.

  Host parameters are unrelated to the scenario for which you configure the Network Load Balancing cluster.
- Port rules (port range, protocols, filtering mode, affinity, load percentage, equal load distribution, and handling priority parameters), which define how the hosts distribute the incoming requests on a port or range of ports.

  Port rules define distribution of client requests for each scenario and must match on every cluster host. If a server attempts to join the cluster with a port rule that is inconsistent with the rest of the cluster or is incorrectly specified, the server is not accepted into the cluster, and the current load distribution is unchanged. The cluster does not complete convergence while there is a host with a port-rule mismatch.

  For information about ports and a useful list of port assignments, see the appendix "TCP and UDP Port Assignments" in the *Microsoft® Windows® 2000 Server Resource Kit TCP/IP Core Networking Guide.*

For information about the basic concepts of Network Load Balancing parameters and about configuring cluster parameters, host parameters, and port rules, see Windows 2000 Network Load Balancing Help.

For information about scenario-driven guidelines to setting values, see "Implementing Network Load Balancing" earlier in this chapter.

Client requests that you do not want to load-balance are a special case. For information about how to prevent load-balancing of a class of client requests, see "Default Handling of Client Requests" later in this chapter.

Cluster Parameters

By default, Network Load Balancing operates in unicast mode to ensure full compatibility with all makes of routers. In some cases, you might want to switch to multicast mode in order to avoid the use of a second network adapter for those communications between cluster hosts that are unrelated to cluster operations (for example, for content replication). If you switch to multicast mode, be sure that your router is compatible with this mode. For more information about the unicast and multicast modes, see Windows 2000 Network Load Balancing Help.

In unicast mode, Network Load Balancing on each host causes the network adapter's media access control address to be replaced with the cluster MAC address. (The media access control address is the hardware address, as distinct from the IP address.)

In multicast mode, the Network Load Balancing driver on each host retains the network adapter's media access control address and adds a multicast media access control address.

Note Network Load Balancing multicast mode is level-2 multicast. Do not confuse it with IP multicast. The virtual IP address must not be an IP multicast address.

For more information about multicast and unicast modes in Network Load Balancing, see Windows 2000 Network Load Balancing Help.

Host Parameters

Although you do not need to change the host parameters according to scenario, you should be aware of the following considerations.

Host Priority ID

This parameter defines the host's priority in being assigned client requests. The possible values are 1 to 32, inclusive; the lower the integer, the higher the host's priority.

Gaps in the numerical sequence of Host Priority IDs are allowed. This means that if a host from the middle of the sequence of Host Priority IDs goes out of service, the cluster can continue responding to clients.

The Host Priority ID does the following:

- Gives each host a unique identifier within the cluster.
- Establishes the default handling priority among hosts for traffic that is not load-balanced by port rules.

Initial State

Usually, you set the Initial parameter so that Network Load Balancing starts when the host boots. However, if the startup of the service receiving the client traffic is delayed after boot and there are many client requests for Network Load Balancing to distribute as soon as Network Load Balancing starts, a backlog of client requests would form before startup of the service for which Network Load Balancing is handling client requests. An example would be some earlier Web servers. In this case, you can filter traffic by starting Network Load Balancing manually or from a script after starting the service that receives the client requests.

Dedicated IP Address and Subnet Mask

Because each host's dedicated IP address is for network traffic that is not related to client requests to the cluster, Network Load Balancing never filters traffic or applies port rules to this address.

The dedicated IP address is normally the first in the list of IP addresses used by TCP/IP for the host's network connection. This ensures that outbound connections from the host use the dedicated address instead of a virtual IP address for their source address. Otherwise, replies for outbound connections could be load-balanced and delivered to another host.

In certain scenarios (such as load-balancing for virtual private networks), you do not set a dedicated IP address in TCP/IP for the host's network connection. These applications require that the cluster's primary IP address be used for outbound connections. Therefore, in these scenarios you do not need to set a value for this parameter. These scenarios are covered in "Scenarios," later in this chapter.

Port Rules

Each port rule configures load-balancing for client requests that use the port or ports covered by the Port Range parameter. How you load-balance your applications is mostly defined by how you add or modify port rules, which you create on each host for any particular port range.

