# IOS Server Load Balancing

**Feature History**

| Release | Modification |
|---------|--------------|
| 12.0(7)XE | This feature was introduced with support for the following platforms: |
| | • Catalyst 6000 Family Switches with Supervisor Engine 1 |
| | • Cisco 7200 Series Routers |
| | The following functions were provided: |
| | • Algorithms for Server Load Balancing, page 5 |
| | • Automatic Server Failure Detection, page 6 |
| | • Automatic Unfail, page 6 |
| | • Bind ID Support, page 7 |
| | • Client-Assigned Load Balancing, page 7 |
| | • Delayed Removal of TCP Connection Context, page 7 |
| | • Dynamic Feedback Protocol for IOS SLB, page 8 |
| | • Maximum Connections, page 10 |
| | • Port-Bound Servers, page 12 |
| | • Slow Start, page 15 |
| | • Sticky Connections, page 15 |
| | • SynGuard, page 15 |
| | • TCP Session Reassignment, page 16 |
| 12.1(1)E | The following functions were added: |
| | • Alternate IP Addresses, page 6 |
| | • Content Flow Monitor Support, page 7 |
| | • Network Address Translation (NAT) and Session Redirection, page 11—Server NAT |
| | • Redundancy Enhancements, page 13—Stateless Backup |
| | • Transparent Webcache Load Balancing, page 16 |

| 12.1(2)E | The following functions were added: |
|---|---|
| | - Probes, page 12—HTTP Probes |
| | - Network Address Translation (NAT) and Session Redirection, page 11—Server and Client NAT |
| | - Redundancy Enhancements, page 13—Stateless and Stateful Backup |
| 12.1(3a)E | The following functions were added: |
| | - Firewall Load Balancing, page 8 |
| | - Probes, page 12—HTTP and Ping Probes |
| | - Protocol Support, page 13 |
| | - Redundancy Enhancements, page 13—Stateless and Stateful Backup, and Active Standby |
| | - WAP Load Balancing, page 16 |
| 12.1(5a)E | The following functions were added: |
| | - Avoiding Attacks on Server Farms and Firewall Farms, page 6 |
| | - Probes, page 12—HTTP, Ping, and WSP Probes |
| 12.1(5)T | The Cisco IOS Release 12.1(1)E feature was migrated to Cisco IOS Release 12.1(5)T, supporting Cisco 7200 Series Routers only. |
| 12.1(7)E | Support for the following platforms was added: |
| | - Cisco 7100 Series Routers |
| | The following functions were added: |
| | - Multiple Firewall Farm Support, page 10 |
| | - Route Health Injection, page 14 |
| 12.1(8a)E | Support for the following platform was added: |
| | - Catalyst 6000 Family Switches with Supervisor Engine 2 |
| | The following functions were added: |
| | - Backup Server Farms, page 7 |
| | - DFP Agent Subsystem Support, page 8 |
| 12.1(9)E | The following functions were added: |
| | - GPRS Load Balancing, page 9 |

This document describes the Cisco IOS Server Load Balancing (SLB) feature in Cisco IOS Release 12.1(9)E. It includes the following sections:

# Overview of the IOS SLB Feature

The IOS SLB feature is an IOS-based solution that provides IP server load balancing. Using the IOS SLB feature, you can define a *virtual server* that represents a group of *real servers* in a cluster of network servers known as a *server farm*. In this environment, the clients connect to the IP address of the virtual server. When a client initiates a connection to the virtual server, the IOS SLB function chooses a real server for the connection based on a configured *load-balancing algorithm*.

> **Note**   IOS SLB does not support load balancing of flows between clients and real servers that are on the same local area network (LAN) or virtual LAN (VLAN). The packets being load balanced cannot enter and leave the load-balancing device on the same interface.

IOS SLB also provides firewall load balancing, which balances flows across a group of *firewalls* called a *firewall farm*.

Figure 1 illustrates a logical view of a simple IOS SLB network.

*Figure 1     Logical View of IOS SLB*
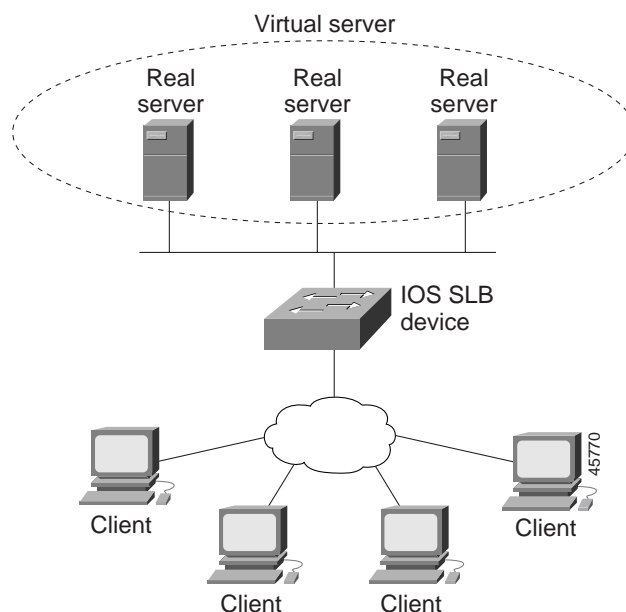
# Functions and Capabilities

This section describes the following functions and capabilities provided by IOS SLB.

✎

**Note** Some IOS SLB functions are specific to one platform and are not described in this feature module. For information about those functions, refer to the appropriate platform-specific documentation.

# Algorithms for Server Load Balancing

IOS SLB provides the following load-balancing algorithms:

You can specify one of these algorithms as the basis for choosing a real server for each new connection request that arrives at the virtual server.

## Weighted Round Robin

The weighted round robin algorithm specifies that the real server used for a new connection to the virtual server is chosen from the server farm in a circular fashion. Each real server is assigned a weight, $n$, that represents its capacity to handle connections, as compared to the other real servers associated with the virtual server. That is, new connections are assigned to a given real server $n$ times before the next real server in the server farm is chosen.

For example, assume a server farm comprised of real server ServerA with $n = 3$, ServerB with $n = 1$, and ServerC with $n = 2$. The first three connections to the virtual server are assigned to ServerA, the fourth connection to ServerB, and the fifth and sixth connections to ServerC.

**Note** Assigning a weight of $n=1$ to all of the servers in the server farm configures the IOS SLB device to use a simple round robin algorithm.

GPRS load balancing requires the weighted round robin algorithm. A server farm that uses weighted least connections can be bound to a virtual server providing GPRS load balancing, but you cannot place the virtual server **INSERVICE**. If you try to do so, IOS SLB issues an error message.

## Weighted Least Connections

The weighted least connections algorithm specifies that the next real server chosen from a server farm for a new connection to the virtual server is the server with the fewest active connections. Each real server is assigned a weight for this algorithm, also. When weights are assigned, the server with the fewest connections is based on the number of active connections on each server, and on the relative capacity of each server. The capacity of a given real server is calculated as the assigned weight of that server divided by the sum of the assigned weights of all of the real servers associated with that virtual server, or $n_1/(n_1+n_2+n_3...)$.

For example, assume a server farm comprised of real server ServerA with $n = 3$, ServerB with $n = 1$, and ServerC with $n = 2$. ServerA would have a calculated capacity of $3/(3+1+2)$, or half of all active connections on the virtual server, ServerB one-sixth of all active connections, and ServerC one-third of all active connections. At any point in time, the next connection to the virtual server would be assigned to the real server whose number of active connections is farthest below its calculated capacity.

**Note** Assigning a weight of $n=1$ to all of the servers in the server farm configures the IOS SLB device to use a simple least-connection algorithm.

GPRS load balancing does not support the weighted least connections algorithm.

# Alternate IP Addresses

IOS SLB enables you to telnet to the load-balancing device using an alternate IP address. To do so, use either of the following methods:

- Use any of the interface addresses to telnet to the load-balancing device.
- Define a secondary IP address to telnet to the load-balancing device.

This function is similar to that provided by the LocalDirector (LD) Alias command.

# Automatic Server Failure Detection

IOS SLB automatically detects each failed Transmission Control Protocol (TCP) connection attempt to a real server, and increments a failure counter for that server. (The failure counter is not incremented if a failed TCP connection from the same client has already been counted.) If a server's failure counter exceeds a configurable *failure threshold*, the server is considered *out of service* and is removed from the list of active real servers.

# Automatic Unfail

When a real server fails and is removed from the list of active servers, it is assigned no new connections for a length of time specified by a configurable *retry timer*. After that timer expires, the server is again eligible for new virtual server connections and IOS SLB sends the server the next qualifying connection. If the connection is successful, the failed server is placed back on the list of active real servers. If the connection is unsuccessful, the server remains out of service and the retry timer is reset.

# Avoiding Attacks on Server Farms and Firewall Farms

IOS SLB relies on a site's firewalls to protect the site from attacks. In general, IOS SLB is no more susceptible to direct attack than is any switch or router. However, a highly secure site can take the following steps to enhance its security:

- Configure real servers on a private network to keep clients from connecting directly to them. This ensures that the clients must go through IOS SLB to get to the real servers.
- Configure input access lists on the access router or on the IOS SLB device to deny flows from the outside network aimed directly at the interfaces on the IOS SLB device. That is, deny *all* direct flows from unexpected addresses.
- To protect against attackers trying to direct flows to real or nonexistent IP addresses in the firewall subnet, configure the firewalls in a private network.
- Configure firewalls to deny *all* unexpected flows targeted at the firewalls, especially flows originating from the external network.

# Backup Server Farms

A backup server farm is a server farm that can be used when none of the real servers defined in a primary server farm is available to accept new connections. When configuring backup server farms, keep in mind the following considerations:

- A server farm can act as both primary and backup at the same time.

- The same real server cannot be defined in both primary and backup at the same time.

- Both primary and backup require the same NAT configuration (none, client, server, or both). In addition, if NAT is specified, both server farms must use the same NAT pool.

# Bind ID Support

The bind ID allows a single physical server to be bound to multiple virtual servers and report a different weight for each one. Thus, the single real server is represented as multiple instances of itself, each having a different bind ID. DFP uses the bind ID to identify for which instance of the real server a given weight is specified. The bind ID is needed only if you are using DFP.

GPRS load balancing does not support bind IDs.

# Client-Assigned Load Balancing

Client-assigned load balancing allows you to limit access to a virtual server by specifying the list of client IP subnets that are permitted to use that virtual server. With this feature, you can assign a set of client IP subnets (such as internal subnets) connecting to a virtual IP address to one server farm or firewall farm, and assign another set of clients (such as external clients) to a different server farm or firewall farm.

GPRS load balancing does not support client-assigned load balancing.

# Content Flow Monitor Support

IOS SLB supports the Cisco Content Flow Monitor (CFM), a web-based status monitoring application within the CiscoWorks2000 product family. You can use CFM to manage Cisco server load-balancing devices. CFM runs on Windows NT and Solaris workstations, and is accessed using a web browser.

# Delayed Removal of TCP Connection Context

Because of IP packet ordering anomalies, IOS SLB might "see" the termination of a TCP connection (a finish [FIN] or reset [RST]) followed by other packets for the connection. This problem usually occurs when there are multiple paths that the TCP connection packets can follow. To correctly redirect the packets that arrive after the connection is terminated, IOS SLB retains the TCP connection information, or *context*, for a specified length of time. The length of time the context is retained after the connection is terminated is controlled by a configurable *delay timer*.

# DFP Agent Subsystem Support

IOS SLB supports the DFP Agent Subsystem feature, which enables client subsystems other than IOS SLB to act as DFP agents. With the DFP Agent Subsystem, you can use multiple DFP agents from different client subsystems at the same time.

For more information about the DFP Agent Subsystem, see the *DFP Agent Subsystem* feature module.

# Dynamic Feedback Protocol for IOS SLB

With IOS SLB Dynamic Feedback Protocol (DFP) support, a *DFP manager* in a load-balancing environment can initiate a TCP connection with a *DFP agent*. Thereafter, the DFP agent collects status information from one or more real host servers, converts the information to relative weights, and reports the weights to the DFP manager. The DFP manager factors in the weights when load balancing the real servers. In addition to reporting at user-defined intervals, the DFP agent sends an early report if there is a sudden change in a real server's status.

The weights calculated by DFP override the static weights you define using the **weight (server farm)** command. If DFP is removed from the network, IOS SLB reverts to the static weights.

You can define IOS SLB as a DFP manager, as a DFP agent for another DFP manager (such as DistributedDirector), or as both at the same time. In such a configuration, IOS SLB sends periodic reports to DistributedDirector, which uses the information to choose the best server farm for each new connection request. IOS SLB then uses the same information to choose the best real server within the chosen server farm.

DFP also supports the use of multiple DFP agents from different client subsystems (such as IOS SLB and GPRS) at the same time.

In GPRS load balancing, you can define IOS SLB as a DFP manager and define a DFP agent on each GGSN in the server farm, and the DFP agent can report the weights of the GGSNs. The DFP agents calculate the weight of each GGSN based on CPU utilization, processor memory, and the maximum number of PDP contexts (mobile sessions) that can be activated for each GGSN. As a first approximation, DFP calculates the weight as the number of existing PDP contexts divided by the maximum allowed PDP contexts:

**(existing PDP contexts)/(maximum PDP contexts)**

Maximum PDP contexts are specified using the **gprs maximum-pdp-context-allowed** command, which defaults to 1000 PDP contexts. If you accept the default value, DFP might calculate a very low weight for the GGSN:

**(existing PDP contexts)/1000 = Low GGSN weight**

Keep this calculation in mind when specifying maximum PDP contexts using the **gprs maximum-pdp-context-allowed** command.

# Firewall Load Balancing

As its name implies, firewall load balancing enables IOS SLB to balance flows to firewalls. Firewall load balancing uses a load-balancing device on each side of a group of firewalls (called a *firewall farm*) to ensure that the traffic for each flow travels to the same firewall, ensuring that the security policy is not compromised.

You can configure more than one firewall farm in each load-balancing device.

Layer 3 firewalls, which have IP-addressable interfaces, are supported by IOS SLB firewall load balancing if they are subnet-adjacent to the firewall load-balancing device and have unique MAC addresses. The device does not modify the IP addresses in the user packet. To send the packet to the chosen firewall, the device determines which interface to use and changes the Layer 2 headers accordingly. This is the standard dispatched routing used by IOS SLB.

Layer 2 firewalls, which do not have IP addresses, are transparent to IOS SLB firewall load balancing. IOS SLB supports Layer 2 firewalls by placing them between two IP-addressable interfaces.

Whereas many Layer 3 firewalls might exist off a single Layer 3 interface on the load-balancing device (for example, a single LAN), only one Layer 2 firewall can exist off each interface.

When configuring the load-balancing device, you configure a Layer 3 firewall using its IP address, and a Layer 2 firewall using the IP address of the interface of the device on the "other side" of the firewall.

To balance flows across the firewalls in a firewall farm, IOS SLB firewall load balancing performs a *route lookup* on each incoming flow, examining the source and destination IP addresses (and optionally the source and destination TCP or User Datagram Protocol [UDP] port numbers). Firewall load balancing applies a *hash algorithm* to the results of the route lookup and selects the best firewall to handle the connection request.

**Note** IOS SLB firewall load balancing *must* examine incoming packets and perform route lookup. On Catalyst 6000 Family Switches, some additional packets might need to be examined. Firewall load balancing impacts internal (secure) side routing performance and must be considered in the complete design.

To maximize availability and resilience in a network with multiple firewalls, configure a separate equal-weight route to each firewall, rather than a single route to only one of the firewalls.

IOS SLB firewall load balancing provides the following capabilities:

- Connections initiated from either side of the firewall farm are load-balanced.

- The load is balanced among a set of firewalls—the firewall farm.

- All packets for a connection travel through the same firewall. Subsequent connections can be "sticky," ensuring that they are assigned to the same firewall.

- Probes are used to detect and recover from firewall failures.

- Redundancy is provided. Hot Standby Router Protocol (HSRP), stateless backup, and stateful backup are all supported.

- Multiple interface types and routing protocols are supported, enabling the external (Internet side) load-balancing device to act as an access router.

- Proxy firewalls are supported.

# GPRS Load Balancing

General Packet Radio Service (GPRS) is the packet network infrastructure based on the European Telecommunications Standards Institute (ETSI) Global System for Mobile Communication (GSM) phase 2+ standards for transferring packet data from the GSM mobile user to the packet data network (PDN). The Cisco gateway GPRS support node (GGSN) interfaces with the serving GPRS support node (SGSN) using the GPRS Tunneling Protocol (GTP), which in turn uses UDP/IP for transport. IOS SLB provides GPRS load balancing and increased reliability and availability for the GGSN.

Tunnel creation messages destined to the virtual GGSN IP address are redirected via Layer 2 to one of the real GGSNs using the weighted round robin load-balancing algorithm. See the "Weighted Round Robin" section on page 5 for more information about this algorithm.

The real GGSNs must be Layer 2-adjacent to the IOS SLB device. The SGSNs need not be Layer 2-adjacent to the IOS SLB device, unless you implement IOS SLB redundancy enhancements.

GPRS load balancing uses standard dispatched routing, so you must configure the real GGSNs with the virtual GGSN IP address as a loopback address, or secondary IP address. See the "Network Address Translation (NAT) and Session Redirection" section on page 11 for more information about dispatched routing. See the "Configuring Logical Interfaces" chapter of the *Cisco IOS Interface Configuration Guide* for more information about configuring the loopback address.

When configuring the network shared by IOS SLB and the GGSNs, keep the following considerations in mind:

- Specify static routes (using **ip route** commands) and real server IP addresses (using **real** commands) such that the Layer 2 information is correct and unambiguous.

- Do not configure default routes or gateways on any of the GGSNs.

- Do not configure a route to IOS SLB's virtual server on the GGSNs. Doing so can prevent messages from reaching GTP.

- Choose subnets carefully, using one of the following methods:

  – Do not overlap virtual template address subnets.

  – Specify next hop addresses to real servers, not to interfaces on those servers.

In GPRS load balancing, IOS SLB knows when a PDP context is established, but it does not know when PDP contexts are cleared, and therefore it cannot know the number of open PDP contexts for each GGSN. Use DFP to calculate GPRS load-balancing weights dynamically. See the "Dynamic Feedback Protocol for IOS SLB" section on page 8 for more information about DFP.

If you have enabled Cisco Express Forwarding (CEF) on a GGSN, you must identify the IP address of the GGSN virtual server to CEF. If you have *not* enabled CEF on the GGSN, do not perform this task. See the "Identifying the GGSN Virtual Server to CEF" section on page 36 for more details.

# Maximum Connections

IOS SLB allows you to configure *maximum connections* for server and firewall load balancing.

- For server load balancing, you can configure a limit on the number of active connections that a real server is assigned. If the maximum number of connections is reached for a real server, IOS SLB automatically switches all further connection requests to another server until the connection number drops below the specified limit.

- For firewall load balancing, you can configure a limit on the number of active TCP or UDP connections that a firewall farm is assigned. If the maximum number of connections is reached for the firewall farm, new connections are dropped until the connection number drops below the specified limit.

# Multiple Firewall Farm Support

You can configure more than one firewall farm in each load-balancing device.

# Network Address Translation (NAT) and Session Redirection

Cisco IOS NAT, RFC 1631, allows unregistered "private" IP addresses to connect to the Internet by translating them into *globally registered* IP addresses. Cisco IOS NAT also increases network privacy by hiding internal IP addresses from external networks.

IOS SLB can operate in one of two session redirection modes:

- Dispatched mode—the virtual server address is known to the real servers; you must configure the virtual server IP address as a loopback address, or secondary IP address, on each of the real servers. IOS SLB redirects packets to the real servers at the media access control (MAC) layer. Since the virtual server IP address is not modified in dispatched mode, the real servers must be Layer 2-adjacent to IOS SLB, or intervening routers might not be able to route to the chosen real server.

    See the "Configuring Logical Interfaces" chapter of the *Cisco IOS Interface Configuration Guide* for more information about configuring the loopback address.

- Directed mode—the virtual server can be assigned an IP address that is not known to any of the real servers. IOS SLB translates packets exchanged between a client and real server, translating the virtual server IP address to a real server IP address through NAT.

IOS SLB supports the following types of NAT:

- **Server NAT**—By replacing the virtual server IP address with the real server IP address (and vice versa):

    - Servers can be many hops away from the load-balancing device.

    - Intervening routers can route to them without requiring tunnelling.

    - Loopback and secondary interfaces are not required on the real server.

    - The real server need not be Layer 2-adjacent to IOS SLB.

    A less common form of server NAT is *server port translation*, which involves replacement of a virtual server port. Server port translation does not require server IP address translation, but the two translations can be used together.

    > **Note** If an IP address is configured as a real IP address for a NAT virtual server, you cannot balance connection requests from that address to a different virtual server (whether NAT or dispatched) on the same load-balancing device.

- **Client NAT**—If multiple load-balancing devices are used, replacing the client IP address with an IP address associated with one of the devices results in proper routing of outbound flows to the correct device. Client NAT also requires that the ephemeral client port be modified since many clients can use the same ephemeral port. Even in cases where multiple load-balancing devices are not used, client NAT can be useful to ensure that packets from load-balanced connections are not routed around the device.

In both dispatched and directed modes, IOS SLB must track connections. Therefore, you must design your network so that there is no alternate network path from the real servers to the client that bypasses the load-balancing device.

> **Note** Both server NAT and client NAT are supported for the same connection.

IOS SLB supports FTP, firewall load balancing, and GPRS load balancing only in dispatched mode. Therefore, FTP, firewall load balancing, and GPRS load balancing cannot use NAT.

# Port-Bound Servers

When you define a virtual server, you must specify the TCP or UDP port handled by that virtual server. However, if you configure NAT on the server farm, you can also configure *port-bound servers*. Port-bound servers allow one virtual server IP address to represent one set of real servers for one service, such as Hypertext Transfer Protocol (HTTP), and a different set of real servers for another service, such as Telnet.

Packets destined for a virtual server address for a port that is not specified in the virtual server definition are not redirected.