For information about port assignments for common applications, such as HTTP and FTP, see Windows 2000 Network Load Balancing Help. For a complete list of TCP and UDP port assignments, see "TCP and UDP Port Assignments" in the *Microsoft® Windows® 2000 Server Resource Kit TCP/IP Core Networking Guide.*

Port Range

To load-balance all client requests with a single port rule, use the default port range (0-65535). By using the default port range, you do not have to worry about which port or ports are associated with the application whose client requests you are load-balancing.

Note For Windows Load Balancing Service on clusters that are running Windows NT version 4.0, the default port range was 1-65535. Make sure to check default port-range values when you build mixed Windows NT 4.0/Windows 2000 clusters or when you perform rolling upgrades from Windows NT 4.0 to Windows 2000.

To specify a single port, enter the same port number for both the start and the end of the range — for example, 80-80.

You might need to use multiple port rules if you load-balance multiple applications with multiple policies. For example, you might set client affinity for one application but not for another.

If you use multiple port rules, make sure that for a specific port rule, the port range covers all the ports that the application uses; protocols such as FTP use more than one port.

Each port rule configures port ranges only for contiguous port numbers. Therefore, in rare circumstances (for example, if one application is associated with two noncontiguous sets of ports, between which there is an intervening port that another application uses) you might have to define more than one port rule for the application that uses the noncontiguous ports. For example, HTTP requests (and therefore most Web requests) use port 80, and Secure Sockets Layer (SSL) requests use port 443.

Protocols

Some applications (such as streaming media applications) use both TCP and UDP ports. In most scenarios, set the Protocols parameter to **Both**.

Network Load Balancing with Network Hardware Switches

As explained in Windows 2000 Network Load Balancing Help, network adapters for Network Load Balancing hosts in a single cluster must all be on the same broadcast subnet and connected to each other through either a hub or a switch.

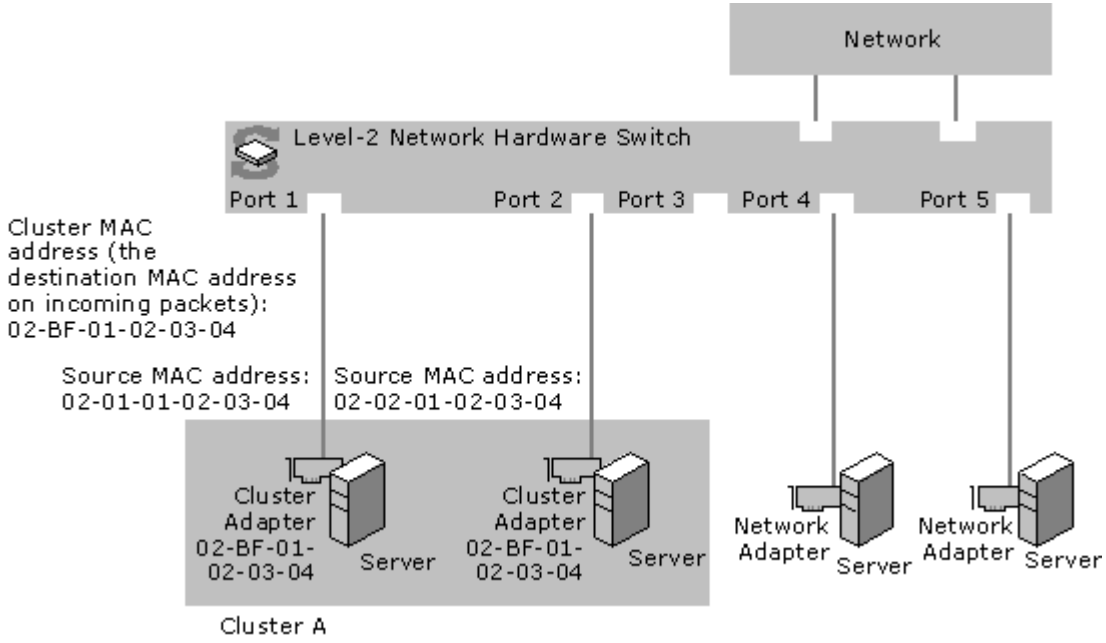
Network hardware switches mediate between a network and computers or other switches, routing packets from the network to the correct computer.

If you connect Network Load Balancing hosts with a switch, the switch must be level-2 rather than level-3 or higher, because all the hosts share the same IP address (the cluster IP address), and level-3 switches direct network packets (incoming client requests) according to the IP address of the destination computer.

In unicast mode, each host's unique media access control address is replaced with the same cluster media access control address. Identifying all the hosts with one media access control address makes it possible to distribute incoming client requests (network packets) to all the hosts.

However, most level-2 switches require that each port be associated with a unique source media access control address. Network Load Balancing addresses this requirement in unicast mode by enabling the MaskSourceMAC feature by default.

When MaskSourceMAC is enabled, Network Load Balancing masks the source media access control address for outgoing packets so that for each port the switch continues to see a unique source media access control address. This satisfies the switch's requirement that each port be associated with a unique media access control address. Figure 19.1 shows a representative configuration of a Network Load Balancing cluster in unicast mode, with MaskSourceMAC enabled, and attached to a level-2 switch.



If your browser does not support inline frames, click here to view on a separate page.

Figure 19.1 Network Load Balancing cluster running in unicast mode with MaskSourceMAC enabled

Masking the cluster media access control address on outgoing packets prevents the switch from associating the cluster media access control address with a single port. When a client request (which contains the cluster media access control address) enters the switch, the switch does not recognize the media access control address in the packet and so sends the packet to all ports. This is called "switch flooding."

In unicast mode, Network Load Balancing induces switch flooding by design, so that packets sent to the cluster's virtual IP address go to all the cluster hosts. Switch flooding is part of the Network Load Balancing strategy of obtaining the best throughput for any specific load of client requests.

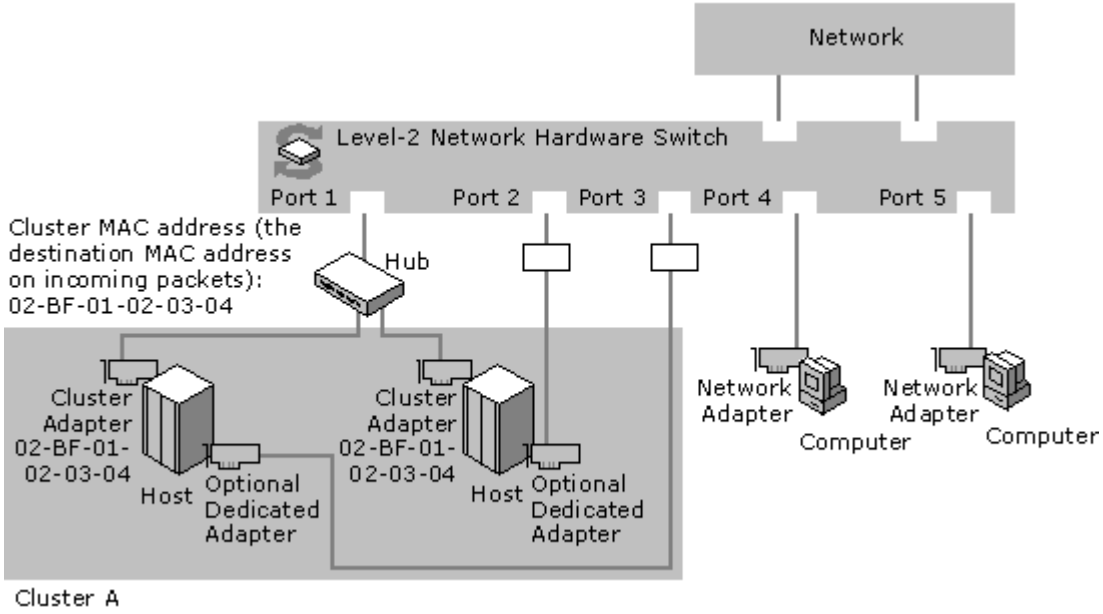
If, however, the cluster shares the switch with other (noncluster) computers or other clusters, switch flooding can add to the other computers' network overhead by including them in the flooding.

You can avoid flooding noncluster computers by putting a network hub between the switch and the Network Load Balancing cluster hosts, and then disabling the MaskSourceMAC feature. The hub delivers each packet to every host, and the switch associates the cluster media access control address with a single port, satisfying the switch's requirement that each port be associated with a unique media access control address.