IOS SLB supports both port-bound and non-port-bound servers, but port-bound servers are recommended.

IOS SLB firewall load balancing and GPRS load balancing do not support port-bound servers.

# Probes

IOS SLB supports HTTP probes, ping probes, and WSP probes.

HTTP and ping probes are a simple way to verify connectivity for devices being server load-balanced, and for firewalls being firewall load-balanced (even devices on the other side of a firewall).

HTTP probes also enable you to monitor applications being server load-balanced. With frequent probes, the *operation* of each application is verified, not just *connectivity* to the application.

HTTP probes do not support HTTP over Secure Socket Layer (HTTPS). That is, you cannot send an HTTP probe to an SSL server.

WSP probes detect failures in the Wireless Application Protocol (WAP) stack on port 9201.

You can configure more than one probe, in any combination of types (HTTP, ping, or WSP), for each server farm, or for each firewall in a firewall farm.

## Probes in Server Load Balancing

Probes determine the status of each real server in a server farm. *All* real servers associated with *all* virtual servers tied to that server farm are probed.

If a real server fails for *one* probe, it is failed for *all* probes. After the real server recovers, *all* probes must acknowledge its recovery before it is restored to service.

## Probes in Firewall Load Balancing

Probes detect firewall failures. *All* firewalls associated with the firewall farm are probed.

If a firewall fails for *one* probe, it is failed for *all* probes. After the firewall recovers, *all* probes must acknowledge its recovery before it is restored to service.

Make sure you configure the HTTP probe to expect status code 401, to eliminate password problems. See the description of the **expect** command in the "Command Reference" section on page 91 for more details.

Use the **ip http server** command to configure an HTTP server on the device. See the description of the **ip http server** command in the *Cisco IOS Configuration Fundamentals Command Reference* for more details.

In a transparent webcache load-balancing environment, an HTTP probe uses the real IP address of the webcache, since there is no virtual IP address configured.

## Protocol Support

IOS SLB supports the following protocols:

- Domain Name System (DNS)
- File Transfer Protocol (FTP)
- GPRS Tunneling Protocol (GTP)
- Hypertext Transfer Protocol (HTTP)
- Hypertext Transfer Protocol over Secure Socket Layer (HTTPS)
- Internet Message Access Protocol (IMAP)
- Mapping of Airline Traffic over IP, Type A (MATIP-A)
- Network News Transport Protocol (NNTP)
- Post Office Protocol, version 2 (POP2)
- Post Office Protocol, version 3 (POP3)
- RealAudio/RealVideo via HTTP
- Remote Authentication Dial-In User Service (RADIUS)
- Simple Mail Transport Protocol (SMTP)
- Telnet
- X.25 over TCP (XOT)

## Redundancy Enhancements

An IOS SLB device can represent a single point of failure, and the servers can lose their connections to the backbone, if either of the following occurs:

- The IOS SLB device fails.
- A link from a switch to the distribution-layer switch becomes disconnected.

To reduce that risk, IOS SLB supports the following redundancy options, based on HSRP:

## Stateless Backup

Stateless backup provides high network availability by routing IP flows from hosts on Ethernet networks without relying on the availability of a single Layer 3 switch. Stateless backup is particularly useful for hosts that do not support a router discovery protocol (such as the Intermediate System-to-Intermediate System [IS-IS] Interdomain Routing Protocol [IDRP]) and do not have the functionality to shift to a new Layer 3 switch when their selected Layer 3 switch reloads or loses power.

If you use stateless backup in GPRS load balancing, the SGSNs must be Layer 2-adjacent to the IOS SLB device.

## Stateful Backup

Stateful backup enables IOS SLB to incrementally backup its load-balancing decisions, or "keep state," between primary and backup switches. The backup switch keeps its virtual servers in a dormant state until HSRP detects failover; then the backup (now primary) switch begins advertising virtual addresses and filtering flows. You can use HSRP to configure how quickly the failover is detected.

Stateful backup provides IOS SLB with a one-to-one stateful or idle backup scheme. This means that only one instance of IOS SLB is handling client or server flows at a given time, and that there is at most one backup platform for each active IOS SLB switch.

GPRS load balancing does not support stateful backup.

## Active Standby

Active standby enables two IOS SLBs to load-balance the same virtual IP address while at the same time acting as backups for each other. If a site has only one virtual IP address to load balance, an access router is used to direct a subset of the flows to each IOS SLB using policy-based routing.

IOS SLB firewall load balancing does not support active standby. That is, you cannot configure two pairs of firewall load balancing devices (one pair on each side of the firewalls), with each device in each pair handling traffic and backing up its partner.

# Route Health Injection

By default, a virtual server's IP address is *advertised* (added to the routing table) when you bring the virtual server into service (using the **inservice** command). If you have a preferred host route to a website's virtual IP address, you can advertise that host route, but you have no guarantee that the IP address is available. However, you can use the **advertise** command to configure IOS SLB to advertise the host route only when IOS SLB has verified that the IP address is available. IOS SLB withdraws the advertisement when the IP address is no longer available. This function is known as *route health injection*.

Note When route health injection is configured, probes require a default route to the virtual server (specified using the **ip route 0.0.0.0 0.0.0.0** command, for example). The route is not used, but it must exist to enable the sockets code to verify that the destination can be reached, which in turn is essential for route health injection to function correctly.

# Slow Start

In an environment that uses weighted least connections load balancing, a real server that is placed in service initially has no connections, and could therefore be assigned so many new connections that it becomes overloaded. To prevent such an overload, slow start controls the number of new connections that are directed to a real server that has just been placed in service.

GPRS load balancing does not support slow start.

# Sticky Connections

When you use sticky connections, new connections from a client IP address or subnet are assigned to the same real server (for server load balancing) or firewall (for firewall load balancing) as were previous connections from that address or subnet.

IOS SLB creates sticky objects to track client assignments. The sticky objects remain in the IOS SLB database after the last sticky connection is deleted, for a user-defined period. New connections from a client are sticky if the following conditions are met:

- The real server is in either OPERATIONAL or MAXCONNS_THROTTLED state.

- The sticky timer is defined on a virtual server or on a firewall farm.

- The amount of time between the end of a previous connection from the client and the start of the new connection is within the sticky timer duration.
  OR
  A connection for the same client already exists. (That is, the connection is not the first for this client.)

Sticky connections allow you to create a sticky object for a subnet, ensuring that all flows from the subnet are sent to the same real server. (Make sure the volume of flows is not so large that it overwhelms the real server.)

Sticky connections also permit the coupling of services that are handled by more than one virtual server or firewall farm. This allows connection requests for related services to use the same real server. For example, web server (HTTP) typically uses TCP port 80, and HTTPS uses port 443. If HTTP virtual servers and HTTPS virtual servers are coupled, connections for ports 80 and 443 from the same client IP address or subnet are assigned to the same real server.

GPRS load balancing does not support sticky connections.

# SynGuard

SynGuard limits the rate of TCP start-of-connection packets (SYNchronize sequence numbers, or SYNs) handled by a virtual server to prevent a type of network problem known as a *SYN flood denial-of-service attack*. A user might send a large number of SYNs to a server, which could overwhelm or crash the server, denying service to other users. SynGuard prevents such an attack from bringing down IOS SLB or a real server. SynGuard monitors the number of SYNs handled by a virtual server at specific intervals and does not allow the number to exceed a configured *SYN threshold*. If the threshold is reached, any new SYNs are dropped.

IOS SLB firewall load balancing and GPRS load balancing do not support SynGuard.

# TCP Session Reassignment

IOS SLB tracks each TCP SYN sent to a real server by a client attempting to open a new connection. If several consecutive SYNs are not answered, or if a SYN is replied to with an RST, the TCP session is reassigned to a new real server. The number of SYN attempts is controlled by a configurable *reassign threshold*.

IOS SLB firewall load balancing and GPRS load balancing do not support TCP session reassignment.

# Transparent Webcache Load Balancing

IOS SLB can load balance port 80 flows across a cluster of transparent webcaches. To set up this function, configure the subnet IP addresses served by the transparent webcaches, or some common subset of them, as virtual servers. Virtual servers used for transparent webcache load balancing do not answer pings on behalf of the subnet IP addresses, and they do not affect traceroute.

In some cases, such as when its cache does not contain needed pages, a webcache might need to initiate its own connections to the Internet. Those connections should not be load balanced back to the same set of webcaches. To address this need, IOS SLB allows you to configure **client exclude** statements, which exclude connections initiated by the webcaches from the load-balancing scheme.

IOS SLB firewall load balancing and GPRS load balancing do not support transparent webcache load balancing.

# WAP Load Balancing

You can use IOS SLB to load balance Wireless Session Protocol (WSP) sessions among a group of WAP gateways or servers on an IP bearer network. WAP runs on top of UDP on a set of well known ports, with each port indicating a different WAP mode:

- Connectionless WSP mode (IP/UDP [9200]/WSP). In connectionless WSP mode, WSP is a simple one-request/one-response protocol in which a single server-bound packet results in a server response of one or more packets.

- Connection-oriented WSP mode (IP/UDP [9201]/WTP/WSP). In connection-oriented WSP mode, WTP handles retransmissions of WDP events, and WSP operates using a defined session bring-up/tear-down sequence. IOS SLB uses a WAP-aware finite state machine (FSM), driven by events in WSP sessions, to reassign sessions. This FSM operates only on port 9201, where the WSP sessions are not encrypted and WTP handles retransmissions.

- Connectionless secure WSP mode (IP/UDP [9202]/WTLS/WSP). This mode functions the same as connectionless WSP mode, but with security provided by WTLS.

- Connection-oriented secure WSP mode (IP/UDP [9203]/WTLS/WTP/WSP). This mode functions the same as connection-oriented WSP mode, but with security provided by WTLS.

IOS SLB uses ping probes to detect failures in the WAP load-balancing device, and WSP probes to detect failures in the WAP stack on port 9201.

# Benefits

IOS SLB shares the same software code base as Cisco IOS and has all the software features sets of Cisco IOS software. IOS SLB is recommended for customers desiring complete integration of SLB technology into traditional Cisco switches and routers.

On the Cisco Catalyst 6500 switch, IOS SLB takes advantage of hardware acceleration to forward data packets at very high speed when running in dispatched mode.

IOS SLB assures continuous, high availability of content and applications with proven techniques for actively managing servers and connections in a distributed environment. By distributing user requests across a cluster of servers, IOS SLB optimizes responsiveness and system capacity, and dramatically reduces the cost of providing Internet, database, and application services for large-, medium-, and small-scale sites.

IOS SLB facilitates scalability, availability, and ease of maintenance:

- The addition of new physical (real) servers, and the removal or failure of existing servers, can occur at any time, transparently, without affecting the availability of the virtual server.

- IOS SLB's slow start capability allows a new server to increase its load gradually, preventing failures caused by assigning the server too many new connections too quickly.

- IOS SLB supports fragmented packets and packets with IP options, buffering your servers from client or network vagaries that are beyond your control.

- IOS SLB firewall load balancing enables you to scale access to your Internet site. You can add firewalls without affecting existing connections, enabling your site to grow without impacting customers.

Using DFP enables IOS SLB to provide weights to another load-balancing system, such as DistributedDirector. IOS SLB can act as a DFP manager, receiving weights from host servers, and it can act as a DFP agent, sending weights to a DFP manager. The functions are enabled independently—you can implement either one, or both, at the same time.

Administration of server applications is easier. Clients know only about virtual servers; no administration is required for real server changes.

Security of the real server is provided because its address is never announced to the external network. Users are familiar only with the virtual IP address. You can filter unwanted flows based on both IP address and TCP or UDP port numbers. Additionally, though it does not eliminate the need for a firewall, IOS SLB can help protect against some denial-of-service attacks.

In a branch office, IOS SLB allows balancing of multiple sites and disaster recovery in the event of full-site failure, and distributes the work of load balancing.

# Restrictions

IOS SLB has the following restrictions:

- **Does not support load balancing of flows between clients and real servers that are on the same local area network (LAN) or virtual LAN (VLAN). The packets being load balanced cannot enter and leave the load-balancing device on the same interface.**

- Operates in a standalone mode and currently does not operate as a MultiNode Load Balancing (MNLB) Services Manager. The presence of IOS SLB does not preclude the use of the existing MNLB Forwarding Agent with an external Services Manager (such as the LocalDirector) in an MNLB environment.

- Does not support coordinating server load-balancing statistics among different IOS SLB instances for backup capability.

- Supports FTP, firewall load balancing, and GPRS load balancing only in dispatched mode.

- Does not support IOS SLB and Cisco Applications and Services Architecture (CASA) configured with the same virtual IP address, even if they are for different services.

- Does not support both IOS server load balancing and firewall load balancing on the same flow, nor on the same server port. You can configure both server load balancing and firewall load balancing on the same device at the same time, but they must apply to different flows (different client-server pairs). These functions *can* run on the same EPIF (for example, server load balancing on port 1 and firewall load balancing on port 2). Load-balancing the server farm after a packet exits the load-balanced firewall farm requires a separate load-balancing device.

- Does not support running both IOS SLB and the Content Switching Module (CSM) on the same switch.

- When operating in dispatched mode, real servers must be Layer 2-adjacent, tag-switched, or via generic routing encapsulation (GRE) tunnel.

- When operating in directed mode with server NAT, real servers need not be Layer 2-adjacent to IOS SLB. This allows for more flexible network design, since servers can be placed several Layer 3 hops away from the IOS SLB switch.

- The DFP agent requires a delay between hello messages of at least 3 seconds. Therefore, if your DFP manager provides a timeout specification, you must set the timeout to at least 3 seconds.

- For backup server farm support:

  - Does not support defining the same real server in both primary and backup server farms.

  - Requires the same NAT configuration (none, client, server, or both) for both primary and backup server farms. In addition, if NAT is specified, both server farms must use the same NAT pool.

  - Does not support HTTP redirect load balancing. If a primary server farm specifies a redirect virtual server, you cannot define that primary as a backup, nor can you define a backup for that primary.

- For firewall load balancing:

  - No longer limited to a single firewall farm in each load-balancing device.

  - Limited to a single active firewall load-balancing device on each side of the firewall farm. Each firewall must have its own unique MAC address and must be Layer 2-adjacent to each device. The firewalls can be connected to individual interfaces on the device, or they can all share a VLAN and connect using a single interface.

  - Requires Ethernet between each firewall load-balancing device and each firewall.

  - On each firewall load-balancing device, requires that each Layer 2 firewall be connected to a single Layer 3 (IP) interface.

  - Flows with a destination IP address on the same subnet as the configured firewall IP addresses are not load-balanced. (Such flows could be a firewall console session or other flows on the firewall LAN.)

  - Does not support the following IOS SLB functions:

    - Active standby

    - Network Address Translation (NAT)

    - Port-bound servers

    - SynGuard

- TCP session reassignment

- Transparent webcache load balancing

- For GPRS load balancing:

  – Operates in dispatched mode only.

  – Cannot load balance network-initiated PDP context requests.

  – Does not support the following IOS SLB functions:

    - Bind IDs

    - Client-assigned load balancing

    - Network Address Translation (NAT)

    - Port-bound servers

    - Slow Start

    - Stateful backup

    - Sticky connections

    - SynGuard

    - TCP session reassignment

    - Transparent webcache load balancing

    - Weighted least connections load-balancing algorithm. GPRS load balancing requires the weighted round robin algorithm. A server farm that uses weighted least connections can be bound to a virtual server providing GPRS load balancing, but you cannot place the virtual server **INSERVICE**. If you try to do so, IOS SLB issues an error message.

- For HTTP probes:

  – HTTP probes do not support HTTP over Secure Socket Layer (HTTPS). That is, you cannot send an HTTP probe to an SSL server.

- For Catalyst 6000 Family Switches:

  – Requires the Multilayer Switched Feature Card (MSFC) and the Policy Feature Card (PFC). When using redundant MSFCs in the same Catalyst 6000 Family switch, stateful backup between the two MSFCs is not supported, but stateless backup between the two MSFCs is supported.

    The term "MSFC" refers to either an MSFC1 or an MSFC2, except when specifically differentiated.

    The term "PFC" refers to either a PFC1 or a PFC2, except when specifically differentiated.

  – Requires that the Multilayer Switching (MLS) flow mode be set to **full**. For more information about how to set the MLS flow, refer to the "Configuring IP Multilayer Switching" section in the *Catalyst 6000 Family MSFC (12.0) & PFC Configuration Guide, Release 5.4*.

  – When operating in dispatched mode, real servers must be Layer 2-adjacent to IOS SLB (that is, not beyond an additional router), with hardware data packet acceleration performed by the PFC. All real servers that can be reached by a single IOS SLB device (that is, all real servers in a given server farm) be on the same VLAN. The loopback address must be configured in the real servers.

  – Requires that all firewall interfaces be on the same VLAN.

- Provides no hardware data packet acceleration in directed mode. (Hardware data packet acceleration is performed by the PFC, and in directed mode the data packets are handled by the MSFC, not the PFC.)

- Supports Native IOS only.

- Does not support GPRS load balancing in 12.1(9)E.

- For the Cisco 7100 Series and Cisco 7200 Series:

  - Provides no hardware acceleration for the IOS SLB function for either dispatched mode or directed mode.

  - Supports Cisco IOS NAT in directed mode with no hardware data packet acceleration.

  - Does not support route health injection in 12.1(9)E.

# Related Features and Technologies

- Content Flow Monitor (CFM)
- Dynamic Feedback Protocol (DFP)
- General Packet Radio Service (GPRS)
- Hot Standby Router Protocol (HSRP)
- Network Address Translation (NAT)
- Wireless Application Protocol (WAP)

# Related Documents

- *Catalyst 4840G Software Feature and Configuration Guide*
- *Cisco IOS IP Configuration Guide*
- *Cisco IOS IP Command Reference, Volume 1 of 3: Addressing and Services, Release 12.2*
- *Cisco IOS Mobile Wireless Configuration Guide*
- *Cisco IOS Mobile Wireless Command Reference*
- *Dynamic Feedback Protocol Support in Distributed Director*
- *Using Content Flow Monitor*

# Supported Platforms

- Catalyst 6000 Family Switches with Supervisor Engine 1
- Catalyst 6000 Family Switches with Supervisor Engine 2
- Cisco 7100 Series Routers
- Cisco 7200 Series Routers

# Supported Standards, MIBs, and RFCs

**Standards**

- No new or modified standards

**MIBs**

- CISCO-SLB-MIB

> ✎
>
> **Note**    Although the objects in this MIB are defined as *read-create*, you cannot use the SNMP SET command to modify them. Instead, you must use the command line to set the associated command line keywords, after which the new values are reflected in SNMP.

**RFCs**

- Cisco IOS NAT, RFC 1631

# Configuration Tasks

Configuring IOS SLB involves identifying server farms, configuring groups of real servers in server farms, and configuring the virtual servers that represent the real servers to the clients.

For configuration examples associated with these tasks, see the "Configuration Examples" section on page 41.

For a complete description of the IOS SLB commands in this section, see the "Command Reference" section on page 91. To locate documentation of other commands that appear in this section, search online using Cisco.com.

To configure IOS SLB, perform the tasks in the following sections:

- Configuring Required and Optional IOS SLB Functions, page 22 (Required)
- Configuring Firewall Load Balancing, page 28 (Optional)
- Configuring Probes, page 32 (Optional)
- Configuring DFP, page 34 (Optional)
- GPRS Load Balancing Configuration Task List, page 34 (Optional)
- Configuring NAT, page 36 (Optional)
- Stateless Backup Configuration Task List, page 37 (Optional)
- Configuring Database Entries, page 38 (Optional)
- Configuring Wildcard Searches, page 38 (Optional)
- Clearing Connections and Counters, page 39 (Optional)
- Purging Connections, page 39 (Optional)
- Monitoring and Maintaining the IOS SLB Feature, page 40

# Configuring Required and Optional IOS SLB Functions

To configure IOS SLB functions, perform the tasks in the following sections. Required and optional tasks are indicated.

- Configuring a Server Farm and Real Server, page 22 (Required)
- Configuring a Virtual Server, page 25 (Required)
- Verifying the Server Farm, page 27 (Optional)
- Verifying the Virtual Server, page 27 (Optional)
- Verifying the Clients, page 27 (Optional)
- Verifying IOS SLB Connectivity, page 28 (Optional)

## Configuring a Server Farm and Real Server

To configure an IOS SLB server farm, use the following commands beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config)# ip slb serverfarm serverfarm-name`<br>`Router(config-slb-sfarm)#` | Adds a server farm definition to the IOS SLB configuration and enters server farm configuration mode. |
| **Step 2** | `Router(config-slb-sfarm)# bindid [bind_id]` | (Optional) Specifies a bind ID on the server farm for use by DFP.<br><br>✎<br>**Note** GPRS load balancing does not support this command. |
| **Step 3** | `Router(config-slb-sfarm)# nat {client pool-name \| server}` | (Optional) Configures NAT client translation mode or NAT server address translation mode on the server farm.<br><br>✎<br>**Note** GPRS load balancing does not support this command. |

| | Command | Purpose |
|---|---|---|
| **Step 4** | `Router(config-slb-sfarm)# predictor [roundrobin | leastconns]` | (Optional) Specifies the algorithm to be used to determine how a real server is selected.<br><br>✎ **Note** GPRS load balancing requires the weighted round robin algorithm, which is the default setting.<br><br>See the following sections for more details about each algorithm:<br>• Weighted Round Robin, page 5<br>• Weighted Least Connections, page 5 |
| **Step 5** | `Router(config-slb-sfarm)# probe name` | (Optional) Associates a probe with the real server. |
| **Step 6** | `Router(config-slb-sfarm)# real ip-address [port_number]` | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode.<br><br>✎ **Note** In GPRS load balancing, specify the IP addresses (virtual template addresses) of the real servers performing the GGSN function. |
| **Step 7** | `Router(config-slb-real)# faildetect numconns number-conns [numclients number-clients]` | (Optional) Specifies the number of consecutive connection failures and, optionally, the number of unique client connection failures, that constitute failure of the real server.<br><br>In GPRS load balancing, if there is only one SGSN in your environment, specify the **numclients** keyword with a value of 1. |
| **Step 8** | `Router(config-slb-real)# maxconns number-conns` | (Optional) Specifies the maximum number of active connections allowed on the real server at one time.<br><br>✎ **Note** GPRS load balancing requires a **maxconns** value of 4294967295, which is the default value. The impact of this command in GPRS load balancing is minimal because sessions are very short-lived. |

| | Command | Purpose |
|---|---|---|
| **Step 9** | `Router(config-slb-real)# reassign threshold` | (Optional) Specifies the threshold of consecutive unacknowledged synchronizations or Create PDP requests that, if exceeded, result in an attempted connection to a different real server. <br><br> **Note**    In GPRS load balancing, you must specify a reassign threshold less than the value specified on the **gprs gtp n3-requests** command (or the equivalent command, if you are not using a Cisco SGSN). |
| **Step 10** | `Router(config-slb-real)# retry retry-value` | (Optional) Specifies the interval, in seconds, to wait between the detection of a server failure and the next attempt to connect to the failed server. |
| **Step 11** | `Router(config-slb-real)# weight weighting-value` | (Optional) Specifies the real server's workload capacity relative to other servers in the server farm. <br><br> **Note**    If you use DFP, the static weights you define using the **weight (server farm)** command are overridden by the weights calculated by DFP. If DFP is removed from the network, IOS SLB reverts to the static weights. |
| **Step 12** | `Router(config-slb-real)# inservice` | Enables the real server for use by IOS SLB. |

**Note**    When performing server load balancing and firewall load balancing together on a Catalyst 6000 Family Switch, use the **mls ip slb wildcard search rp** command to reduce the probability of exceeding the capacity of the TCAM on the PFC. See the "Configuring Wildcard Searches" section on page 38 for more details.