Placing the Network Load Balancing hosts downstream (toward the cluster) from a hub does not reduce the bandwidth for downstream packets. However, all upstream (from the cluster) traffic must flow through the hub. To optimize use of the hub, you can also connect each host's second network adapter back to another port in the switch for outbound packets, as shown in Figure 19.2. This has the following benefits:

- Routing outbound packets through network adapters that are not attached to the hub improves use of the hub's capacity.
- Use of the capacity for multiple upstream pipes from the switch to the network is improved, because multiple cluster hosts can simultaneously send traffic to different upstream pipes.

- Using two network adapters to separate each cluster host's inbound and outbound network traffic improves the cluster hosts' handling of network traffic.



If your browser does not support inline frames, click here to view on a separate page.

Figure 19.2 Network Load Balancing cluster running in unicast mode, with MaskSourceMAC disabled

Finally, if you choose not to use a hub as described here (for example, if the Network Load Balancing cluster does not share the level-2 switch with any other computers), you can put a level-3 switch upstream from the level-2 switch to prevent switch flooding of other interconnected level-2 switches.

Scenarios

The scenarios in this section are some representative configurations for which you might use a Network Load Balancing cluster. Each scenario includes information about configuring the cluster.

IIS Server (Web Farm)

An IIS Server Web farm is the most common scenario for Network Load Balancing. The scenarios "Servicing Multiple Web Sites (Multihoming)" and "Servicing a Web Site with Active Server Pages," which are discussed later in this chapter, are variations on the theme.

Port-rule Settings

Filtering Mode: Multiple Hosts.

Affinity: None, unless session state must be maintained; to maintain session state, either Single or Class C.

Load Weight/Equal load distribution: Equal. However, if one host has a greater capacity than the others, you can use this parameter to increase its share of the client requests.

Servicing Multiple Web Sites (Multihoming)

This scenario is a variation on the IIS Server Web farm scenario. The port-rule settings are the same; the scenarios differ in how you configure additional IP addresses for the cluster in the **Advanced TCP/IP Settings** dialog box.

For more information about port-rule settings, see "IIS Server (Web Farm)," earlier in this chapter.

Note that you enter additional virtual IP addresses in the **Advanced TCP/IP Settings** dialog box, not in the **Network Load Balancing Properties** dialog box.

Before performing the following procedure, be sure to enter the host's dedicated IP address and the cluster's primary IP address in the **Network Load Balancing Properties** dialog box and in the **Advanced TCP/IP Settings** dialog box, as described in Windows 2000 Advanced Server Help.

To host multiple Web sites with different IP addresses on a Network Load Balancing cluster

1. Click the **Start** menu, point to **Settings**, and then open **Network and Dial-up Connections**.
2. Click the **Local Area Connection** for which Network Load balancing is enabled, and then, in the **Local Area Connection Status** dialog box, click **Properties**.
3. In the **Local Area Connection Properties** dialog box, click **Internet Protocol (TCP/IP)**, and then click **Properties**.
4. Click **Advanced**.

   The list under **IP Addresses** should already contain the host's dedicated IP address and the cluster's primary IP address. This cluster IP address corresponds to the cluster's primary IP address that you entered in the **Network Load Balancing Properties** dialog box under **Cluster parameters**.
5. For each additional virtual IP address that your cluster needs in order to run a multihomed server, click **Add**, and then enter the cluster IP address, followed by the appropriate subnet mask.

Servicing a Web Site with Active Server Pages

Web sites that use Active Server Pages (ASP) can maintain session state across client connections. Network Load Balancing helps preserve client access to session information by ensuring that all TCP/IP connections from a single client are directed to the same cluster host. To do so, set Affinity to either Single or Class C.

There are, however, situations in which a client can connect with one cluster host, and then have subsequent connections load-balanced to different hosts. Such situations include the following:

- A host is added to the cluster, and Network Load Balancing load-balances new connections from this client to the host.

  Note that existing connections are unaffected.
- Multiple client-side proxy servers cause multiple connections from the same client to originate from different IP addresses.

If either of the preceding situations arises, ASP applications must provide a means to retrieve and manage session state even if a client connects to multiple cluster hosts as part of a single session. The following are two strategies for addressing this issue:

- Use a means at the ASP level, such as a cookie, to retrieve the ASP client state across the Network Load Balancing cluster nodes.
- Encapsulate in a client-side cookie the state from a specific client request. The cookie gives the server the context for subsequent client requests. This solution works only if there is a relatively small amount of state associated with each client transaction. As state grows larger, it becomes increasingly difficult to have the client forward a large cookie to the server with every request.

For more information about port-rule settings, see "IIS Server (Web Farm)," earlier in this chapter.

Servicing a Web Site That Uses Secure Sockets Layer

If you service a Web site that uses Secure Sockets Layer (SSL) to build secure connections with clients, whether in conjunction with unsecured connections or not, create the following port rules.

Port-rule Settings

Port Range: 443, or use the default (0-65535).

Filtering Mode: Multiple Hosts

Affinity: Single

If you are load-balancing for a particular Class C address space, such as a corporate proxy array, or firewall, set Affinity to Class C.

Load Weight/Equal load distribution: Use the default.

Creating a Virtual Private Network

This scenario's discussion applies to setting up a virtual private network (VPN) by using the Point-to-Point Tunneling Protocol (PPTP).

When using Network Load Balancing with VPN servers to load-balance PPTP clients, it is important to configure the TCP/IP properties correctly to ensure compatibility with clients running earlier versions of Windows (such as Microsoft® Windows® 98 and Windows NT 4.0). To do this, assign only a single virtual IP address to the network adapter used by Network Load Balancing, and do not assign another IP address on any network adapter on this subnet. This restriction does not apply for Windows 2000 clients. Assigning only a single virtual IP address to the network adapter used by Network Load Balancing ensures that network traffic returning from the host to the client originates from the virtual IP address to which the client sent the request.

Set bindings so that Network Load Balancing is enabled for the cluster network adapter (the network adapter with the cluster's virtual address).

Note If a particular host fails, client sessions handled by that host handle will also break. Clients are prompted to log on again; their new session is handled by one of the remaining hosts.

Port-rules Settings

To provide load-balancing for virtual private network clients, use the default port rule for all hosts, as follows:

Port Range: Set the range to 0-65535, (the default). Setting the range to the default covers all the ports, so the port rule remains valid even if there is a change in the port numbers you want to cover.

Filtering Mode: Accept the default.

Affinity: Single (default).

Load Weight/Equal load distribution: Accept the default.

Streaming Media

You can use Network Load Balancing to distribute client requests among several streaming media servers.

If you add a host to the cluster, the statistical-mapping algorithm in Network Load Balancing maps some clients to the new host. Because Network Load Balancing cannot detect the start and termination of streams that use the UDP protocol, active streams for clients that are mapped to the new host mid-stream are interrupted. (This behavior does not occur for streams that use the TCP protocol.) As much as possible, add hosts to the cluster only at times that minimize possible disruptions to clients.

Port-rules Settings

Port Range: 0–65535

Filtering Mode: Multiple Hosts

Affinity: Single

Load Weight/Equal load distribution: Equal

Single-Server Failover Support

Although you can use Network Load Balancing to provide failover support for applications, managing the application as a resource in a server cluster is the preferred solution. However, if you choose to achieve failover support with Network Load Balancing, this section describes how.

In this scenario, start the application on every host to which the cluster traffic can fail over.

In all scenarios, Network Load Balancing does not restart the application on failover. It assumes that an instance of the application is running on each host in the cluster.

For Network Load Balancing to provide single-server failover support for a specific application, the files that the application uses must be simultaneously accessible to all hosts that run the application. These files normally reside on a back-end file server. Some applicationsrequire that these files be continuously open exclusively by one instance of the group; in a Network Load Balancing cluster, you cannot have two instances of a single file open for writing. These failover issues are addressed by server clusters, which run the Cluster service.

Other applications open files only on client request. For these applications, providing single-server failover support in a Network Load Balancing cluster works well. Again, the files must be visible to all cluster hosts. You can accomplish this by placing the files on a back-end file server or by replicating them across the Network Load Balancing cluster.

There are two alternatives for configuring the port rules for single-server failover support:

- Use no port rules.