## Configuring a Virtual Server

To configure an IOS SLB virtual server, use the following commands beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config)# ip slb vserver` *virtual_server-name* | Identifies a virtual server and enters virtual server configuration mode. |
| **Step 2** | `Router(config-slb-vserver)# virtual` *ip-address* [*network-mask*] {**tcp** \| **udp**} [*port-number* \| **wsp** \| **wsp-wtp** \| **wsp-wtls** \| **wsp-wtp-wtls**] [**service** *service-name*] | Specifies the virtual server IP address, type of connection, TCP or UDP port number, WSP mode, and optional service coupling.<br><br>**Note** In GPRS load balancing, you must specify a virtual GGSN IP address as the virtual server, and use the **udp** and **service gtp** keyword options. Port number **3386** is recommended, if the GGSNs and SGSNs are in compliance with the ETSI standard. |
| **Step 3** | `Router(config-slb-vserver)# serverfarm` *primary-serverfarm-name* [**backup** *backup-serverfarm-name* [**sticky**]] | Associates a real server farm with a virtual server, or configures a backup server farm.<br><br>**Note** GPRS load balancing does not support the **sticky** attribute. |
| **Step 4** | `Router(config-slb-vserver)# advertise` | (Optional) Controls the installation of a static route to the Null0 interface for a virtual server address. |
| **Step 5** | `Router(config-slb-vserver)# client` *ip-address* *network-mask* | (Optional) Specifies which clients are allowed to use the virtual server.<br><br>**Note** GPRS load balancing does not support this command. |
| **Step 6** | `Router(config-slb-vserver)# delay` *duration* | (Optional) Specifies the amount of time IOS SLB maintains TCP connection context after a connection has terminated. |

| | Command | Purpose |
|---|---|---|
| **Step 7** | Router(config-slb-vserver)# **idle** *duration* | (Optional) Specifies the minimum amount of time IOS SLB maintains connection context in the absence of packet activity. |
| | | ✎ |
| | | **Note** In GPRS load balancing, you must specify an idle timer greater than the longest possible longest possible interval between PDP context requests on the SGSN. Also, the default is 30 seconds in GPRS load balancing, but 3600 seconds (1 hour) in all other environments. |
| **Step 8** | Router(config-slb-vserver)# **replicate casa** *listening-ip remote-ip port-number* [*interval*] [**password** [**0**|**7**] *password timeout*] | (Optional) Configures a stateful backup of IOS SLB decision tables to a backup switch. |
| | | ✎ |
| | | **Note** GPRS load balancing does not support this command. |
| **Step 9** | Router(config-slb-vserver)# **sticky** *duration* [**group** *group-id*] [**netmask** *netmask*] | (Optional) Specifies that connections from the same client use the same real server, as long as the interval between client connections does not exceed the specified duration. |
| | | ✎ |
| | | **Note** GPRS load balancing does not support this command. |
| **Step 10** | Router(config-slb-vserver)# **synguard** *syn-count interval* | (Optional) Specifies the rate of TCP SYNs handled by a virtual server in order to prevent a SYN flood denial-of-service attack. |
| | | ✎ |
| | | **Note** GPRS load balancing does not support this command. |
| **Step 11** | Router(config-slb-vserver)# **inservice** | Enables the virtual server for use by IOS SLB. |
| **Step 12** | Router(config-slb-vserver)# **client** *ip-address network-mask* | Specifies which clients are allowed to use the virtual server. |

## Verifying the Virtual Server

The following **show ip slb vserver** command verifies the configuration of the virtual servers PUBLIC_HTTP and RESTRICTED_HTTP:

```
Router# show ip slb vserver

slb vserver       prot  virtual                    state        conns
-------------------------------------------------------------------
PUBLIC_HTTP       TCP   10.0.0.1:80                OPERATIONAL    0
RESTRICTED_HTTP   TCP   10.0.0.2:80                OPERATIONAL    0
Router#
```

## Verifying the Server Farm

The following **show ip slb reals** command displays the status of server farms PUBLIC and RESTRICTED, the associated real servers, and their status:

```
Router# show ip slb real

real                   farm name       weight  state        conns
-----------------------------------------------------------------------
10.1.1.1               PUBLIC          8       OPERATIONAL    0
10.1.1.2               PUBLIC          8       OPERATIONAL    0
10.1.1.3               PUBLIC          8       OPERATIONAL    0
10.1.1.20              RESTRICTED      8       OPERATIONAL    0
10.1.1.21              RESTRICTED      8       OPERATIONAL    0
Router#
```

The following **show ip slb serverfarm** command displays the configuration and status of server farms PUBLIC and RESTRICTED:

```
Router# show ip slb serverfarm

server farm      predictor     nat    reals  bind id
---------------------------------------------------
PUBLIC           ROUNDROBIN    none   3      0
RESTRICTED       ROUNDROBIN    none   2      0
Router#
```

## Verifying the Clients

The following **show ip slb conns** command verifies the restricted client access and status:

```
Router# show ip slb conns

vserver         prot client            real              state     nat
---------------------------------------------------------------------------
RESTRICTED_HTTP TCP  10.4.4.0:80        10.1.1.20         CLOSING   none
Router#
```

The following **show ip slb conns** command displays detailed information about the restricted client access status:

```
Router# show ip slb conns client 10.4.4.0 detail
VSTEST_UDP, client = 10.4.4.0:80
  state = CLOSING, real = 10.1.1.20, nat = none
  v_ip = 10.0.0.2:80, TCP, service = NONE
  client_syns = 0, sticky = FALSE, flows attached = 0
Router#
```

## Verifying IOS SLB Connectivity

To verify that the IOS SLB feature has been installed and is operating correctly, ping the real servers from the IOS SLB switch, then ping the virtual servers from the clients.

The following **show ip slb stats** command displays detailed information about the IOS SLB network status:

```
Router# show ip slb stats
Pkts via normal switching:  0
Pkts via special switching: 6
Connections Created:        1
Connections Established:    1
Connections Destroyed:      0
Connections Reassigned:     0
Zombie Count:               0
Connections Reused:         0
```

Normal switching is when IOS SLB packets are handled on normal IOS switching paths (CEF, fastswitching, and process level switching). Special switching is when IOS SLB packets are handled on hardware-assisted switching paths.

See the "Monitoring and Maintaining the IOS SLB Feature" section on page 40 for additional commands used to verify IOS SLB networks and connections.

# Configuring Firewall Load Balancing

This section describes the tasks required to configure a basic IOS SLB firewall load-balancing network.

IOS SLB firewall load balancing uses probes to detect and recover from failures. You must configure a probe on each real server in the firewall farm. Ping probes are recommended; see the "Configuring Ping Probes" section on page 33 for more details. If a firewall does not allow ping probes to be forwarded, use HTTP probes instead. See the "Configuring HTTP Probes" section on page 32 for more details. You can configure more than one probe, in any combination of types (HTTP or ping), for each firewall in a firewall farm.

When performing server load balancing and firewall load balancing together on a Catalyst 6000 Family Switch, use the **mls ip slb wildcard search rp** command to reduce the probability of exceeding the capacity of the TCAM on the PFC. See the"Configuring Wildcard Searches" section on page 38 for more details.

This section describes the following IOS SLB firewall load-balancing configuration tasks. Required and optional tasks are indicated.

- Configuring the Firewall Farm, page 29 (Required)
- Verifying the Firewall Farm, page 30 (Optional)
- Verifying Firewall Connectivity, page 31 (Optional)

## Configuring the Firewall Farm

To configure an IOS SLB firewall load-balancing network, enter the following commands in order, beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| Step 1 | `Router(config)# ip slb firewallfarm`<br>`firewallfarm-name`<br>`Router(config-slb-fw)#` | Adds a firewall farm definition to the IOS SLB configuration and enters firewall farm configuration mode. |
| Step 2 | `Router(config-slb-fw)# real ip-address` | Identifies a firewall by IP address as a member of a firewall farm and enters real server configuration mode. |
| Step 3 | `Router(config-slb-fw-real)# probe name` | Associates a probe with the firewall. |
| Step 4 | `Router(config-slb-fw-real)# weight weighting-value` | (Optional) Specifies the firewall's workload capacity relative to other firewalls in the firewall farm. |
| Step 5 | `Router(config-slb-fw-real)# inservice` | Enables the firewall for use by the firewall farm and by IOS SLB. |
| Step 6 | `Router(config-slb-fw)# access [source`<br>`source-ip-address network-mask] [destination`<br>`destination-ip-address network-mask]` | (Optional) Routes specific flows to a firewall farm. |
| Step 7 | `Router(config-slb-fw)# predictor hash address [port]` | (Optional) Specifies whether the source and destination TCP or UDP port numbers, in addition to the source and destination IP addresses, are to be used when selecting a firewall. |
| Step 8 | `Router(config-slb-fw)# replicate casa listening-ip`<br>`remote-ip port-number [interval] [password [0|7]`<br>`password [timeout]]` | (Optional) Configures a stateful backup of IOS SLB firewall load balancing decision tables to a backup switch. |
| Step 9 | `Router(config-slb-fw)# tcp` | (Optional) Enters TCP protocol configuration mode. |
| Step 10 | `Router(config-slb-fw-tcp)# delay duration` | (Optional) For firewall farm TCP protocol configuration mode, specifies the amount of time IOS SLB firewall load balancing maintains TCP connection context after a connection has terminated. |
| Step 11 | `Router(config-slb-fw-tcp)# idle duration` | (Optional) For firewall farm TCP protocol configuration mode, specifies the minimum amount of time IOS SLB firewall load balancing maintains connection context in the absence of packet activity. |
| Step 12 | `Router(config-slb-fw-tcp)# maxconns number-conns` | (Optional) For firewall farm TCP protocol configuration mode, specifies the maximum number of active connections allowed on the firewall farm at one time. |

| | Command | Purpose |
|---|---|---|
| **Step 13** | `Router(config-slb-fw-tcp)# `**`sticky`**` duration [`**`netmask`**` netmask]` | (Optional) For firewall farm TCP protocol configuration mode, specifies that connections from the same IP address use the same firewall if either of the following conditions is met: <br><br> • As long as any connection from that IP address exists. <br><br> • For a period, defined by *duration*, after the last connection is destroyed. |
| **Step 14** | `Router(config-slb-fw)# `**`udp`** | (Optional) Enters UDP protocol configuration mode. |
| **Step 15** | `Router(config-slb-fw-udp)# `**`idle`**` duration` | (Optional) For firewall farm UDP protocol configuration mode, specifies the minimum amount of time IOS SLB firewall load balancing maintains connection context in the absence of packet activity. |
| **Step 16** | `Router(config-slb-fw-udp)# `**`maxconns`**` number-conns` | (Optional) For firewall farm UDP protocol configuration mode, specifies the maximum number of active connections allowed on the firewall farm at one time. |
| **Step 17** | `Router(config-slb-fw-udp)# `**`sticky`**` duration [`**`netmask`**` netmask]` | (Optional) For firewall farm UDP protocol configuration mode, specifies that connections from the same IP address use the same firewall if either of the following conditions is met: <br><br> • As long as any connection from that IP address exists. <br><br> • For a period, defined by *duration*, after the last connection is destroyed. |
| **Step 18** | `Router(config-slb-fw)# `**`inservice`** | Enables the firewall farm for use by IOS SLB. |

## Verifying the Firewall Farm

The following **show ip slb reals** command displays the status of firewall farm FIRE1, the associated real servers, and their status:

```
Router# show ip slb real

real               farm name       weight   state         conns
--------------------------------------------------------------------
10.1.1.2           FIRE1           8        OPERATIONAL   0
10.1.2.2           FIRE1           8        OPERATIONAL   0
```

The following **show ip slb firewallfarm** command displays the configuration and status of firewall farm FIRE1:

```
Router# show ip slb firewallfarm

firewall farm    hash        state        reals
---------------------------------------------------
FIRE1            IPADDR      INSERVICE    2
```

## Verifying Firewall Connectivity

To verify that IOS SLB firewall load balancing has been configured and is operating correctly:

**Step 1**   Ping the external real servers (the ones outside the firewall) from the IOS SLB firewall load-balancing switch.

**Step 2**   Ping the internal real servers (the ones inside the firewall) from the clients.

**Step 3**   Use the **show ip slb stats** command to display detailed information about the IOS SLB firewall load-balancing network status:

```
Router# show ip slb stats
Pkts via normal switching:  0
Pkts via special switching: 0
Connections Created:        1911871
Connections Established:    1967754
Connections Destroyed:      1313251
Connections Reassigned:     0
Zombie Count:               0
Connections Reused:         59752
Connection Flowcache Purges:1776582
Failed Connection Allocs:   17945
Failed Real Assignments:    0
```

Normal switching is when IOS SLB packets are handled on normal IOS switching paths (CEF, fastswitching, and process level switching). Special switching is when IOS SLB packets are handled on hardware-assisted switching paths.

**Step 4**   Use the **show ip slb real detail** command to display detailed information about the IOS SLB firewall load-balancing real server status:

```
Router# show ip slb real detail
10.1.1.3, FIRE1, state = OPERATIONAL, type = firewall
  conns = 299310, dummy_conns = 0, maxconns = 4294967295
  weight = 10, weight(admin) = 10, metric = 104, remainder = 2
  total conns established = 1074779, hash count = 4646
  server failures = 0
  interface FastEthernet1/0, MAC 0010.f68f.7020
```

**Step 5**   Use the **show ip slb conns** command to display detailed information about the active IOS SLB firewall load-balancing connections:

```
Router# show ip slb conns

vserver         prot client              real            state   nat
-------------------------------------------------------------------------------
FirewallTCP     TCP  80.80.50.187:40000   10.1.1.4        ESTAB   none
FirewallTCP     TCP  80.80.50.187:40000   10.1.1.4        ESTAB   none
FirewallTCP     TCP  80.80.50.187:40000   10.1.1.4        ESTAB   none
FirewallTCP     TCP  80.80.50.187:40000   10.1.1.4        ESTAB   none
FirewallTCP     TCP  80.80.50.187:40000   10.1.1.4        ESTAB   none
```

**Step 6**   See the for additional commands used to verify IOS SLB networks and connections.

# Configuring Probes

IOS SLB uses probes to verify connectivity and detect failures. For a detailed description of each type of probe, see the "Probes" section on page 12.

The following sections describe how to configure and verify probes. Required and optional tasks are indicated.

- Configuring HTTP Probes, page 32 (Required)
- Configuring Ping Probes, page 33 (Required)
- Configuring WSP Probes, page 33 (Required)
- Associating the Probe, page 33 (Required)
- Verifying the Probe, page 34 (Optional)

## Configuring HTTP Probes

To configure an HTTP probe, enter the following commands in order, beginning in global configuration mode:

|  | Command | Description |
|---|---|---|
| Step 1 | Router(config)# **ip slb probe** *name* **http** | Configures the IOS SLB probe name and changes to HTTP probe configuration submode. |
| Step 2 | Router(config-slb-probe)# **address** [*ip-address*] | (Optional) Configures an IP address to which to send the HTTP probe. |
| Step 3 | Router(config-slb-probe)# **credentials** {*username* [*password*]} | (Optional) Configures header values for the HTTP probe. |
| Step 4 | Router(config-slb-probe)# **expect** [**status** *number*] [**regex** *regular-expression*] | (Optional) Configures the expected HTTP status code or regular expression. |
| Step 5 | Router(config-slb-probe)# **header** {**name** *field-name* [*field-value*]} | (Optional) Configures header values for the HTTP probe. |
| Step 6 | Router(config-slb-probe)# **interval** *seconds* | (Optional) Configures the HTTP probe transmit timers. |
| Step 7 | Router(config-slb-probe)# **port** *port-number* | (Optional) Configures the port to which the HTTP probe is to connect. |
| Step 8 | Router(config-slb-probe)# **request** [**method** {**get** \| **post** \| **head** \| **name** *name*}] [**url** *path*] | (Optional) Configures the URL path to request from the server, and the method used to perform the request to the server. |

In addition, HTTP probes require a route to the virtual server. The route is not used, but it must exist to enable the sockets code to verify that the destination can be reached, which in turn is essential for HTTP probes to function correctly. The route can be either a host route (advertised by the virtual server) or a default route (specified using the **ip route 0.0.0.0 0.0.0.0** command, for example).

## Configuring Ping Probes

To configure a ping probe, enter the following commands in order, beginning in global configuration mode:

|        | Command | Description |
|--------|---------|-------------|
| Step 1 | `Router(config)# ip slb probe name ping` | Configures the IOS SLB probe name and changes to ping probe configuration submode. |
| Step 2 | `Router(config-slb-probe)# address [ip-address]` | (Optional) Configures an IP address to which to send the ping probe. |
| Step 3 | `Router(config-slb-probe)# faildetect number-of-pings` | (Optional) Specifies the number of consecutive unacknowledged pings that constitute failure of the real server or firewall. |
| Step 4 | `Router(config-slb-probe)# interval seconds` | (Optional) Configures the ping probe transmit timers. |

## Configuring WSP Probes

To configure a WSP probe, enter the following commands in order, beginning in global configuration mode:

|        | Command | Description |
|--------|---------|-------------|
| Step 1 | `Router(config)# ip slb probe name wsp` | Configures the IOS SLB probe name and changes to WSP probe configuration submode. |
| Step 2 | `Router(config-slb-probe)# address [ip-address]` | (Optional) Configures an IP address to which to send the WSP probe. |
| Step 3 | `Router(config-slb-probe)# interval seconds` | (Optional) Configures the WSP probe transmit timers. |
| Step 4 | `Router(config-slb-probe)# url [path]` | (Optional) Configures the WSP probe URL path. |

## Associating the Probe

After configuring a probe, you must associate it with a real server or firewall, using the **probe** command. See the "Configuring a Server Farm and Real Server" section on page 22 and the "Configuring Firewall Load Balancing" section on page 28 for more details.

**Note** You cannot associate a WSP probe with a firewall.

## Verifying the Probe

To verify that a probe is configured correctly, use the **show ip slb probe** command:

```
Router# show ip slb probe

Server:Port          State          Outages  Current  Cumulative
----------------------------------------------------------------
10.1.1.1:80          OPERATIONAL          0  never    00:00:00
10.1.1.2:80          OPERATIONAL          0  never    00:00:00
10.1.1.3:80          OPERATIONAL          0  never    00:00:00
Router#
```

# Configuring DFP

You can define IOS SLB as a DFP manager, as a DFP agent for another DFP manager (such as DistributedDirector), or as both at the same time. Depending on your network configuration, you might enter the commands for configuring IOS SLB as a DFP manager and the commands for configuring IOS SLB as a DFP agent on the same device or on different devices.

To configure IOS SLB as a DFP manager, and to identify a DFP agent with which IOS SLB can initiate connections, enter the following commands in order, beginning in global configuration mode:

|        | Command | Description |
|--------|---------|-------------|
| Step 1 | Router(config)# **ip slb dfp** [**password** [**0**\|**7**] *password* [*timeout*]] | Configures DFP, supplies an optional password, and enters DFP configuration mode. |
| Step 2 | Router(config-slb-dfp)# **agent** *ip_address port-number* [*timeout* [*retry_count* [*retry_interval*]]] | Identifies a DFP agent to which IOS SLB can connect. |

To configure IOS SLB as a DFP agent, see the *DFP Agent Subsystem* feature module.

# GPRS Load Balancing Configuration Task List

This section lists the tasks used to configure GPRS load balancing. Detailed configuration information is contained in the referenced sections of this or other documents. Required and optional tasks are indicated.

- Configuring a Server Farm and Real Server, page 22 (Required)

When you configure the server farm and real server for GPRS load balancing, keep the following considerations in mind:

- – Accept the default setting (the weighted round robin algorithm) for the **predictor** command.
- – Specify the IP addresses (virtual template addresses) of the real servers performing the GGSN function, using the **real** command.
- – Accept the default value (4294967295) for the **maxconns** command.
- – Specify a reassign threshold less than the value specified on the **gprs gtp n3-requests** command (or the equivalent command, if you are not using a Cisco SGSN), using the **reassign** command.

- Configuring a Virtual Server, page 25 (Required)

    When you configure the virtual server for GPRS load balancing, keep the following considerations in mind:

    – Specify a virtual GGSN IP address as the virtual server, and use the **udp** and **service gtp** keyword options, using the **virtual** command. Port number **3386** is recommended, if the GGSNs and SGSNs are in compliance with the ETSI standard.

    – Specify an idle timer greater than the longest possible longest possible interval between PDP context requests on the SGSN, using the **idle** command.

- Configuring the virtual IP address as a loopback on each of the GGSNs in the server (Required)

    See the "Configuring a Loopback Interface" section in the *Cisco IOS Interface Configuration Guide* for more information.

- Configuring DFP, page 34 (Optional)

    – Configuring the Maximum DFP Weight for a GGSN, page 35 (Optional)

    – Configuring the Maximum Number of PDP Contexts for a GGSN, page 35 (Optional)

- Identifying the GGSN Virtual Server to CEF, page 36 (Optional)

- Routing each GGSN to each associated SGSN (Required)

    The route can be static or dynamic, but the GGSN needs to be able to reach the SGSN. See the "Configuring Network Access to the GGSN" section of the *Cisco IOS Mobile Wireless Configuration Guide* for more details.

- Routing each SGSN to the virtual templates on each associated GGSN, and to the GPRS load-balancing virtual server (Required)

    See the configuration guide for your SGSN for more details.

## Configuring the Maximum DFP Weight for a GGSN

If you use DFP with GPRS load balancing, each GGSN that acts as a DFP agent has a maximum weight that it can send to a DFP manager. For each GGSN, you can accept the default maximum weight, or you can specify a different maximum weight.

To specify the maximum weight for a GGSN, enter the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **gprs dfp max-weight** [*max-weight-value*] | Specifies the maximum weight of a GGSN that is acting as a DFP agent. |

## Configuring the Maximum Number of PDP Contexts for a GGSN

If you use DFP with GPRS load balancing, you must specify a maximum number of PDP contexts for each GGSN, using the **gprs maximum-pdp-context-allowed** command. *Do not* accept the default value of 1000 PDP contexts. A value of **45000** is recommended. Significantly lower values, including the default value of 1000, can impact performance in a GPRS load-balancing environment.

To configure a maximum number of PDP contexts for a GGSN, enter the following command in global configuration mode:

| Command | Purpose |
|---------|---------|
| Router(config)# **gprs maximum-pdp-context-allowed** [*pdp_contexts*] | Specifies the maximum number of PDP contexts (mobile sessions) that can be activated on the GGSN. |

See the "Customizing the GPRS Configuration" section in the *Cisco IOS Mobile Wireless Configuration Guide* for more information.

## Identifying the GGSN Virtual Server to CEF

If you have enabled CEF on a GGSN, you must identify the IP address of the GGSN virtual server to CEF. (This IP address is also a loopback address on the GGSN.) If you have *not* enabled CEF on the GGSN, do not perform this task.

To identify the IP address of the GGSN virtual server to CEF, enter the following command in global configuration mode:

| Command | Purpose |
|---------|---------|
| Router(config)# **gprs slb cef** [*virtual-server-address*] | Specifies the IP address of the GGSN virtual server instance used by clients to connect to the server farm, for use by CEF. This command is required only if the GGSN is using CEF switching. Do not use this command if the GGSN is not using CEF switching. |

# Configuring NAT

To configure the IOS SLB NAT client address pool, enter the following command in global configuration mode:

| Command | Purpose |
|---------|---------|
| Router(config)# **ip slb natpool** *pool-name start-ip end-ip* [**netmask** *netmask* \| **prefix-length** *leading_1_bits*] [**entries** *init-addr* [*max-addr*]] | Configures the client address pool.<br><br>✎<br>**Note**   GPRS load balancing does not support this command. |

You must also specify either NAT client translation mode or NAT server address translation mode on the server farm, using the **nat** command. See the "Configuring a Server Farm and Real Server" section on page 22 for more details.

# Stateless Backup Configuration Task List

This section lists the tasks used to configure stateless backup over VLANs between IOS SLB devices. Detailed configuration information is contained in the referenced sections of this or other documents. Required and optional tasks are indicated.

- Configuring Required and Optional IOS SLB Functions, page 22 (Required for server load balancing)

- Configuring Firewall Load Balancing, page 28 (Required for firewall load balancing)

- Configuring the IP Routing Protocol (Required)

  See the "IP Routing Protocols" chapter of the *Cisco IOS IP and IP Routing Configuration Guide* for more details.

- Configuring the VLAN between the IOS SLB devices (Required)

  See the "Virtual LANs" chapter of the *Cisco IOS Switching Services Configuration Guide* for more details.

- Enabling HSRP on IOS SLB (Required)

  See the "Configuring SLB Redundancy" chapter of the *Catalyst 4840G Software Feature and Configuration Guide* for more details.

- Verifying the Stateless Backup Configuration, page 37 (Optional)

✎

**Note** For active standby, in which multiple IOS SLB devices share a virtual IP address, you must use exclusive client ranges and you must use policy routing to forward flows to the correct IOS SLB device.

## Verifying the Stateless Backup Configuration

For server load balancing, to verify that stateless backup has been configured and is operating correctly, use the following **show ip slb vserver** commands to display information about the IOS SLB virtual server status:

```
Router# show ip slb vserver

slb vserver      prot  virtual                  state        conns
-------------------------------------------------------------------
VS1              TCP   10.10.10.12:23           OPERATIONAL  2
VS2              TCP   10.10.10.18:23           OPERATIONAL  2

Router# show ip slb vserver detail
VS1, state = OPERATIONAL, v_index = 10
  virtual = 10.10.10.12:23, TCP, service = NONE, advertise = TRUE
  server farm = SERVERGROUP1, delay = 10, idle = 3600
  sticky timer = 0, sticky subnet = 255.255.255.255
  sticky group id = 0
  synguard counter = 0, synguard period = 0
  conns = 0, total conns = 0, syns = 0, syn drops = 0
  standby group = None
VS2, state = INSERVICE, v_index = 11
  virtual = 10.10.10.18:23, TCP, service = NONE, advertise = TRUE
  server farm = SERVERGROUP2, delay = 10, idle = 3600
  sticky timer = 0, sticky subnet = 255.255.255.255
  sticky group id = 0
  synguard counter = 0, synguard period = 0
  conns = 0, total conns = 0, syns = 0, syn drops = 0
  standby group = None
```

For firewall load balancing, to verify that stateless backup has been configured and is operating correctly, use the following **show ip slb firewallfarm** commands to display information about the IOS SLB firewall farm status:

```
Router# show ip slb firewallfarm

firewall farm    hash        state         reals
-------------------------------------------------
FIRE1            IPADDR      INSERVICE     2

Router# show ip slb firewallfarm details
FIRE1, hash = IPADDRPORT, state = INSERVICE, reals = 2
  FirewallTCP:
   sticky timer = 0, sticky subnet = 255.255.255.255
   idle = 3600, delay = 10, syns = 1965732, syn drop = 0
   maxconns = 4294967295, conns = 597445, total conns = 1909512
  FirewallUDP:
   sticky timer = 0, sticky subnet = 255.255.255.255
   idle = 3600
   maxconns = 1, conns = 0, total conns = 1
  Real firewalls:
    10.1.1.3, weight = 10, OPERATIONAL, conns = 298823
    10.1.1.4, weight = 10, OPERATIONAL, conns = 298622
  Total connections = 597445
```

# Configuring Database Entries

To configure an initial allocation and a maximum value for IOS SLB database entries, enter the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **ip slb entries** [**conn** [*init-conn* [*max-conn*]] \| **frag** [*init-frag* [*max-frag*]] \| **sticky** [*init-sticky* [*max-sticky*]]] | Specifies an initial allocation and a maximum value for IOS SLB database entries. |

# Configuring Wildcard Searches

To specify the behavior of IOS SLB wildcard searches, enter the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **mls ip slb search wildcard** [**pfc** \| **rp**] | Specifies whether IOS SLB wildcard searches are to be performed by the PFC or by the route processor. |
| | When performing server load balancing and firewall load balancing together on a Catalyst 6000 Family Switch, specify the **rp** keyword to reduce the probability of exceeding the capacity of the TCAM on the PFC. |
| | This command is supported for Catalyst 6000 Family Switches only. |

# Clearing Connections and Counters

To clear IP IOS SLB connections or counters, enter the following command in privileged EXEC mode:

| Command | Purpose |
|---|---|
| Router# **clear ip slb** {**connections** [**serverfarm** *farm_name* \| **vserver** *server_name*] \| **counters**} | Clears the IOS SLB connection database for one or more server farms or virtual servers, or clears the IOS SLB counters. |

# Purging Connections

IOS SLB enables you to reset, or purge, client TCP connections to failed serverfarms and firewalls without waiting for the TCP retransmission timeout.

To purge connections when a real server fails, enter the following commands in order, beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| Step 1 | Router(config)# **ip slb serverfarm** *serverfarm-name* | Enters server farm configuration mode. |
| Step 2 | Router(config-slb-sfarm)# **failaction purge** | Resets client TCP connections to failed real servers in the server farm without waiting for the TCP retransmission timeout. |

To purge connections when a firewall fails, enter the following commands in order, beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| Step 1 | Router(config)# **ip slb firewallfarm** *firewallfarm-name* | Enters firewall farm configuration mode. |
| Step 2 | Router(config-slb-fw)# **failaction purge** | Resets client TCP connections to failed firewalls in the firewall farm without waiting for the TCP retransmission timeout. |

# Monitoring and Maintaining the IOS SLB Feature

To obtain and display runtime information about IOS SLB, use the following commands in EXEC mode:

| Command | Purpose |
| --- | --- |
| Router# **show ip slb conns** [**vserver** *virtual_server-name* \| **client** *ip-address* \| **firewall** *firewallfarm-name*] [**detail**] | Displays all connections handled by IOS SLB, or, optionally, only those connections associated with a particular virtual server or client. |
| Router# **show ip slb dfp** [**agent** *agent_ip_address port-number* \| **manager** *manager_ip_address* \| **detail** \| **weights**] | Displays information about DFP and DFP agents, and about the weights assigned to real servers. |
| Router# **show ip slb firewallfarm** [**detail**] | Displays information about firewall farms. |
| Router# **show ip slb natpool** [**name** *pool_name*] [**detail**] | Displays information about the IOS SLB NAT configuration. |
| Router# **show ip slb probe** [**name** *probe_name*] [**detail**] | Displays information about HTTP and ping probes defined to IOS SLB. |
| Router# **show ip slb reals** [**vserver** *virtual_server-name*] [**detail**] | Displays information about the real servers defined to IOS SLB. |
| Router# **show ip slb replicate** | Displays information about the IOS SLB replication configuration. |
| Router# **show ip slb serverfarms** [**name** *serverfarm-name*] [**detail**] | Displays information about the server farms defined to IOS SLB. |
| Router# **show ip slb stats** | Displays IOS SLB statistics. |
| Router# **show ip slb sticky** [**client** *ip-address*] | Displays information about the sticky connections defined to IOS SLB. |
| Router# **show ip slb vserver** [**name** *virtual_server-name*] [**detail**] | Displays information about the virtual servers defined to IOS SLB. |

# Configuration Examples

This section provides real-world examples of IOS SLB configurations. For a complete description of the IOS SLB commands in this section, see the "Command Reference" section on page 91. To locate documentation of other commands that appear in this section, search online using Cisco.com.

This section includes the following examples:

> **Note** The IP and network addresses in these examples are generic; you must replace them with the actual addresses for your network.

## Basic IOS SLB Network Configuration Example

Figure 2 shows a sample IOS SLB network with the following components:

- Two server farms—one configured to allow access by the public and named PUBLIC, one configured to allow limited access and named RESTRICTED.
- Five real servers configured as follows:
  - Three real servers in the PUBLIC server farm with IP addresses 10.1.1.1, 10.1.1.2, and 10.1.1.3
  - Two real servers in the restricted server farm with IP addresses 10.1.1.20 and 10.1.1.21

- Two virtual servers—one configured to allow access by the public and named PUBLIC_HTTP and one configured to allow limited access and named RESTRICTED_HTTP.

  – Virtual server PUBLIC_HTTP is configured with IP address 10.0.0.1 load balancing TCP connections on the WWW port (80).

  – Virtual server RESTRICTED_HTTP is configured with IP address 10.0.0.2 load balancing TCP connections on the WWW port (80) and allows access only from clients from network 10.4.4.0 255.255.255.0.

*Figure 2      Example IOS SLB Network*
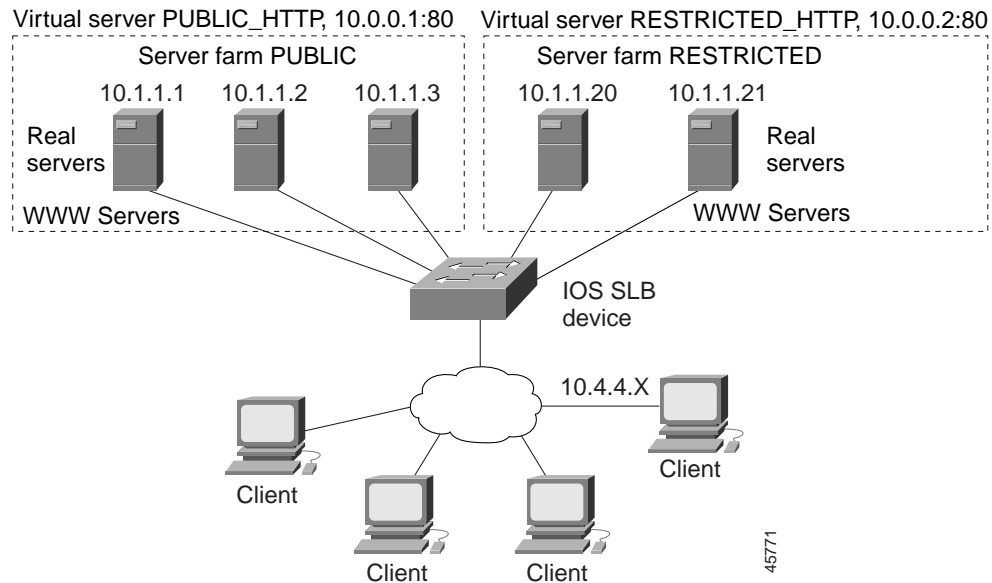


The following sections include examples of the configuration commands used to configure and verify the IOS SLB network shown in Figure 2:

- Server Farm Configuration, page 42
- Virtual Server Configuration, page 43
- Restricted Client Configuration, page 43

## Server Farm Configuration

The following example shows the configuration for the server farm PUBLIC, associated with three real servers:

```
ip slb serverfarm PUBLIC
  real 10.1.1.1
    reassign 2
    faildetect numconns 4 numclients 2
    retry 20
    inservice
    exit
  real 10.1.1.2
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
    exit
```

```
real 10.1.1.3
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
  end
```

The following example shows the configuration for the server farm RESTRICTED, associated with two real servers:

```
ip slb serverfarm RESTRICTED
  real 10.1.1.20
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
    exit
  real 10.1.1.21
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
    end
```

## Virtual Server Configuration

The following example shows the configuration for the virtual servers PUBLIC_HTTP and RESTRICTED_HTTP:

```
ip slb vserver PUBLIC_HTTP
  virtual 10.0.0.1 tcp www
  serverfarm PUBLIC
  idle 120
  delay 5
  inservice
  exit
ip slb vserver RESTRICTED_HTTP
  virtual 10.0.0.2 tcp www
  serverfarm RESTRICTED
  idle 120
  delay 5
  inservice
  end
```

## Restricted Client Configuration

The following example shows the configuration for the virtual server RESTRICTED_HTTP:

```
ip slb vserver RESTRICTED_HTTP
  no inservice
  client 10.4.4.0 255.255.255.0
  inservice
  end
```

# Complete IOS SLB Configuration Example

The following example provides a complete configuration using most of the commands described in this feature module:

```
ip slb probe PROBE2 http
 request method POST url /probe.cgi?all
 header Cookie Monster
!
ip slb serverfarm PUBLIC
 nat server
 real 10.1.1.1
  reassign 4
  faildetect numconns 16
  retry 120
  inservice
 real 10.1.1.2
  reassign 4
  faildetect numconns 16
  retry 120
  inservice
 probe PROBE2
!
ip slb serverfarm RESTRICTED
 predictor leastconns
 bindid 309
 real 10.1.1.1
  weight 32
  maxconns 1000
  reassign 4
  faildetect numconns 16
  retry 120
  inservice
 real 10.1.1.20
  reassign 4
  faildetect numconns 16
  retry 120
  inservice
 real 10.1.1.21
  reassign 4
  faildetect numconns 16
  retry 120
  inservice
!
ip slb vserver PUBLIC_HTTP
 virtual 10.0.0.1 tcp www
 serverfarm PUBLIC
 no inservice
!
ip slb vserver RESTRICTED_HTTP
 virtual 10.0.0.2 tcp www
 serverfarm RESTRICTED
 no advertise
 sticky 60 group 1
 idle 120
 delay 5
 client 10.4.4.0 255.255.255.0
 synguard 3600000
 inservice
```

# Layer 3 Switch with ISL, VLAN, and BVI with GEC Example

This example configuration focuses on both the Inter-Switch Link (ISL) and virtual LANs (VLANs), and on integrated routing and bridging (IRB) using a bridge-group virtual interface (BVI) over Gigabit EtherChannel (GEC). The Cisco proprietary ISL allows any Fast Ethernet port to be configured as a trunk. The Spanning-Tree Protocol detects and breaks loops on all the VLANs carried across the trunk.

The Gigabit Ethernet interface information applies to both two-port and eight-port Gigabit Ethernet interfaces for a Catalyst 8540 campus Layer 3 switch. This example also includes port snooping and Network Time Protocol (NTP) configurations.

```
ip subnet-zero
no ip domain-lookup
ip name-server 171.69.2.132
ip name-server 198.92.30.32
ip multicast-routing
ip dvmrp route-limit 20000
bridge irb
!
interface FastEthernet1
    no ip address
    no ip directed-broadcast
    no keepalive
!
interface FastEthernet1.128
    ip address 172.68.16.10 255.255.255.0
    ip helper-address 172.68.16.15
    no ip redirects
    no ip directed-broadcast
    ip pim dense-mode
    ip multicast ttl-threshold 1
    encapsulation isl 128
!
interface FastEthernet1.199
    ip address 172.68.17.15 255.255.255.0
    ip helper-address 172.68.16.16
    ip helper-address 172.68.16.17
    ip helper-address 172.68.16.18
    no ip redirects
    no ip directed-broadcast
    ip pim dense-mode
    ip multicast ttl-threshold 1
    encapsulation isl 199
!
interface FastEthernet1.201
    ip address 172.68.18.10 255.255.255.0
    ip helper-address 172.68.16.16
    ip helper-address 172.68.16.17
    ip helper-address 172.68.16.18
    no ip redirects
    no ip directed-broadcast
    ip pim dense-mode
    ip multicast ttl-threshold 1
    encapsulation isl 201
!
interface FastEthernet2
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
```

```
interface FastEthernet3
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet4
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet5
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet6
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet7
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet8
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet9
    ip address 172.68.19.10 255.255.255.0
    ip helper-address 172.68.16.16
    ip helper-address 172.68.16.17
    ip helper-address 172.68.16.18
    no ip redirects
    no ip directed-broadcast
    ip pim dense-mode
    ip multicast ttl-threshold 1
    ip sdr listen
    no keepalive
!
interface FastEthernet10
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
interface FastEthernet11
    no ip address
    no ip directed-broadcast
    no keepalive
    shutdown
!
```

```
interface GigabitEthernet41
 snoop interface FastEthernet3 direction both
 snoop interface FastEthernet5 direction both
 snoop interface FastEthernet6 direction both
    ip address 172.68.21.10 255.255.255.0
    ip helper-address 172.68.16.19
    ip helper-address 172.68.16.20
    ip helper-address 172.68.16.21
!
interface GigabitEthernet42
    ip address 172.68.1.1 255.255.255.0
    no ip directed-broadcast
    ip pim sparse-dense-mode
!
interface BVI1
    ip address 171.201.1.2 255.255.255.0
    no ip directed-broadcast
    ip pim dense-mode
    no ip route-cache cef
!
interface Ethernet0
    ip address 172.68.20.10 255.255.255.0
    no ip directed-broadcast
!
router eigrp 170
 network 171.200.0.0
 network 171.201.0.0
 network 172.68.0.0
 network 172.69.0.0
 no auto-summary
!
router bgp 180
 network 172.68.1.0
 network 172.69.1.0
 no auto-summary
!
ip classless
!
bridge 1 protocol ieee
bridge 1 route ip
!
ip http server
!
line con 0
line aux 0
line vty 0 4
    login
!
ntp clock-period 17181168
ntp update-calendar
ntp server 171.71.150.52
ntp server 171.69.4.143
ntp server 171.69.5.10
end
```

# IOS SLB with Firewall Load Balancing Example

Figure 3 shows a sample IOS SLB firewall load-balancing network with the following components:

- Two firewalls with IP addresses as shown
- An internal firewall load-balancing device on the secure side of the firewalls
- An external firewall load-balancing device on the Internet side of the firewalls
- One firewall farm named FIRE1, containing both firewalls

*Figure 3      IOS SLB with Layer 3 Firewalls in Different Subnets*
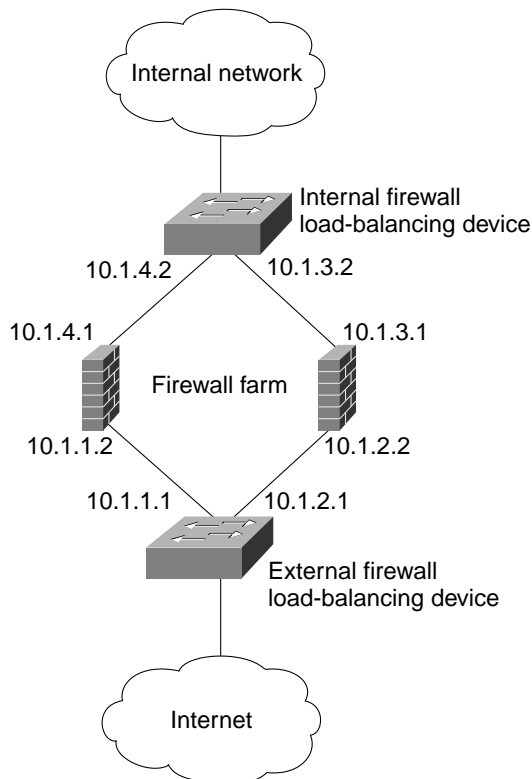


When you configure IOS SLB firewall load balancing, the load-balancing devices use route lookup to recognize flows destined for the firewalls. To enable route lookup, you must configure each device with the IP address of each firewall that will route flows to that device.