  All the traffic goes to the host with the highest priority (the Host Priority ID with the lowest value). If that host fails, all the traffic switches to the host with the next-highest priority.
- For each application for which you're configuring single-server failover support, create a different port rule for the application's port range, in which:
  - Filtering Mode is set to Single.
  - Handling priorities are set according to the desired failover priority across the cluster hosts.

  This option overrides the Host Priority IDs with handling priorities for each application's port range. With this configuration, you can run two single-server applications on separate hosts and fail in opposite directions.

  For example, if applications Red and Blue are assigned Handling Priority IDs as indicated in Table 19.1, the applications will run on different hosts and fail over to different secondary hosts.

Table 19.1 Hypothetical Assignment of Handling Priority IDs

Host	Application Red's Port	Application Blue's Port
Host A	Handling Priority 1	Handling Priority 2
Host B	Handling Priority 2	Handling Priority 1

Port-rule Settings

Filtering Mode: Single host.

Affinity: Not available when filtering mode is single host.

Load Weight/Equal load distribution: Not available when filtering mode is single host.

Handling Priority: See the application issues discussion for this scenario.

Default Handling of Client Requests

Network Load Balancing is designed so that network traffic is not affected for the virtual IP address(es) of applications that are not being load-balanced. All traffic not explicitly load-balanced with port rules is sent to the default host. For example, incoming Telnet requests for the Virtual IP addresses are handled by the default host.

Therefore, if you do not want to load-balance some traffic for the virtual IP address, you do not define a port rule for it; the default host then handles all the traffic for that address. However, the default port range (0-65535) covers all ports, so you need to make sure that the port range for any port rules you define does not include ports associated with applications whose client requests you do not want to load-balance.

Note Undesired load-balancing is not an issue for dedicated IP addresses. Incoming network traffic for the dedicated IP address on each host is never affected by Network Load Balancing.

Wlbs Display Command

The **wlbs display** command-line command provides much valuable information about the Network Load Balancing host on which it is carried out. It is intended for use in diagnosing problems with NLB configurations. The information comprises the following areas:

- Details of the current cluster configuration.

  This section includes the current values for the cluster parameters, host parameters, and port rules, and other registry values for the host.

  For more information about the cluster parameters, host parameters, and port rules, see Windows 2000 Network Load Balancing Help.
- Last 10 event messages.

  The output for each message is the output of the event log. Each message is followed by two lines of hexadecimal numbers.

  For the sake of brevity, all the event messages but one were deleted from the following sample output.
- IP configuration of the host operating system and network adapter.

  This is the output of the **ipconfig** command.
- Current state of the cluster.

  This is the membership of the cluster as of the last convergence.

The following is an example of the output of the **wlbs display** command:

```
D:\>wlbs display
WLBS Cluster Control Utility V2.3. (c) 1997-99 Microsoft Corporation

=== Configuration: ===

Current time = Thu Jul 01 13:02:23 1999
ParametersVersion = 4
VirtualNICName = \Device\{31270FF0-11FD-11D3-8B19- 02BFAC1FF0AB}
AliveMsgPeriod = 1000
AliveMsgTolerance = 5
NumActions = 50
NumPackets = 100
NumAliveMsgs = 66
ClusterNetworkAddress = 02-bf-00-00-00-00
ClusterName = cluster.reskit.com
```

```
ClusterIPAddress = 0.0.0.0
ClusterNetworkMask = 255.255.248.0
DedicatedIPAddress = 172.31.240.170
DedicatedNetworkMask = 255.255.248.0
HostPriority = 1
ClusterModeOnStart = ENABLED
LicenseKey =
DescriptorsPerAlloc = 512
MaxDescriptorAllocs = 512
ScaleSingleClient = 0
NBTSupportEnable = 1
MulticastSupportEnable = 0
MulticastARPEnable = 1
MaskSourceMAC = 1
IPToMACEnable = 1
ConnectionCleanupDelay = 300000
RemoteControlEnabled = 1
RemoteControlUDPPort = 2504
RemoteControlCode = 0x0
RemoteMaintenanceEnabled = 0x0
CurrentVersion = V2.3
InstallDate = 0x36D75CDA
VerifyDate = 0x1C9E7553
NumberOfRules = 1
PortRules
Start End Prot Mode Pri Load Affinity
1 65535 Both Multiple Equal S

00000001 0000FFFF 6FFFF001 00000002 00000003 00000001 00010001 00000000

=== Event messages: ===

#3852 ID: 0x40070024 Type: 4 Category: 0 Time: Fri Jun 11 15:29:16 1999
WLBS : registry parameters successfully reloaded.
00000070 00420042 00490041 002D004E 00310057 004E002D 00350054 00000000
00370031 002E0032 00310033 0032002E 00300034 0031002E 00310037

=== IP configuration: ===

Windows NT IP Configuration

Host Name . . . . . : NLB-HOST-1
Primary Domain Name . . . . : testclus.reskit.com
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : Yes
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

Adapter Domain Name . . . . :
DNS servers . . . . . :
```

```
Description . . . . . . . . : 3Com EtherLink XL 10/100 PCI TX NIC (3C905
B-TX)
Physical Address. . . . . . : 00-C0-4F-57-0E-34
DHCP Enabled. . . . . . . . : No
IP Address. . . . . . . . . : 0.0.0.0
Subnet Mask . . . . . . . . : 0.0.0.0
IP Address. . . . . . . . . : 172.31.240.170
Subnet Mask . . . . . . . . : 255.255.248.0
Default Gateway . . . . . . :

=== Current state: ===

Host 1 is stopped and does not know convergence state of the cluster.
```