In the following firewall farm configuration samples:

- The internal (secure side) firewall load-balancing device is configured with firewall IP addresses 10.1.3.1 and 10.1.4.1.
- The external (Internet side) firewall load-balancing device is configured with firewall IP addresses 10.1.1.2 and 10.1.2.2.

## Internal Firewall Load-Balancing Device

The following example shows the configuration for ping probe PROBE1, HTTP probe PROBE2, and firewall farm FIRE1, associated with the two real servers for the load-balancing device on the internal (secure) side of the firewall:

```
!-----Ping probe
ip slb probe PROBE1 ping
!-----IP address of other load-balancing device
  address 10.1.1.1
  faildetect 4
!-----HTTP probe
  ip slb probe PROBE2 http
!-----IP address of other load-balancing device
  address 10.1.2.1
  expect status 401
!-----Firewall farm FIRE1
ip slb firewallfarm FIRE1
!-----First firewall
  real 10.1.4.1
    probe PROBE1
!-----Enable first firewall
    inservice
!-----Second firewall
    real 10.1.3.1
    probe PROBE2
!-----Enable second firewall
    inservice
    exit
  inservice
```

## External Firewall Load-Balancing Device

The following example shows the configuration for ping probe PROBE1, HTTP probe PROBE2, and firewall farm FIRE1, associated with the two real servers for the load-balancing device on the external (Internet) side of the firewall:

```
!-----Ping probe
ip slb probe PROBE1 ping
!-----IP address of other load-balancing device
  address 10.1.4.2
  faildetect 4
!-----HTTP probe
ip slb probe PROBE2 http
!-----IP address of other load-balancing device
  address 10.1.3.2
  expect status 401
!-----Firewall farm FIRE1
ip slb firewallfarm FIRE1
!-----First firewall
  real 10.1.1.2
    probe PROBE1
!-----Enable first firewall
    inservice
!-----Second firewall
  real 10.1.2.2
    probe PROBE2
!-----Enable second firewall
    inservice
    exit
  inservice
```

# IOS SLB with Server Load Balancing and Firewall Load Balancing Example

Figure 4 shows a sample IOS SLB load-balancing network with server load balancing and firewall load balancing running together, and the following components:

- Two real servers with IP addresses as shown

- One server farm named PUBLIC, containing both real servers

- Two firewalls with IP addresses as shown

- One firewall farm named FIRE1, containing both firewalls

- An internal IOS SLB device on the secure side of the firewalls, performing server load balancing and firewall load balancing

- An external firewall load-balancing device on the Internet side of the firewalls

*Figure 4      IOS SLB with Server Load Balancing and Firewall Load Balancing*
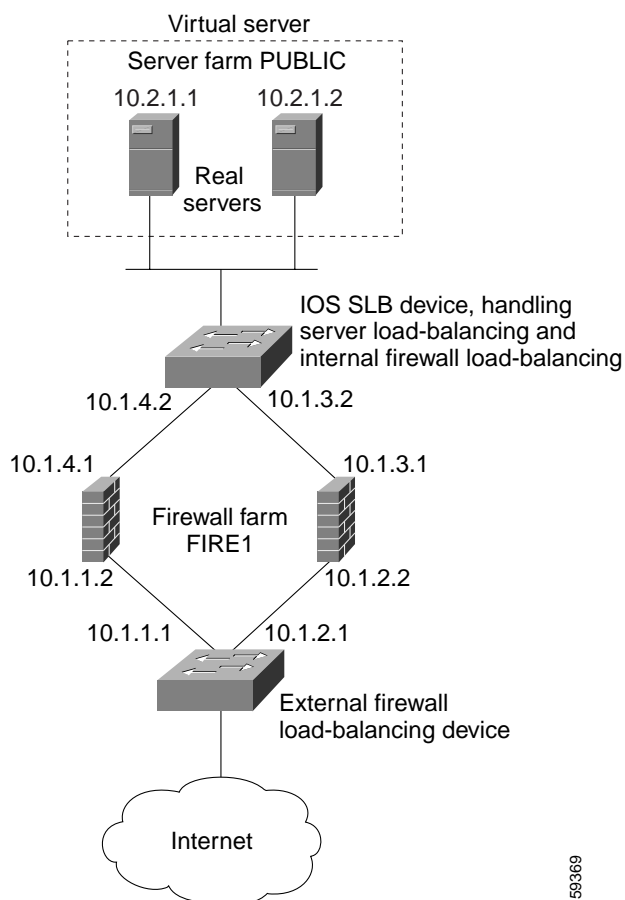


In the following firewall farm configuration samples:

- The internal (secure side) firewall load-balancing device is configured with firewall IP addresses 10.1.3.1 and 10.1.4.1.

- The external (Internet side) firewall load-balancing device is configured with firewall IP addresses 10.1.1.2 and 10.1.2.2.

## Internal Server and Firewall Load-Balancing Device

The following example shows the configuration for ping probes ABCPROBE and XYZPROBE, firewall farm FIRE1, and server farm PUBLIC for the load-balancing device on the internal (secure) side of the firewalls:

```
ip slb probe ABCPROBE ping
  address 10.1.1.1
ip slb probe XYZPROBE ping
  address 10.1.2.1
!
ip slb firewallfarm FIRE1
  real 10.1.4.1
    probe ABCPROBE
    inservice
  real 10.1.3.1
    probe XYZPROBE
    inservice
  inservice
!
ip slb serverfarm PUBLIC
  nat server
  real 10.2.1.1
    inservice
    real 10.2.1.2
    inservice
!
ip slb vserver HTTP1
  virtual 128.1.0.1 tcp www
  serverfarm PUBLIC
  idle 120
  delay 5
  inservice
```

## External Firewall Load-Balancing Device

The following example shows the configuration for ping probes ABCPROBE and XYZPROBE and firewall farm FIRE1 for the load-balancing device on the external (Internet) side of the firewalls:

```
ip slb probe ABCPROBE ping
  address 10.1.4.2
  ip slb probe XYZPROBE ping
  address 10.1.3.2
  ip slb firewallfarm FIRE1
  real 10.1.1.2
    probe ABCPROBE
    inservice
    probe XYZPROBE
    inservice
  inservice
```

# IOS SLB with Multiple Firewall Farms Example

Figure 5 shows a sample IOS SLB load-balancing network with multiple firewall farms and the following components:

- Four firewalls with IP addresses as shown
- An internal firewall load-balancing device on the secure side of the firewalls
- An external firewall load-balancing device on the Internet side of the firewalls
- One firewall farm named ABCFARM, containing the two firewalls on the left.
- One firewall farm named XYZFARM, containing the two firewalls on the right.

*Figure 5      IOS SLB with Multiple Firewall Farms*
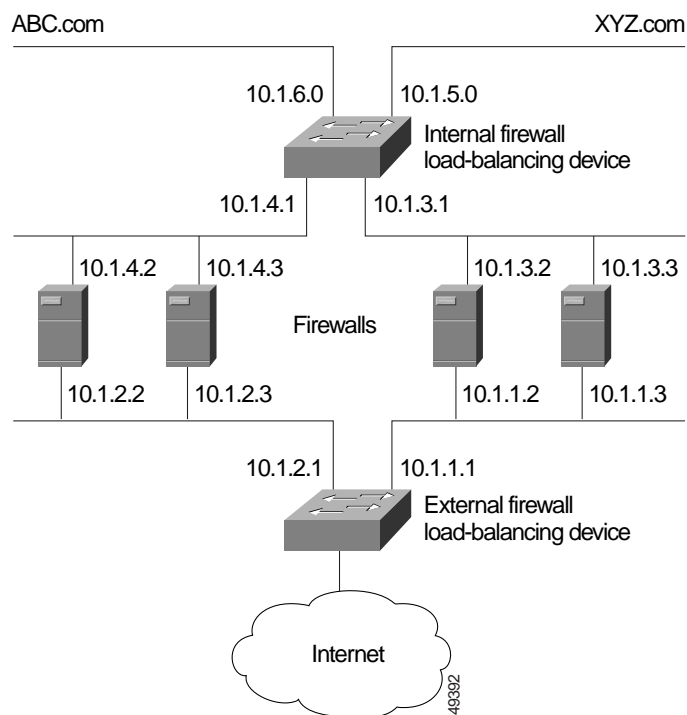


In the following firewall farm configuration samples:

- The internal (secure side) firewall load-balancing device is configured with firewall IP addresses 10.1.3.1 and 10.1.4.1.
- The external (Internet side) firewall load-balancing device is configured with firewall IP addresses 10.1.1.2 and 10.1.2.2.

## Internal Firewall Load-Balancing Device

The following example shows the configuration for ping probes ABCPROBE and XYZPROBE and firewall farms ABCFARM and XYZFARM for the load-balancing device on the internal (secure) side of the firewalls:

```
ip slb probe ABCPROBE ping
  address 10.1.2.1
  ip slb probe XYZPROBE ping
  address 10.1.1.1
  ip slb firewallfarm ABCFARM
  access source 10.1.6.0 255.255.255.0
  inservice
  real 10.1.4.2
    probe ABCPROBE
    inservice
    real 10.1.4.3
    probe ABCPROBE
    inservice
ip slb firewallfarm XYZFARM
  access source 10.1.5.0 255.255.255.0
  inservice
  real 10.1.3.2
    probe XYZPROBE
    inservice
    real 10.1.3.3
    probe XYZPROBE
    inservice
    exit
  inservice
```

## External Firewall Load-Balancing Device

The following example shows the configuration for ping probes ABCPROBE and XYZPROBE and firewall farms ABCFARM and XYZFARM for the load-balancing device on the external (Internet) side of the firewalls:

```
ip slb probe ABCPROBE ping
  address 10.1.4.1
  ip slb probe XYZPROBE ping
  address 10.1.3.1
  ip slb firewallfarm ABCFARM
  access destination 10.1.6.0 255.255.255.0
  inservice
  real 10.1.2.2
    probe ABCPROBE
    inservice
    real 10.1.2.3
    probe ABCPROBE
    inservice
ip slb firewallfarm XYZFARM
  access destination 10.1.5.0 255.255.255.0
  inservice
  real 10.1.1.2
    probe XYZPROBE
    inservice
    real 10.1.1.3
    probe XYZPROBE
    inservice
    exit
  inservice
```

# IOS SLB with Probes Example

Figure 6 shows an example configuration with IOS SLB real server connections configured as part of a server farm, focusing on using ping and HTTP probes to monitor applications being server load-balanced.

*Figure 6      Sample Ping and HTTP Probe Topology*

:
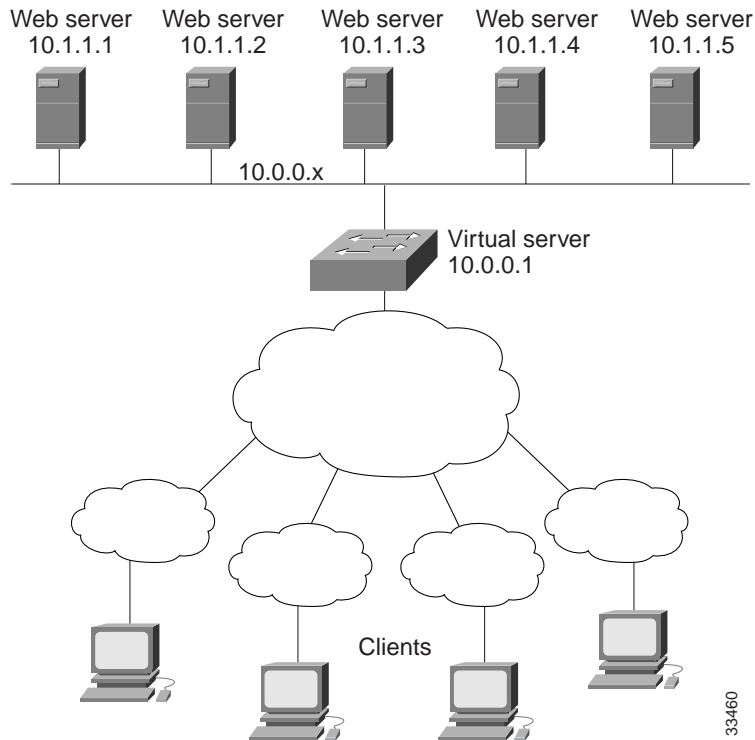


The topology shown in Figure 6 is a heterogeneous server farm servicing a single virtual server. Following are the configuration statements for this topology, including a ping probe named *PROBE1* and an HTTP probe named *PROBE2*:

```
! Configure ping probe PROBE1, change CLI to IOS SLB probe configuration mode
ip slb probe PROBE1 ping
! Configure probe to receive responses from IP address 13.13.13.13
  address 13.13.13.13
! Configure unacknowledged ping threshold to 16
  faildetect 16
! Configure ping probe timer interval to transmit every 11 seconds
  interval 11
! Configure HTTP probe PROBE2
  ip slb probe PROBE2 http
! Configure request method as POST, set URL as /probe.cgi?all
  request method post url /probe.cgi?all
! Configure header Cookie
  header Cookie Monster
! Configure basic authentication username and password
  credentials Semisweet chips
! Exit to global configuration mode
  exit
```

```
! Enter IOS SLB server farm configuration mode for server farm PUBLIC
ip slb serverfarm PUBLIC
! Configure NAT server and real servers on the server farm
  nat server
  real 10.1.1.1
   inservice
  real 10.1.1.2
   inservice
  real 10.1.1.3
   inservice
  real 10.1.1.4
   inservice
  real 10.1.1.5
   inservice
! Configure ping probe on the server farm
  probe PROBE1
! Configure HTTP probe on the server farm
  probe PROBE2
  end
```

# Layer 3 Switch Configured with IOS SLB Example

Figure 7 shows an example configuration with IOS SLB server connections configured as part of a server farm, using real and virtual servers over Fast Ethernet interfaces.

*Figure 7      Network Configuration for IOS SLB*
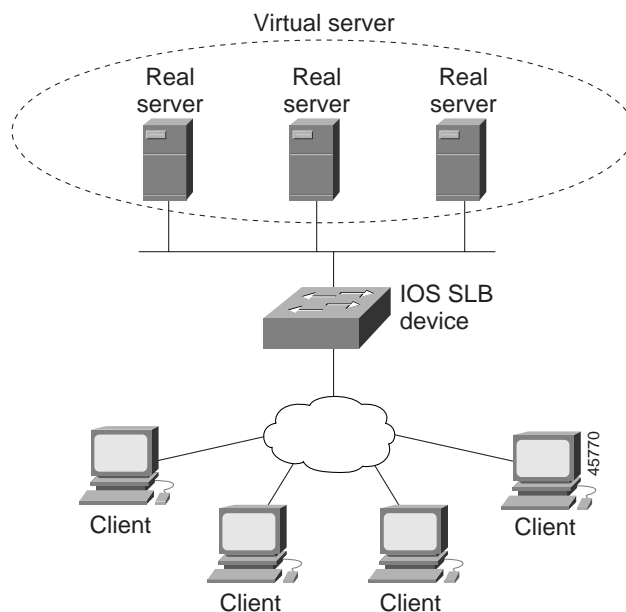


As shown in the following sample configuration, the example topology has three public web servers and two restricted web servers for privileged clients in subnet 10.4.4.x. The public web servers are weighted according to their capacity, with server 10.1.1.2 having the lowest capacity and having a connection limit imposed on it. The restricted web servers are configured as members of the same sticky group, so that HTTP connections and Secure Socket Layer (SSL) connections from the same client use the same real server.

The network configuration to provide the previously described IOS SLB functionality follows:

```
! Unrestricted web server farm
ip slb serverfarm PUBLIC
! Use weighted least connections algorithm
  predictor leastconns
! First real server
  real 10.1.1.1
    weight 16
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Second real server
  real 10.1.1.2
    weight 4
!  Restrict maximum number of connections
    maxconns 1000
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Third real server
  real 10.1.1.3
    weight 24
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Restricted web server farm
ip slb serverfarm RESTRICTED
! Use weighted least connections algorithm
  predictor leastconns
! First real server
  real 10.1.1.20
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Second real server
  real 10.1.1.21
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
!
! Unrestricted web virtual server
ip slb vserver PUBLIC_HTTP
! Handle HTTP requests
  virtual 10.0.0.1 tcp www
! Use public web server farm
  serverfarm PUBLIC
  idle 120
  delay 5
  inservice
!
! Restricted HTTP virtual server
ip slb vserver RESTRICTED_HTTP
! Handle HTTP requests
  virtual 10.0.0.1 tcp www
! Use restricted web server farm
  serverfarm RESTRICTED
! Only allow clients from 10.4.4.x
  client 10.4.4.0 255.255.255.0
```

```
! Couple connections with RESTRICTED_SSL
  sticky 60 group 1
  idle 120
  delay 5
  inservice
!


! Restricted SSL virtual server
ip slb vserver RESTRICTED_SSL
! Handle SSL requests
  virtual 10.0.0.1 tcp https
! Use restricted web server farm
  serverfarm RESTRICTED
! Only allow clients from 10.4.4.x
  client 10.4.4.0 255.255.255.0
! Couple connections with RESTRICTED_WEB
  sticky 60 group 1
  idle 120
  delay 5
  inservice
```

# IOS SLB with NAT Example

Figure 8 shows an example configuration with IOS SLB real server connections configured as part of a server farm, focusing on the configuration of the NAT server and address pool of clients.

*Figure 8       Sample IOS SLB NAT Topology*
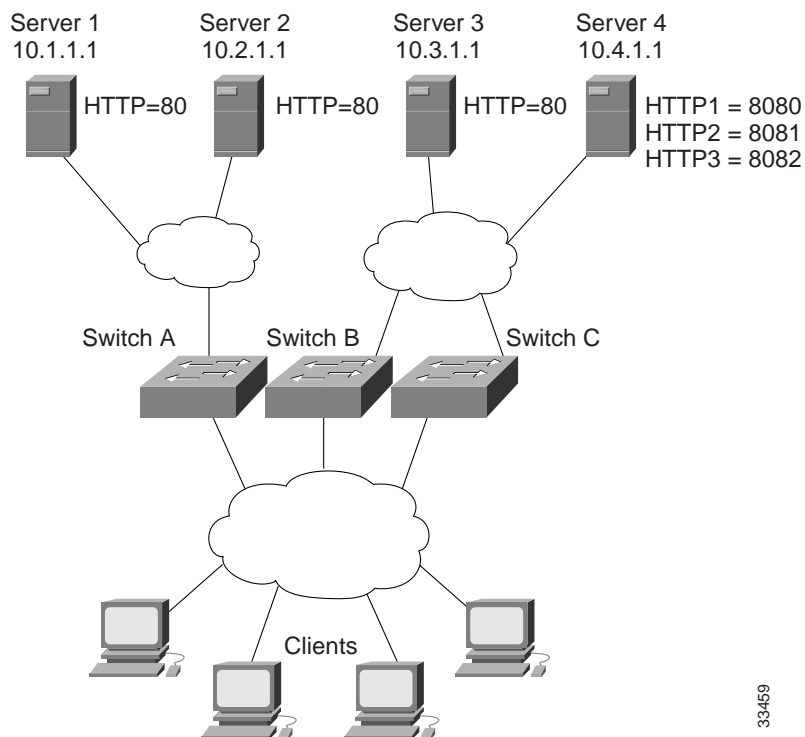
The topology in Figure 8 has four web servers, configured as follows:

- Servers 1, 2, and 3 are running single HTTP server applications listening on port 80.
- Server 4 has multiple HTTP server applications listening on ports 8080, 8081, and 8082.

Server 1 and Server 2 are load-balanced using Switch A, which is performing server address translation.

Server 3 and Server 4 are load-balanced using Switch B and Switch C. These two switches are performing both server and client address translation since there are multiple paths between the clients and the servers. These switches also must perform server port translation for HTTP packets to and from Server 4.

## Switch A Configuration Statements

```
ip slb serverfarm FARM1
! Translate server addresses
  nat server
! Server 1 port 80
  real 10.1.1.1
    reassign 2
    faildetect numconns 4 numclients 2
    retry 20
    inservice
! Server 2 port 80
  real 10.2.1.1
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
!
ip slb vserver HTTP1
! Handle HTTP (port 80) requests
  virtual 128.1.0.1 tcp www
  serverfarm FARM1
  idle 120
  delay 5
  inservice
```

## Switch B Configuration Statements

```
ip slb natpool web-clients 128.3.0.1 128.3.0.254
! NAT address pool for clients
ip slb serverfarm FARM2
! Translate server addresses
  nat server
! Translate client addresses
  nat client web-clients
! Server 3 port 80
  real 10.3.1.1
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Server 4 port 8080
  real 10.4.1.1 port 8080
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
```

```
! Server 4 port 8081
  real 10.4.1.1 port 8081
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Server 4 port 8082
  real 10.4.1.1 port 8082
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
!
ip slb vserver HTTP2
! Handle HTTP (port 80) requests
  virtual 128.2.0.1 tcp www
  serverfarm FARM2
  idle 120
  delay 5
  inservice
```

## Switch C Configuration Statements

```
ip slb natpool web-clients 128.5.0.1 128.5.0.254
! NAT address pool for clients
ip slb serverfarm FARM2
! Translate server addresses
  nat server
! Translate client addresses
  nat client web-clients
! Server 3 port 80
  real 10.3.1.1
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Server 4 port 8080
  real 10.4.1.1 port 8080
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Server 4 port 8081
  real 10.4.1.1 port 8081
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
! Server 4 port 8082
  real 10.4.1.1 port 8082
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
!
ip slb vserver HTTP2
! Handle HTTP (port 80) requests
  virtual 128.4.0.1 tcp www
  serverfarm FARM2
  idle 120
  delay 5
  inservice
```

# Examples of IOS SLB with Stateless Backup

There are several different ways in which you can configure IOS SLB stateless backup. The differences between the configurations depend on the networking capabilities of your load balancing devices, and on the capabilities of the distribution devices that direct client traffic to those load balancing devices.

- If a load balancing device is capable of Layer 2 switching and VLAN trunking (such as the Catalyst 6000 Family Switch), you can wire the device directly to its real servers, and it can handle outbound flows from the real servers while acting as a standby for IOS SLB. HSRP is used on the server-side VLANs of the load balancing device, with the real servers routing to the HSRP address.

- If a load balancing device is *not* capable of both Layer 2 switching and VLAN trunking, you must connect it and its real servers to a Layer 2 switch. This configuration is required in order to use HSRP on the server-side VLANs.

- If a distribution device is capable of Layer 3 switching, it can use route redistribution to direct flows to the active load balancing device.

- If a distribution device is capable of Layer 2 switching, it can use client-side HSRP on the load balancing device to direct flows to the active load balancing device.

- While HSRP offers faster failover times, routing converges quickly enough for most configurations. If you use both client-side and server-side HSRP on the load balancing devices, you must use HSRP interface tracking and priorities to synchronize the client-side and server-side HSRP groups.

This section contains the following examples, illustrating several different IOS SLB stateless backup configurations:

> **Note**  Stateful backup is omitted from these examples in the interest of simplicity. To see an example that uses stateful backup, see the "IOS SLB with Stateful Backup Example" section on page 69.

## Example with Dynamic Routing and Trunking

Figure 9 shows a sample IOS SLB stateless backup configuration with the following characteristics:

- The IP address for real server 1 is 10.10.1.3, and for real server 2 is 10.10.1.4, routed to clients through 10.10.1.100.
- The IP address for the virtual server is 10.10.14.1.
- The IP address for VLAN 1 is 10.10.1.0, with a subnet mask of 255.255.255.0.
- The IP address for Subnet 2 is 10.10.2.0, with a subnet mask of 255.255.255.0.
- The IP address for Subnet 3 is 10.10.3.0, with a subnet mask of 255.255.255.0.
- The distribution device uses EIGRP to learn the route to 10.10.14.1 via either 10.10.2.1 or 10.10.3.1, depending on which IOS SLB is active.

*Figure 9      Stateless Backup with Layer 3 and Trunking*
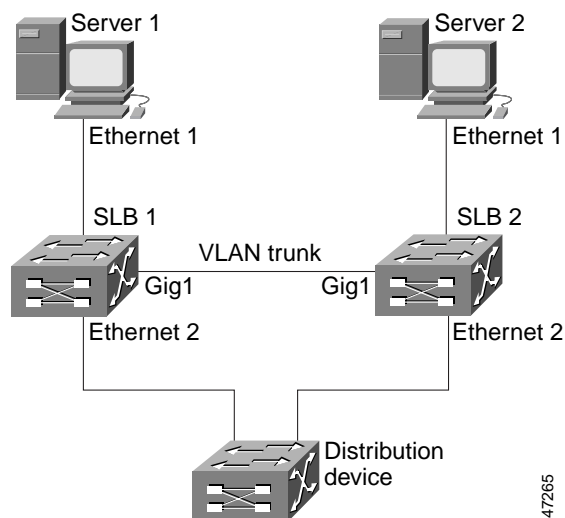


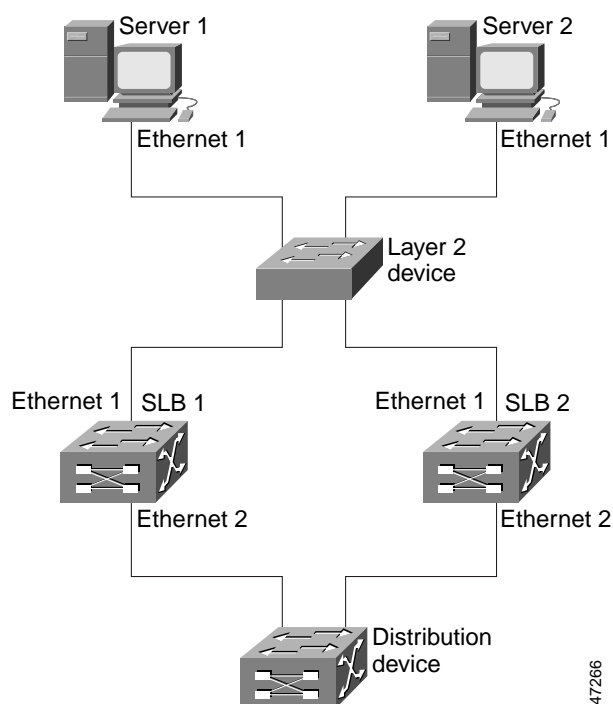## SLB 1 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4 numclients 2
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
  delay 5
  inservice standby SERVER
...
int Eth1
  switchport
  switchport vlan 1
int Eth2
  ip address 10.10.2.1 255.255.255.0
int vlan 1
  ip address 10.10.1.1 255.255.255.0
  standby ip 10.10.1.100
  standby priority 10 preempt delay sync 20
  standby name SERVER
  standby track Eth2
router eigrp 666
  redistribute static
  network 10.0.0.0
```

## SLB 2 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
  delay 5
  inservice standby SERVER
...
int Gig1
  no ip address
  switchport
  switchport trunk encapsulation isl
int Eth1
  switchport
  switchport vlan 1
int Eth2
  ip address 10.10.3.1 255.255.255.0
int vlan 1
  ip address 10.10.1.2 255.255.255.0
  standby ip 10.10.1.100
  standby priority 5 preempt delay sync 20
  standby name SERVER
router eigrp 666
  redistribute static
  network 10.0.0.0
```

# Example with Dynamic Routing and No Trunking

Figure 10 shows a sample IOS SLB stateless backup configuration with the following characteristics:

- The IP address for real server 1 is 10.10.1.3, and for real server 2 is 10.10.1.4, routed to clients through 10.10.1.100.

- The IP address for the virtual server is 10.10.14.1.

- The IP address for Subnet 2 is 10.10.2.0, with a subnet mask of 255.255.255.0.

- The IP address for Subnet 3 is 10.10.3.0, with a subnet mask of 255.255.255.0.

- The distribution device uses EIGRP to learn the route to 10.10.14.1 via either 10.10.2.2 or 10.10.3.2, depending on which IOS SLB is active.

*Figure 10    Stateless Backup with Layer 3 and No Trunking*



## SLB 1 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
```

```
      delay 5
      inservice standby SERVER
...
int Eth1
   ip address 10.10.1.1 255.255.255.0
   standby ip 10.10.1.100
   standby priority 10 preempt delay sync 20
   standby name SERVER
   standby track Eth2
int Eth2
   ip address 10.10.2.1 255.255.255.0
router eigrp 666
   redistribute static
   network 10.0.0.0
```

## SLB 2 Configuration Statements

```
ip slb serverfarm SF1
   real 10.10.1.3
      reassign 2
      faildetect numconns 4
      retry 20
      inservice
   real 10.10.1.4
      reassign 2
      faildetect numconns 4
      retry 20
      inservice
ip slb vserver VS1
   virtual 10.10.14.1 tcp www
   serverfarm SF1
   idle 120
   delay 5
   inservice standby SERVER
...
int Eth1
   ip address 10.10.1.2 255.255.255.0
   standby ip 10.10.1.100
   standby priority 5 preempt delay sync 20
   standby name SERVER
int Eth2
   ip address 10.10.3.1 255.255.255.0
router eigrp 666
   redistribute static
   network 10.0.0.0
```

# Example with Static Routing and Trunking

Figure 11 shows a sample IOS SLB stateless backup configuration with the following characteristics:

- The IP address for real server 1 is 10.10.1.3, and for real server 2 is 10.10.1.4, routed to clients through 10.10.1.100.

- The IP address for the virtual server is 10.10.14.1.

- The IP address for VLAN 1 is 10.10.1.0, with a subnet mask of 255.255.255.0.

- The IP address for Subnet 2 is 10.10.2.0, with a subnet mask of 255.255.255.0.

- The IP address for Subnet 3 is 10.10.3.0, with a subnet mask of 255.255.255.0.

- The configuration uses static routing to the HSRP route on the distribution device.

*Figure 11    Stateless Backup with Layer 2 and Trunking*
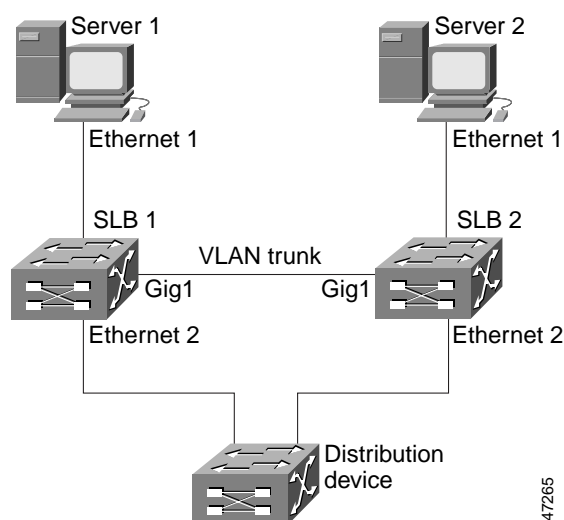


## SLB 1 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
  delay 5
  inservice standby SERVER
...
int Eth1
  switchport
  switchport vlan 1
```

```
int Eth2
  ip address 10.10.2.1 255.255.255.0
  standby ip 10.10.2.100
  standby priority 10 preempt delay sync 20
  standby track vlan1
int vlan 1
  ip address 10.10.1.1 255.255.255.0
  standby ip 10.10.1.100
  standby priority 10 preempt delay sync 20
  standby name SERVER
  standby track Eth2
```

## SLB 2 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
  delay 5
  inservice standby SERVER
...
int Gig1
  no ip address
  switchport
  switchport trunk encapsulation isl
int Eth1
  switchport
  switchport vlan 1
int Eth2
  ip address 10.10.2.2 255.255.255.0
  standby ip 10.10.2.100
  standby priority 5 preempt delay sync 20
int vlan 1
  ip address 10.10.1.2 255.255.255.0
  standby ip 10.10.1.100
  standby priority 5 preempt delay sync 20
  standby name SERVER
```

## Distribution Device Configuration Statements

```
int Eth1
  switchport
  switchport distribution vlan 2
int Eth2
  switchport
  switchport distribution vlan 2
int vlan2
  ip address 10.10.2.3 255.255.255.0
  no shut
ip route 10.10.14.1 255.255.255.255 10.10.2.100
```

## Example with Static Routing and No Trunking

Figure 12 shows a sample IOS SLB stateless backup configuration with the following characteristics:

- The IP address for real server 1 is 10.10.1.3, and for real server 2 is 10.10.1.4, routed to clients through 10.10.1.100.

- The IP address for the virtual server is 10.10.14.1.

- The IP address for Subnet 2 is 10.10.2.0, with a subnet mask of 255.255.255.0.

- The IP address for Subnet 3 is 10.10.3.0, with a subnet mask of 255.255.255.0.

- The configuration uses static routing to the HSRP route on the distribution device.

*Figure 12      Stateless Backup with Layer 2 and No Trunking*
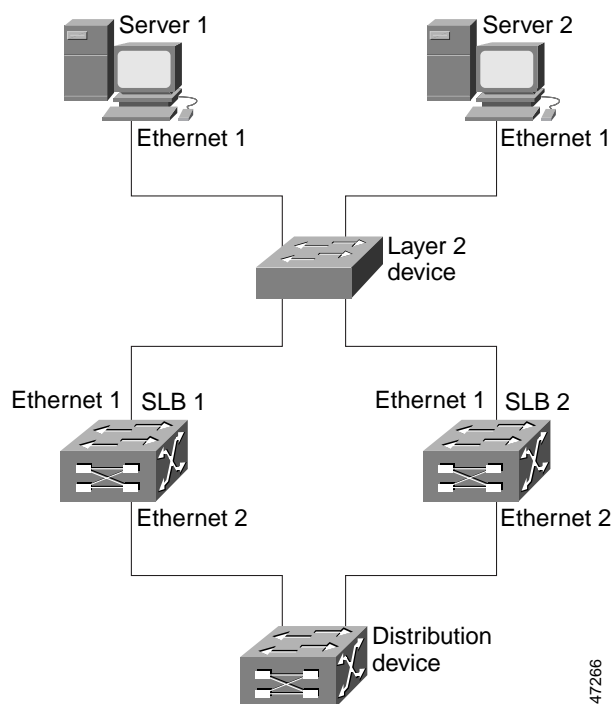


### SLB 1 Configuration Statements

```
ip slb serverfarm SF1
  real 10.10.1.3
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
  real 10.10.1.4
    reassign 2
    faildetect numconns 4
    retry 20
    inservice
ip slb vserver VS1
  virtual 10.10.14.1 tcp www
  serverfarm SF1
  idle 120
  delay 5
```

```
      inservice standby SERVER
...
int Eth1
   ip address 10.10.1.1 255.255.255.0
   standby ip 10.10.1.100
   standby priority 10 preempt delay sync 20
   standby name SERVER
   standby track eth2
int Eth2
   ip address 10.10.2.1 255.255.255.0
   standby ip 10.10.2.100
   standby priority 10 preempt delay sync 20
   standby track eth1
```

## SLB 2 Configuration Statements

```
ip slb serverfarm SF1
   real 10.10.1.3
      reassign 2
      faildetect numconns 4
      retry 20
      inservice
   real 10.10.1.4
      reassign 2
      faildetect numconns 4
      retry 20
      inservice
ip slb vserver VS1
   virtual 10.10.14.1 tcp www
   serverfarm SF1
   idle 120
   delay 5
   inservice standby SERVER
...
int Eth1
   ip address 10.10.1.2 255.255.255.0
   standby ip 10.10.1.100
   standby priority 5 preempt delay sync 20
   standby name SERVER
int Eth2
   ip address 10.10.2.2 255.255.255.0
   standby ip 10.10.2.100
   standby priority 5 preempt delay sync 20
```

## Distribution Device Configuration Statements

```
int Eth1
   switchport
   switchport distribution vlan 2
int Eth2
   switchport
   switchport distribution vlan 2
int vlan2
   ip address 10.10.2.3 255.255.255.0
   no shut
ip route 10.10.14.1 255.255.255.255 10.10.2.100
```

# IOS SLB with Stateful Backup Example

This example configuration focuses on the IOS SLB real server connections configured as part of a server farm, with real and virtual servers over Fast Ethernet interfaces configured with stateful backup standby connections.

Figure 13 is an example of a stateful backup configuration, using HSRP on both the client and server sides to handle failover. The real servers route outbound flows to 10.10.3.100, which is the HSRP address on the server side interfaces. The client (or access router), routes to the virtual IP address (10.10.10.12) through 10.10.2.100, HSRP address on the client side.

Notice the loopback interfaces configured on both devices for the exchange of these messages. Each IOS SLB should also be given duplicate routes to the other switch loopback address. This allows replication messages to flow despite an interface failure.

**Note**   To allow HSRP to function properly, **set spantree portfast** must be configured on any Layer 2 device between the IOS SLB switches.

*Figure 13      IOS SLB Stateful Environment*
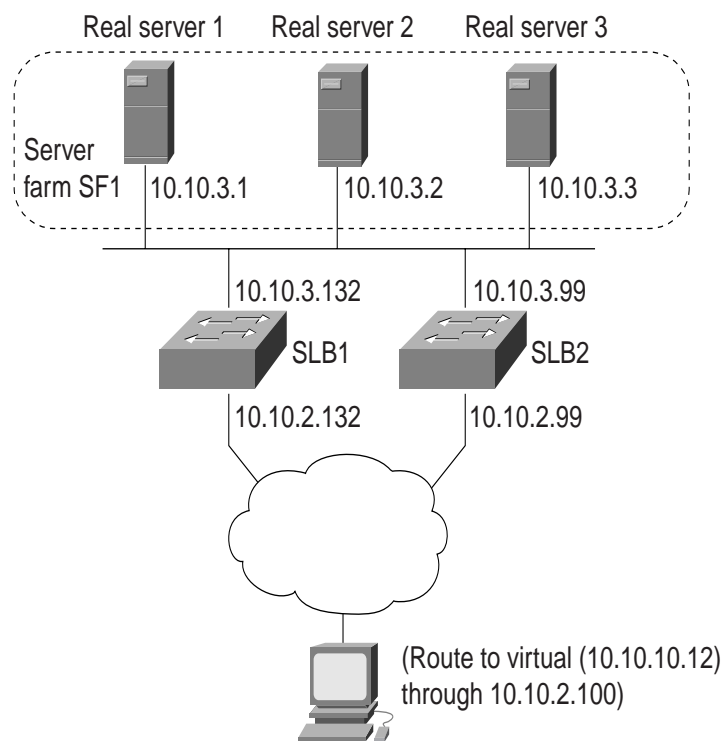
## Switch SLB1 Configuration Statements

```
ip slb serverfarm SF1
  nat server
  real 10.10.3.1
    inservice
  real 10.10.3.2
    inservice
  real 10.10.3.3
    inservice
!
ip slb vserver VS1
  virtual 10.10.10.12 tcp telnet
  serverfarm SF1
  replicate casa 10.10.99.132 10.10.99.99 1024 password PASS
  inservice standby virt
!
interface loopback 1
  ip address 10.10.99.132 255.255.255.255
!
interface FastEthernet1
  ip address 10.10.3.132 255.255.255.0
  no ip redirects
  no ip mroute-cache
  standby priority 5 preempt delay sync 20
  standby name out
  standby ip 10.10.3.100
  standby track FastEthernet2
!
interface FastEthernet2
  ip address 10.10.2.132 255.255.255.0
  no ip redirects
  standby priority 5 preempt delay sync 20
  standby name virt
  standby ip 10.10.2.100
  standby track FastEthernet1
```

## Switch SLB2 Configuration Statements

```
ip slb serverfarm SF1
  nat server
  real 10.10.3.1
    inservice
  real 10.10.3.2
    inservice
  real 10.10.3.3
    inservice
!
ip slb vserver VS1
  virtual 10.10.10.12 tcp telnet
  serverfarm SF1
  replicate casa 10.10.99.99 10.10.99.132 1024 password PASS
  inservice standby virt
!
interface loopback 1
  ip address 10.10.99.99 255.255.255.255
!
interface FastEthernet2
  ip address 10.10.2.99 255.255.255.0
  no ip redirects
  no ip route-cache
  no ip mroute-cache
```

```
  standby priority 10 preempt delay sync 20
  standby name virt
  standby ip 10.10.2.100
  standby track FastEthernet3
!
interface FastEthernet3
  ip address 10.10.3.99 255.255.255.0
  no ip redirects
  no ip route-cache
  no ip mroute-cache
  standby priority 10 preempt delay sync 20
  standby name out
  standby ip 10.10.3.100
  standby track FastEthernet2
```

# IOS SLB with Active Standby Example

Figure 14 shows an IOS SLB network configured for active standby, with two IOS SLB devices load-balancing the same virtual IP address while backing up each other. If either device fails, the other takes over its load via normal HSRP failover and IOS SLB stateless redundancy.

*Figure 14    IOS SLB Active Standby*
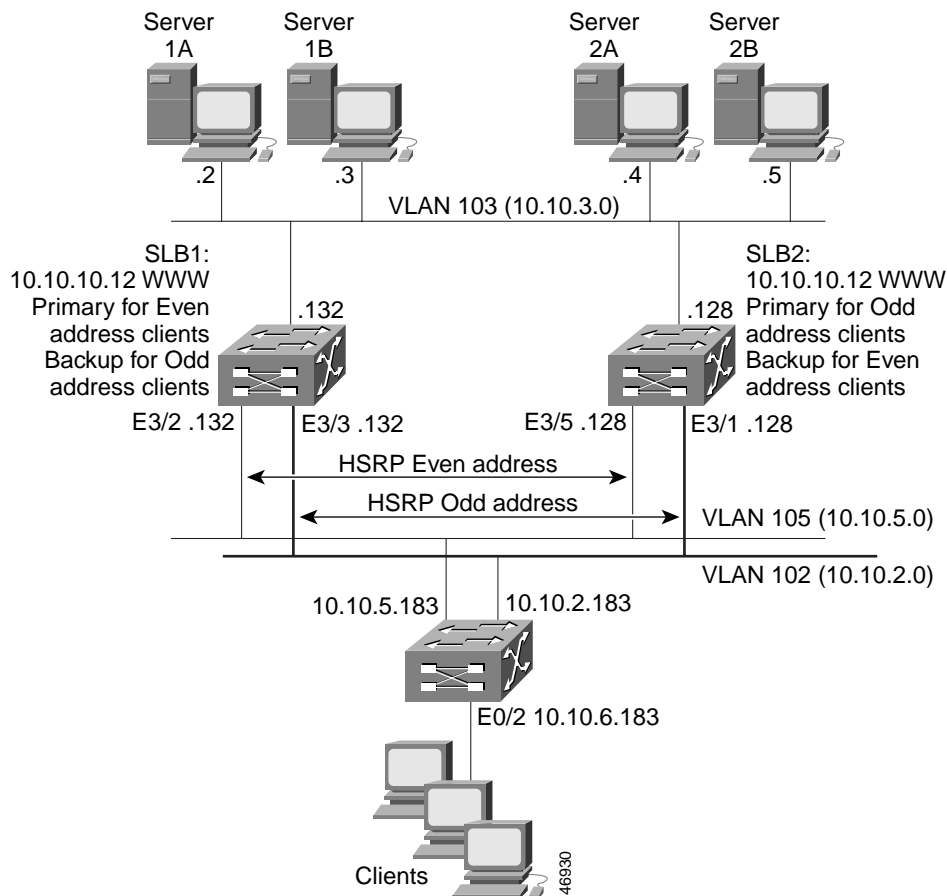
The sample network configuration in Figure 14 has the following characteristics:

- SLB 1 balances servers 1A and 1B and SLB 2 balances 2A and 2B.

- A single virtual IP address (10.10.10.12 for web) is supported across the two IOS SLB devices.

- Client traffic is divided in an access router, sending clients with even IP addresses to HSRP1 (10.10.5.100) and clients with odd IP addresses to HSRP2 (10.10.2.100). SLB 1 is configured as primary for clients with odd IP addresses, and SLB 2 is primary for clients with even IP addresses.

- The IOS SLB devices balance the traffic to disjoint sets of real servers. (If client NAT was used in this example, this would not be a requirement).

- Each set of real servers has a default gateway configured to its IOS SLB device.

- The HSRP address on VLAN 105 is 10.10.5.100. The HSRP address on VLAN 102 is 10.10.2.100.

## SLB 1 Configuration Statements