Changing Network Load Balancing Resource Limits in the Registry

You can tune Network Load Balancing performance by manually modifying the following entries in the registry in the HKEY_LOCAL_MACHINE \SYSTEM \CurrentControlSet \Services \WLBS \Parameters subkey.

- **AliveMsgPeriod**

  Determines the period (in milliseconds) between Network Load Balancing heartbeat messages broadcast by each host.

  Default value: 1000 (one second)

  Possible range: 100–10000

- **AliveMsgTolerance**

  Determines the number of **AliveMsgPeriod** periods to wait after the last message from a host before the host is declared offline and the cluster performs convergence.

  Default value: 5

  Possible range: 5–100

- **DescriptorsPerAlloc**

  Determines the number of connection descriptors allocated at a time. Connection descriptors are used to track TCP connections.

  Default value: 512

  Possible range: 16–1024

- **MaxDescriptorAllocs**

  Determines the maximum number of times that connection descriptors can be allocated (this value limits the maximum memory footprint of Network Load Balancing).

  Default value: 512

  Possible range: 1–1024

- **NumActions**

  An internal Network Load Balancing entry. Increase the value of this entry only if you encounter an event log message that advises you to do so.

  Default value: 50

  Possible range: 5–500

- **NumPackets**

  An internal Network Load Balancing entry. Increase the value of this entry only if you encounter an event log message that advises you to do so.

  Default value: 100

  Possible range: 5–500

- **NumAliveMsgs**

  An internal Network Load Balancing entry. Increase the value of this entry only if you encounter an event log message that advises you to do so.

  Default value: 10

  Possible range: 5–500

- **MaskSourceMAC**

  Enables masking of the Source media access control address.

  If the host is connected to a switch when Network Load Balancing is running in unicast mode, set the value of **MaskSourceMAC** to 1 (the default). If the Network Load Balancing host is running in unicast mode and is attached to a hub that is connected to a switch, set the value of this entry to 0. If Network Load Balancing is running in multicast mode, this setting has no effect.

  Default value: 1

  Possible range: 0–1

- **RemoteControlUDPPort**

  Determines the UDP port that is used by Network Load Balancing to receive remote control messages. Note that for backwards compatibility, Network Load Balancing (and, on Windows NT 4.0, Windows Load Balancing Service) automatically listens to port 1717. If you decide to firewall port 2504 to block remote control messages, you also need to firewall port 1717.

  Default value: 2504

- **NetmonAliveMsgs**

  Determines whether Network Monitor (NetMon) captures Network Load Balancing heartbeat messages on the local host.

  To allow NetMon to capture Network Load Balancing heartbeat messages on the local host, set the value of this entry to 1. To get the best performance, leave the value of this entry at its default value of 0.

Additional Resources

- For information about setting up, installing, and operating a server cluster, see Windows 2000 Advanced Server Help.
- For more information about the Windows Clustering API, see the Microsoft Platform SDK link on the Web Resources page at http://windows.microsoft.com/windows2000/reskit/webresources .