```
ip slb serverfarm EVEN
 nat server
 real 10.10.3.2
  reassign 2
  faildetect numconns 4 numclients 2
  retry 20
  inservice
 real 10.10.3.3
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
!
ip slb serverfarm ODD
 nat server
 real 10.10.3.2
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
 real 10.10.3.3
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
!
!-----Same EVEN virtual server as in SLB 2
ip slb vserver EVEN
 virtual 10.10.10.12 tcp www
 serverfarm EVEN
 client 0.0.0.0 0.0.0.1
 idle 120
 delay 5
!-----See standby name in Ethernet 3/3 below
 inservice standby STANDBY_EVEN
!
!-----Same ODD virtual server as in SLB 2
ip slb vserver ODD
 virtual 10.10.10.12 tcp www
 serverfarm ODD
 client 0.0.0.1 0.0.0.1
 idle 120
 delay 5
```

```
!-----See standby name in Ethernet 3/2 below
 inservice standby STANDBY_ODD
!
interface Ethernet3/2
 ip address 10.10.5.132 255.255.255.0
 standby priority 20 preempt delay sync 20
!-----See standby name in SLB 2, Ethernet 3/5
 standby name STANDBY_ODD
 standby ip 10.10.5.100
!
interface Ethernet3/3
 ip address 10.10.2.132 255.255.255.0
 standby priority 10
!-----See standby name in SLB 2, Ethernet 3/1
 standby name STANDBY_EVEN
 standby ip 10.10.2.100
```

## SLB 2 Configuration Statements

```
ip slb serverfarm EVEN
 nat server
 real 10.10.3.4
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
 real 10.10.3.5
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
!
ip slb serverfarm ODD
 nat server
 real 10.10.3.4
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
 real 10.10.3.5
  reassign 2
  faildetect numconns 4
  retry 20
  inservice
!
!-----Same EVEN virtual server as in SLB 1
ip slb vserver EVEN
 virtual 10.10.10.12 tcp www
 serverfarm EVEN
 client 0.0.0.0 0.0.0.1
 idle 120
 delay 5
!-----See standby name in Ethernet 3/1 below
 inservice standby STANDBY_EVEN
!
!-----Same ODD virtual server as in SLB 1
ip slb vserver ODD
 virtual 10.10.10.12 tcp www
 serverfarm ODD
 client 0.0.0.1 0.0.0.1
 idle 120
 delay 5
```

```
!-----See standby name in Ethernet 3/5 below
 inservice standby STANDBY_ODD
!
interface Ethernet3/1
 ip address 10.10.2.128 255.255.255.0
 standby priority 20 preempt delay sync 20
!-----See standby name in SLB 1, Ethernet 3/3
 standby name STANDBY_EVEN
 standby ip 10.10.2.100
!
interface Ethernet3/5
 ip address 10.10.5.128 255.255.255.0
 standby priority 10 preempt delay sync 20
!-----See standby name in SLB 1, Ethernet 3/2
 standby name STANDBY_ODD
 standby ip 10.10.5.100
```

## Access Router Configuration Statements

```
interface Ethernet0/0
 ip address 10.10.5.183 255.255.255.0
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet0/1
 ip address 10.10.2.183 255.255.255.0
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet0/2
 ip address 10.10.6.183 255.255.255.0
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 ip policy route-map virts
!
access-list 100 permit ip 0.0.0.1 255.255.255.254 host 10.10.10.12
access-list 101 permit ip 0.0.0.0 255.255.255.254 host 10.10.10.12
route-map virts permit 10
match ip address 100
set ip next-hop 10.10.5.100
!
route-map virts permit 15
match ip address 101
set ip next-hop 10.10.2.100
```

# IOS SLB with Redistribution of Static Routes Example

Figure 15 shows an IOS SLB network configured to distribute static routes to a virtual server's IP address. The route to the address is added to the routing table as **static** if you *advertise* the address when you bring the virtual server into service (using the **inservice** command). See the description of the **advertise** command in the "Command Reference" section on page 91 for more details about advertising virtual server IP addresses.

Because the routing configuration varies from protocol to protocol, sample configurations for several different routing protocols are given.

*Figure 15     IOS SLB Redistribution of Static Routes*
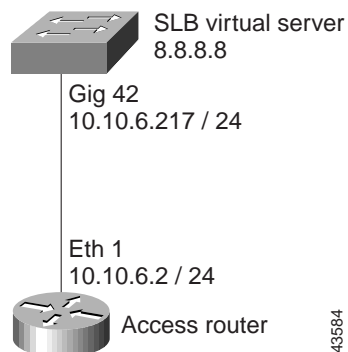


## Routing Information Protocol (RIP)

Following is the RIP static route redistribution configuration for the IOS SLB switch shown in Figure 15:

```
router rip
 redistribute static
 network 10.0.0.0
 network 8.0.0.0
```

Following is the RIP static route redistribution configuration for the access router that is listening for routing updates shown in Figure 15:

```
router rip
 network 10.0.0.0
 network 8.0.0.0
```

## Open Shortest Path First (OSPF)

Following is the OSPF static route redistribution configuration for the IOS SLB switch shown in Figure 15:

```
router ospf 1
 redistribute static subnets
 network 10.10.6.217 0.0.0.0 area 0
 network 8.8.8.0 0.0.0.255 area 0
```

Following is the OSPF static route redistribution configuration for the access router that is listening for routing updates shown in Figure 15:

```
router ospf 1
 network 10.10.6.2 0.0.0.0 area 0
 network 8.8.8.0 0.0.0.255 area 0
```

## Interior Gateway Routing Protocol (IGRP)

Following is the IGRP static route redistribution configuration for the IOS SLB switch shown in Figure 15:

```
router igrp 1
 redistribute connected
 redistribute static
 network 8.0.0.0
 network 10.0.0.0
```

Following is the IGRP static route redistribution configuration for the access router that is listening for routing updates shown in Figure 15:

```
router igrp 1
 network 8.0.0.0
 network 10.0.0.0
```

## Enhanced Interior Gateway Routing Protocol (Enhanced IGRP)

Following is the Enhanced IGRP static route redistribution configuration for the IOS SLB switch shown in Figure 15:

```
router eigrp 666
 redistribute static
 network 10.0.0.0
 network 8.0.0.0
```

Following is the Enhanced IGRP static route redistribution configuration for the access router that is listening for routing updates shown in Figure 15:

```
router eigrp 666
 network 10.0.0.0
 network 8.0.0.0
```

# Examples of IOS SLB with WAP Load Balancing

Figure 16 shows an IOS SLB network configured to balance WAP flows. In this example:

- WAP flows are balanced between WAP gateways 10.10.2.1, 10.10.2.2, and 10.10.2.3.

- The clients connect to 10.10.1.1, the IOS SLB virtual server address.

- For a given session, load-balancing decisions change if the connection idles longer than the virtual server's idle connection timer (3000 seconds in this example).

*Figure 16    IOS SLB with WAP Load Balancing*
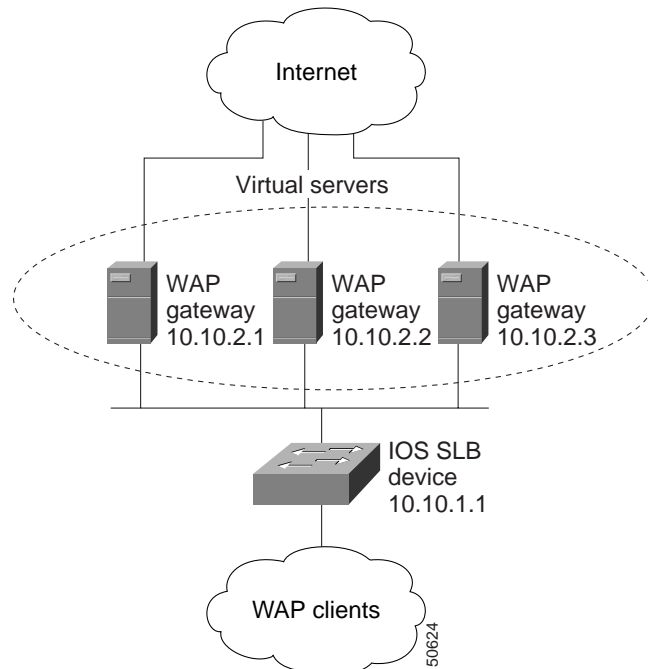


There are two ways to configure IOS SLB load balancing for WAP:

- To load balance sessions running in connection-oriented WSP mode, define a WSP probe and use WAP load balancing. WAP load balancing requires a WAP virtual server configured on one of the WAP ports.

- To load balance sessions running in connectionless WSP, connectionless secure WSP, and connection-oriented secure WSP modes, define a ping or WSP probe and use standard UDP load balancing with a low idle timer.

## Example with WAP Load Balancing

The following example shows the configuration for the IOS SLB device shown in Figure 16, which balances WAP flows on UDP port 9201 (WSP/WTP/UDP):

```
ip slb probe PROBE3 wsp
  url http://localhost/test.txt
!
ip slb serverfarm WAPFARM
  nat server
  real 10.10.2.1
  inservice
  real 10.10.2.2
  inservice
  real 10.10.2.3
  inservice
  probe PROBE3
!
ip slb vserver VSERVER
  virtual 10.10.1.1 udp 9201
  serverfarm WAPFARM
  idle 3000
  inservice
```

## Example with UDP Load Balancing

The following example shows the configuration for the IOS SLB device shown in Figure 16, which balances WAP flows on UDP port 9203 (WSP/WTP/WTLS/UDP):

```
ip slb probe PROBE1 ping
!
ip slb serverfarm WAPFARM
  nat server
  real 10.10.2.1
  inservice
  real 10.10.2.2
  inservice
  real 10.10.2.3
  inservice
  probe PROBE1
!
ip slb vserver VSERVER
  virtual 10.10.1.1 udp 9203
  serverfarm WAPFARM
  idle 3000
  inservice
```

# Examples of IOS SLB with Route Health Injection

This section contains the following examples, illustrating several different IOS SLB route health injection configurations:

# Example with Two Distributed Sites with One Web Server Each

Figure 17 shows an IOS SLB network configured with route health injection with the following characteristics:

- Both IOS SLB devices are configured with the same virtual IP address.
- Each IOS SLB device has a server farm containing only the locally attached web server as a real server.
- The path to SLB A has the lower weight.

*Figure 17    Two Distributed Sites with One Web Server Each*
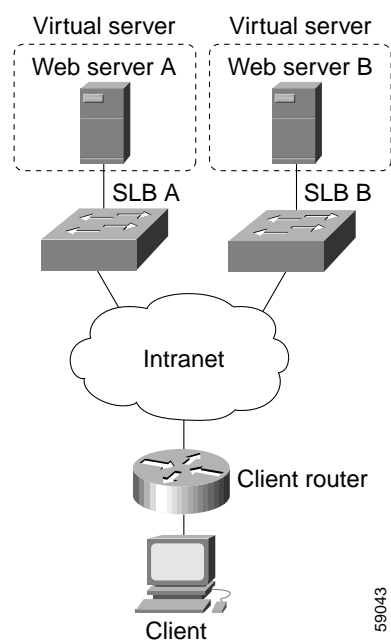


When both web servers in Figure 17 are operational, the client router receives the host route from both IOS SLB devices.

If Web Server A fails, the virtual server for the virtual IP address on SLB A enters **FAILED** state and stops advertising the host route for the virtual IP address. The client router then begins using the route to SLB B.

When Web Server A is again available, the virtual server again advertises the host route for the virtual IP address, and the client router begins using SLB A.

# Example with Two Distributed Sites with Two Web Servers Each

Figure 18 shows an IOS SLB network configured with route health injection with the following characteristics:

- Both IOS SLB devices are configured with the same virtual IP address.
- Each IOS SLB device has a server farm containing two locally attached web servers as real servers.
- The path to SLB A has the lower weight.

*Figure 18    Two Distributed Sites with Two Web Servers Each*
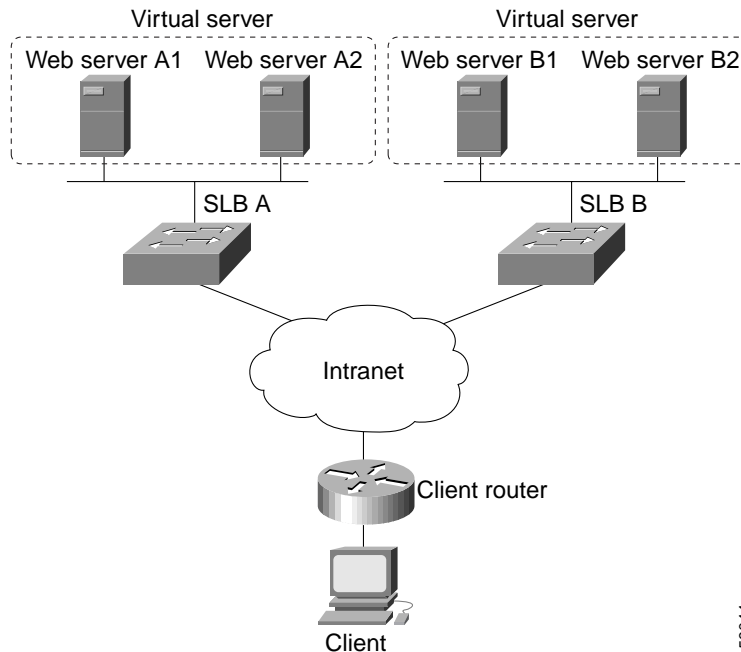


When all web servers in Figure 18 are operational, the client router receives the host route from both IOS SLB devices.

If one web server in either server farm fails, the route continues to be advertised by the given IOS SLB device.

If both Web Server A1 and Web Server A2 fail, the virtual server for the virtual IP address on SLB A enters **FAILED** state and stops advertising the host route for the virtual IP address. The client router then begins using the route to SLB B.

When either Web Server A1 or Web Server A2 is again available, the virtual server again advertises the host route for the virtual IP address, and the client router begins using SLB A.

## Example with Two Distributed Sites with One Web Server and a Backup IOS SLB Switch Each

Figure 19 shows an IOS SLB network configured with route health injection with the following characteristics:

- Both IOS SLB devices are configured with the same virtual IP address.
- Each IOS SLB device has a server farm containing only the locally attached web server as a real server.
- Each site has a primary IOS SLB device and a backup IOS SLB device.
- The path to SLB A has the lower weight.

*Figure 19     Two Distributed Sites with One Web Server and a Backup IOS SLB Switch Each*
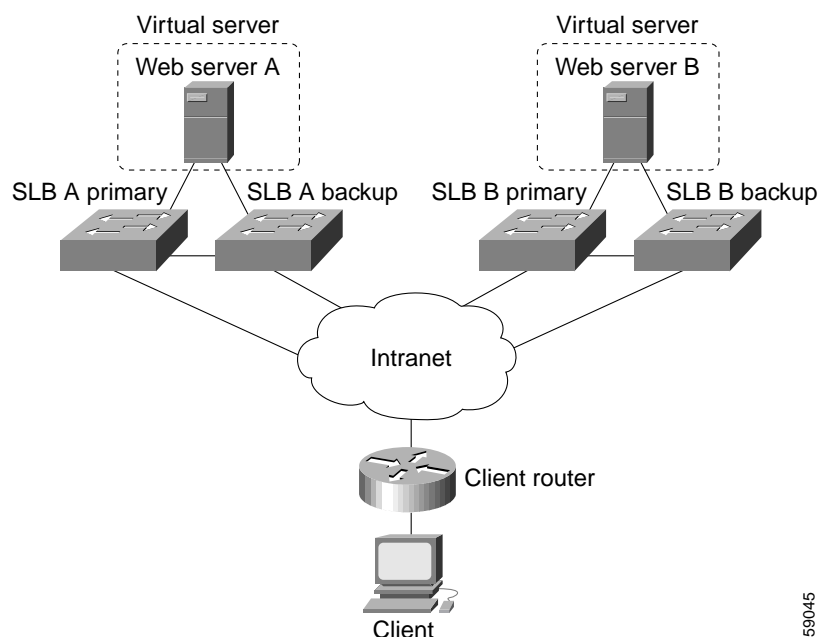


When both web servers in Figure 19 are operational, the client router receives the host route from both SLB A Primary and SLB B Primary.

If SLB A Primary fails, SLB A Backup begins advertising the host route to the virtual IP address. If SLB A Backup also fails, the virtual server for the virtual IP address on SLB A Primary and SLB A Backup enters **FAILED** state and stops advertising the host route for the virtual IP address. The client router then begins using the route to SLB B Primary (or to SLB B Backup, if SLB B Primary is not available).

When either SLB A Primary or SLB A Backup is again available, the virtual server again advertises the host route for the virtual IP address, and the client router begins using SLB A Primary or SLB A Backup.

# IOS SLB with GPRS Load Balancing Example

Figure 20 shows a typical GPRS load-balancing configuration. In this configuration:

- IOS SLB can balance GPRS flows across multiple real GGSNs. The SGSN "sees" the real GGSNs as a single virtual GGSN. This configuration increases the flow-handling capability of the real GGSNs and increases the reliability and availability.

- The virtual template address of the SGSN is 10.111.111.111.

- The virtual template address of GGSN1 is 192.168.1.1.

- The virtual template address of GGSN2 is 192.168.2.2.

- The virtual template address of GGSN3 is 192.168.3.3.

*Figure 20    IOS SLB with GPRS Load Balancing*
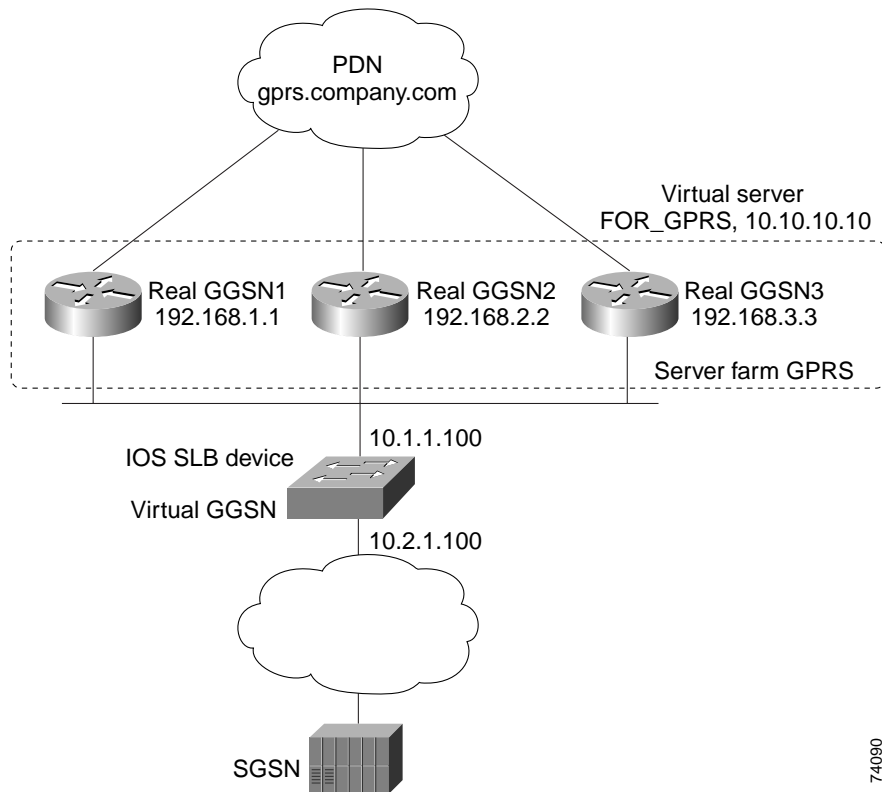


Following are the configuration statements for the configuration shown in Figure 20:

For more detailed GGSN configuration examples, see the *Cisco IOS Mobile Wireless Configuration Guide*.

## IOS SLB Configuration Statements

```
hostname GTP_SLB
!
ip domain-name gprs.com
!
ip slb serverfarm GPRS
 real 192.168.1.1
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
 real 192.168.2.2
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
 real 192.168.3.3
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
ip slb vserver FOR_GPRS
 virtual 10.10.10.10 udp 3386 service gtp
 serverfarm GPRS
 inservice
!
ip slb dfp password Cookies 0
 agent 10.1.1.201 1111 30 0 10
 agent 10.1.1.202 1111 30 0 10
 agent 10.1.1.203 1111 30 0 10
!
interface FastEthernet1/0
 description TO SERVERFARM GPRS
 ip address 10.1.1.100 255.255.255.0
 no ip redirects
 duplex half
!
interface FastEthernet3/0
 description TO SGSN
 ip address 10.2.1.100 255.255.255.0
 no ip mroute-cache
 duplex half
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
ip route 192.168.1.1 255.255.255.255 10.1.1.201
ip route 192.168.2.2 255.255.255.255 10.1.1.202
ip route 192.168.3.3 255.255.255.255 10.1.1.203
```

## GGSN1 Configuration Statements

```
service gprs ggsn
!
hostname GGSN1
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
```

```
interface loopback 1
 description LOOPBACK SAME AS IOS SLB VSERVER ADDRESS
 ip address 10.10.10.10 255.255.255.255
 no ip route-cache
 no ip mroute-cache
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.201 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
!
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.1.1 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
gprs slb cef 10.10.10.10
```

## GGSN2 Configuration Statements

```
service gprs ggsn
!
hostname GGSN2
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
interface loopback 1
 description LOOPBACK SAME AS IOS SLB VSERVER ADDRESS
 ip address 10.10.10.10 255.255.255.255
 no ip route-cache
 no ip mroute-cache
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.202 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
!
```

```
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.2.2 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
gprs slb cef 10.10.10.10
```

## GGSN3 Configuration Statements

```
service gprs ggsn
!
hostname GGSN3
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
interface loopback 1
 description LOOPBACK SAME AS IOS SLB VSERVER ADDRESS
 ip address 10.10.10.10 255.255.255.255
 no ip route-cache
 no ip mroute-cache
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.203 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
!
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.3.3 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
```

```
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
gprs slb cef 10.10.10.10
```

# IOS SLB with GPRS Load Balancing and NAT Example

The following example uses the same basic configuration as in the "IOS SLB with GPRS Load Balancing Example" section on page 82, including the network shown in Figure 20, but with the addition of NAT:

For more detailed GGSN configuration examples, see the *Cisco IOS Mobile Wireless Configuration Guide*.

## IOS SLB Configuration Statements

```
hostname GTP_SLB
!
ip domain-name gprs.com
!
ip slb serverfarm GPRS
 nat server
 real 192.168.1.1
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
 real 192.168.2.2
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
 real 192.168.3.3
  weight 1
  faildetect numconns 1 numclients 1
  inservice
!
ip slb vserver FOR_GPRS
 virtual 10.10.10.10 udp 3386 service gtp
 serverfarm GPRS
 inservice
!

ip slb dfp password Cookies 0
```

```
 agent 10.1.1.201 1111 30 0 10
 agent 10.1.1.202 1111 30 0 10
 agent 10.1.1.203 1111 30 0 10
!
interface FastEthernet1/0
 description TO SERVERFARM GPRS
 ip address 10.1.1.100 255.255.255.0
 no ip redirects
 duplex half
!
interface FastEthernet3/0
 description TO SGSN
 ip address 10.2.1.100 255.255.255.0
 no ip mroute-cache
 duplex half
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
ip route 192.168.1.1 255.255.255.255 10.1.1.201
ip route 192.168.2.2 255.255.255.255 10.1.1.202
ip route 192.168.3.3 255.255.255.255 10.1.1.203
```

## GGSN1 Configuration Statements

```
service gprs ggsn
!
hostname GGSN1
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.201 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.1.1 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
```

## GGSN2 Configuration Statements

```
service gprs ggsn
!
hostname GGSN2
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.202 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.2.2 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
```

## GGSN3 Configuration Statements

```
service gprs ggsn
!
hostname GGSN3
!
ip dfp agent gprs
 port 1111
 password Cookies 0
 inservice
!
ip domain-name gprs.com
!
interface FastEthernet1/0
 description TO SLB
 ip address 10.1.1.203 255.255.255.0
 ip directed-broadcast
 no ip mroute-cache
 duplex half
!
```

```
interface Virtual-Template1
 description GTP VIRTUAL TEMPLATE
 ip address 192.168.3.3 255.255.255.0
 encapsulation gtp
 gprs access-point-list gprs1
!
ip route 10.111.111.111 255.255.255.255 FastEthernet1/0
!
gprs access-point-list gprs1
  access-point 1
    access-point-name gprs.company.com
    access-mode non-transparent
    ip-address-pool dhcp-proxy-client
    dhcp-server 10.100.0.5 10.100.0.6
    dhcp-gateway-address 10.27.3.1
    exit
!
gprs maximum-pdp-context-allowed 45000
gprs qos map canonical-qos
gprs gtp path-echo-interval 0
gprs dfp max-weight 32
```

# IOS SLB with Sticky Connections Example

The following sample configuration assigns all HTTP connections from a subnet to the same real server in server farm PUBLIC:

```
ip slb vserver http
  serverfarm PUBLIC
  sticky 30 group 1 netmask 255.255.255.248
  virtual 20.20.20.20 tcp 80
  inservice
```

The following sample configuration adds HTTPS connections to the above configuration, using the same sticky information but with a different virtual server:

```
ip slb vserver https
  serverfarm PUBLIC
  sticky 30 group 1 netmask 255.255.255.248
  virtual 20.20.20.20 tcp 443
  inservice
```

Now, all HTTP *and* HTTPS connections from the subnet are assigned to the same real server. For example, if a user connects to HTTP, then a second user connects to HTTPS, both connections are assigned to the same real server.

# IOS SLB with Transparent Webcache Load Balancing

In the following sample configuration, virtual server WEBCACHE examines all web flows passing through the load-balancing device and dispatches them to server farm WEBCACHE-FARM. The **client exclude** statement excludes flows originating from subnet 80.80.7.0, enabling the real servers 80.80.7.188 and 80.80.7.189 to communicate with the Internet as needed.

```
ip slb serverfarm WEBCACHE-FARM
  real 80.80.7.188
   inservice
  real 80.80.7.189
   inservice
!
ip slb vserver WEBCACHE
  virtual 0.0.0.0 0.0.0.0 tcp www
  serverfarm WEBCACHE-FARM
  client 80.80.7.0 255.255.255.0 exclude
  inservice
```

# Command Reference

This section documents only new and modified commands.

# access

To route specific flows to a firewall farm, use the **access** firewall farm configuration command. To restore the default settings, use the **no** form of this command.

> **access** [**source** *source-ip-address network-mask*] [**destination** *destination-ip-address network-mask*]

> **no access** [**source** *source-ip-address network-mask*] [**destination** *destination-ip-address network-mask*]

| Syntax Description | | |
|---|---|---|
| | **source** | (Optional) Routes flows based on source IP address. |
| | *source-ip-address* | (Optional) Specifies the source IP address. The default is 0.0.0.0 (all sources). |
| | *network-mask* | (Optional) Specifies the source IP network mask. The default is 0.0.0.0 (all source subnets). |
| | **destination** | (Optional) Routes flows based on destination IP address. |
| | *destination-ip-address* | (Optional) Specifies the destination IP address. The default is 0.0.0.0 (all destinations). |
| | *network-mask* | (Optional) Specifies the destination IP network mask. The default is 0.0.0.0 (all destination subnets). |

**Defaults**

The default source IP address is 0.0.0.0 (route flows from all sources to this firewall farm).

The default source IP network mask is 0.0.0.0 (route flows from all source subnets to this firewall farm).

The default destination IP address is 0.0.0.0 (route flows from all destinations to this firewall farm).

The default destination IP network mask is 0.0.0.0 (route flows from all destination subnets to this firewall farm).

**Command Modes**

Firewall farm configuration

| Command History | Release | Modification |
|---|---|---|
| | 12.1(7)E | This command was introduced. |

**Usage Guidelines**

You can specify more than one source or destination for each firewall farm. To do so, configure multiple access statements, making sure the network masks do not overlap each other.

**Examples**

The following example routes flows with a destination IP address of *10.1.6.0* to firewall farm *FIRE1*:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# access destination 10.1.6.0 255.255.255.0
```

**Related Commands**

| Command | Description |
| --- | --- |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |

# address (HTTP probe)

To configure an IP address to which to send HTTP probes, use the **address** HTTP probe configuration command. To restore the default settings, use the **no** form of this command.

**address** [*ip-address*]

**no address** [*ip-address*]

**Syntax Description**

| *ip-address* | (Optional) Configures the destination IP address that is to respond to the HTTP probe. |
|---|---|

**Defaults**

If the HTTP probe is associated with a firewall farm, you must specify an *ip-address*.

If the HTTP probe is associated with a server farm, and you do not specify an *ip-address*, the address is inherited from the server farm real servers.

**Command Modes**

HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example configures an HTTP probe named *PROBE2*, changes the CLI to IOS SLB HTTP probe submode, and configures the probe to receive responses from IP address *13.13.13.13*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# address 13.13.13.13
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (HTTP probe)** | Configures an HTTP probe name and changes to HTTP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# address (ping probe)

To configure an IP address to which to send ping probes, use the **address** ping probe configuration command. To restore the default settings, use the **no** form of this command.

> **address** [*ip-address*]

> **no address** [*ip-address*]

| Syntax Description | *ip-address* | (Optional) Configures the destination IP address that is to respond to the ping probe. |
| --- | --- | --- |

**Defaults**

If the ping probe is associated with a firewall farm, you must specify an *ip-address*.

If the ping probe is associated with a server farm, and you do not specify an *ip-address*, the address is inherited from the server farm real servers.

**Command Modes**

Ping probe configuration

**Command History**

| Release | Modification |
| --- | --- |
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example configures a ping probe named *PROBE1*, changes the CLI to IOS SLB ping probe submode, and configures the probe to receive responses from IP address *13.13.13.13*:

```
Router(config)# ip slb probe PROBE1 ping
Router(config-slb-probe)# address 13.13.13.13
```

**Related Commands**

| Command | Description |
| --- | --- |
| **ip slb probe (ping probe)** | Configures a ping probe name and changes to ping probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# address (WSP probe)

To configure an IP address to which to send WSP probes, use the **address** WSP probe configuration command. To restore the default settings, use the **no** form of this command.

**address** [*ip-address*]

**no address** [*ip-address*]

**Syntax Description**

| | |
|---|---|
| *ip-address* | (Optional) Configures the destination IP address that is to respond to the WSP probe. |

**Defaults**

If the WSP probe is associated with a firewall farm, you must specify an *ip-address*.

If the WSP probe is associated with a server farm, and you do not specify an *ip-address*, the address is inherited from the server farm real servers.

In dispatched mode, *ip-address* is the same as the virtual server IP address. In directed (NAT) mode, *ip-address* is unnecessary.

**Command Modes**

WSP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(5a)E | This command was introduced. |

**Examples**

The following example configures a ping probe named *PROBE3*, changes the CLI to IOS SLB WSP probe submode, and configures the probe to receive responses from IP address *13.13.13.13*:

```
Router(config)# ip slb probe PROBE3 wsp
Router(config-slb-probe)# address 13.13.13.13
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (WSP probe)** | Configures a WSP probe name and changes to WSP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# advertise

To control the installation of a static route to the Null0 interface for a virtual server address, use the **advertise** virtual server configuration command. To prevent the installation of a static route for the virtual server IP address, use the **no** form of this command.

> **advertise** [**active**]

> **no advertise** [**active**]

**Syntax Description**

| | |
|---|---|
| **active** | (Optional) Indicates that the host route is to be advertised only when the virtual IP address is available (that is, when there is at least one real server in OPERATIONAL, DFP_THROTTLED, or MAXCONNS state). |

**Defaults**

The virtual server IP address is *advertised*. That is, a static route to the Null0 interface is installed for the virtual server IP addresses and it is added to the routing table.

If you do not specify **active**, the host route is advertised regardless of whether the virtual IP address is available.

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(7)E | The **active** keyword was added. |

**Usage Guidelines**

Advertisement of a static route using the routing protocol requires that you configure redistribution of static routes for the routing protocol.

The **advertise** command does not affect virtual servers used for transparent webcache load balancing.

HTTP probes and route health injection require a route to the virtual server. The route is not used, but it must exist in order for HTTP probes and route health injection to function correctly.

- For HTTP probes, the route can be either a host route (advertised by the virtual server) or a default route (specified using the **ip route 0.0.0.0 0.0.0.0** command, for example). If you specify either **no advertise** or **advertise active**, you must specify a default route.

- For route health injection, the route must be a default route.

HTTP probes and route health injection can both use the same default route; you do not need to specify two unique default routes.

**Examples**     The following example prevents advertisement of the virtual server's IP address in routing protocol updates:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# no advertise
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

# agent

To identify a DFP agent with which IOS SLB can initiate connections, use the **agent** DFP configuration command. To remove a DFP agent definition from the DFP configuration, use the **no** form of this command.

**agent** *ip-address port-number* [*timeout* [*retry_count* [*retry_interval*]]]

**no agent** *ip-address port*

| Syntax Description | | |
|---|---|---|
| *ip-address* | Agent IP address | |
| *port-number* | Agent TCP or UDP port number | |
| *timeout* | (Optional) Time period, in seconds, during which the DFP manager must receive an update from the DFP agent. The valid range is 0 to 65535 seconds. The default is 0 seconds, which means there is no timeout. | |
| *retry_count* | (Optional) Number of times the DFP manager attempts to establish the TCP connection to the DFP agent. The valid range is 0 to 65535 times. The default is 0 retries, which means there are infinite retries. | |
| *retry_interval* | (Optional) Interval, in seconds, between retries. The valid range is 1 to 65535 seconds. The default is 180 seconds. | |

**Defaults**

Timeout default: 0 seconds (no timeout)

Retry count default: 0 (infinite retries)

Retry interval default: 180 seconds

**Command Modes**

DFP configuration

| Command History | Release | Modification |
|---|---|---|
| | 12.0(7)XE | This command was introduced. |

**Usage Guidelines**

The password specified on the **ip slb dfp** command in the DFP manager must match the password specified on the **password** command in the DFP agent.

You can identify up to 1024 DFP agents.

**Examples**

The following example sets the DFP password to *Cookies* (to match the DFP agent's password), sets the timeout to *360* seconds, changes the CLI to DFP configuration mode, enables IOS SLB to connect to the DFP agent with IP address *10.1.1.1* and port number *2221*, and sets a timeout of *30* seconds, an infinite number of retries, and a retry interval of *10* seconds:

```
Router(config)# ip slb dfp password Cookies 360
Router(config-slb-dfp)# agent 10.1.1.1 2221 30 0 10
```

| Related Commands | Command | Description |
|---|---|---|
| | **ip dfp agent** | Identifies a DFP agent subsystem and enters DFP agent configuration mode. |
| | **ip slb dfp** | Configures DFP, supplies an optional password, and enters DFP configuration mode. |

# bindid

To configure a bind ID, use the **bindid** server farm configuration command. To remove a bind ID from the server farm configuration, use the **no** form of this command.

> **bindid** [*bind_id*]

> **no bindid** [*bind_id*]

**Syntax Description**

| | |
|---|---|
| *bind_id* | (Optional) Bind ID number. The default bind ID is 0. |

**Defaults**          The default bind ID is 0.

**Command Modes**          Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**          You can configure one bind ID on each **bindid** command.

The bind ID allows a single physical server to be bound to multiple virtual servers and report a different weight for each one. Thus, the single real server is represented as multiple instances of itself, each having a different bind ID. DFP uses the bind ID to identify for which instance of the real server a given weight is specified.

In GPRS load balancing, bind IDs are not supported, therefore do not use the **bindid** command in a GPRS load-balancing environment.

**Examples**          The following example configures bind ID 309:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# bindid 309
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb dfp** | Configures DFP, supplies an optional password, and enters DFP configuration mode. |
| **show ip slb serverfarms** | Displays information about the IOS SLB server farms. |

# clear ip slb

To clear IP IOS SLB connections or counters, use the **clear ip slb** command.

**clear ip slb** {**connections** [**serverfarm** *farm_name* | **vserver** *server_name*] | **counters**}

**Syntax Description**

| | |
|---|---|
| **connections** | Clears the IOS SLB connection database. |
| **serverfarm** | (Optional) Clears the connection database for the specified server farm. |
| *farm_name* | (Optional) Character string used to identify the serverfarm. |
| **vserver** | (Optional) Clears the connection database for the specified virtual server. |
| *server_name* | (Optional) Character string used to identify the virtual server. |
| **counters** | Clears the IOS SLB counters. |

**Defaults**      No default behavior or values.

**Command Modes**      Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.1(1)E | This command was introduced. |

**Usage Guidelines**      In GPRS load balancing, the **clear ip slb connections** command clears connections, not sessions.

**Examples**      The following example clears the connection database of server farm FARM1:

```
Router# clear ip slb connections serverfarm FARM1
```

The following example clears the connection database of virtual server VSERVER1:

```
Router# clear ip slb connections vserver VSERVER1
```

The following example clears the IOS SLB counters:

```
Router# clear ip slb counters
```

**Related Commands**

| Command | Description |
| --- | --- |
| **show ip slb conns** | Displays information about active IOS SLB connections. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb serverfarms** | Displays information about the IOS SLB server farms. |
| **show ip slb stats** | Displays IOS SLB statistics. |
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

# client (virtual server)

To define which clients are allowed to use the virtual server, use the **client** virtual server configuration command. You can use more than one client command to define more than one client. To remove a client definition from the IOS SLB configuration, use the **no** form of this command.

**client** *ip-address network-mask* **[exclude]**

**no client** *ip-address network-mask*

**Syntax Description**

| | |
|---|---|
| *ip-address* | Client IP address. The default is 0.0.0.0 (all clients). |
| *network-mask* | Client IP network mask. The default is 0.0.0.0 (all subnets). |
| **exclude** | (Optional) Excludes connections initiated by the client IP address from the load-balancing scheme. |

**Defaults**

The default client IP address is 0.0.0.0 (all clients).

The default client IP network mask is 0.0.0.0 (all subnets).

Taken together, the default is client 0.0.0.0 0.0.0.0 (allows all clients on all subnets to use the virtual server).

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(1)E | The **exclude** keyword was added. |

**Usage Guidelines**

The *network-mask* value is applied to the source IP address of incoming connections. The result must match the *ip-address* value for the client to be allowed to use the virtual server.

If you configure probes in your network, you must also do one of the following:

- Configure the **exclude** keyword on the **client** command on the virtual server, to exclude connections initiated by the client IP address from the load-balancing scheme.

- Configure IP addresses on the IOS SLB device that are Layer 3-adjacent to the real servers used by the virtual server.

**Examples**

The following example allows only clients from 10.4.4.x access to the virtual server:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# client 10.4.4.0 255.255.255.0
```

| Related Commands | Command | Description |
|---|---|---|
| | **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |
| | **virtual (virtual server)** | Configures the virtual server attributes. |

# credentials

To configure basic authentication values for the HTTP IOS SLB probe, use the **credentials** HTTP probe configuration command. To remove a **credentials** configuration, use the **no** form of this command.

**credentials** *username* [*password*]

**no credentials** *username* [*password*]

**Syntax Description**

| | |
|---|---|
| *username* | Configures the authentication username of the HTTP probe header. The character string is limited to 15 characters. |
| *password* | (Optional) Configures the authentication password of the HTTP probe header. The character string is limited to 15 characters. |

**Defaults**    No default behavior or values.

**Command Modes**    HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Examples**    The following example configures an HTTP probe named *PROBE2*, changes the CLI to IOS SLB HTTP probe submode, sets the HTTP authentication to username *lauren*, and sets the password to *develop*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# credentials lauren develop
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# debug gprs dfp

To display debug messages for GPRS DFP weight calculation, use the **debug gprs dfp** EXEC command. To stop debug output, use the **no** form of this command.

> **debug gprs dfp**

> **no debug gprs dfp**

**Syntax Description**    This command has no arguments or keywords.

**Defaults**    No default behavior or values.

**Command Modes**    EXEC configuration

**Command History**

| Release | Modification |
|---------|--------------|
| 12.1(9)E | This command was introduced. |

**Usage Guidelines**    See the following caution before using **debug** commands:

⚠

**Caution**    Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network flows and fewer users. Debugging during these periods reduces the effect these commands have on other users on the system.

This command displays debug messages for GPRS DFP weight calculation. To display debug messages for the DFP agent subsystem, use the **debug ip dfp agent** command.

**Examples**    The following example configures a debug session to check all GPRS DFP weight calculation:

```
Router# debug gprs dfp
GPRS DFP debugging is on
Router#
```

The following example stops all debugging:

```
Router# no debug all
All possible debugging has been turned off
Router#
```

# debug ip slb

To display debug messages for IOS SLB, use the **debug ip slb** EXEC command. To stop debug output, use the **no** form of this command.

> **debug ip slb** {**conns** | **dfp** | **firewallfarm** | **icmp** | **natpool** | **probe** | **reals** | **replication** | **sessions** | **vservers** | **all**}

> **no debug ip slb** {**conns** | **dfp** | **firewallfarm** | **icmp** | **natpool** | **probe** | **reals** | **replication** | **sessions** | **vservers** | **all**}

**Syntax Description**

| | |
|---|---|
| **all** | Displays all debug messages for IOS SLB. |
| **conns** | Displays debug messages for all connections being handled by IOS SLB, including WSP events and states. |
| **dfp** | Displays debug messages for DFP.<br><br>• To display debug messages for the DFP agent subsystem, use the **debug ip dfp agent** command.<br><br>• To display debug messages for the GPRS DFP weight calculation, use the **debug gprs dfp** command. |
| **firewallfarm** | Displays debug messages related to firewall load balancing. |
| **icmp** | Displays all Internet Control Message Protocol debug messages for IOS SLB. |
| **natpool** | Displays debug messages related to the IOS SLB client NAT pool. |
| **probe** | Displays debug messages related to probes. |
| **reals** | Displays debug messages for all real servers defined to IOS SLB. |
| **replication** | Displays debug messages related to IOS SLB stateful backup virtual server. |
| **sessions** | Displays debug messages for all sessions being handled by IOS SLB. |
| **vservers** | Displays debug messages for all virtual servers defined to IOS SLB. |

**Defaults**

No default behavior or values.

**Command Modes**

EXEC configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(2)E | The **natpool** and **replication** keywords were added. |
| 12.1(3a)E | The **firewallfarm** keyword was added. |
| 12.1(7)E | The **vservers** keyword was added. |
| 12.1(9)E | The **sessions** keyword was added. |

**Usage Guidelines**    See the following caution before using **debug** commands:

⚠️

**Caution**    Because debugging output is assigned high priority in the CPU process, it can render the system unusable. For this reason, use **debug** commands only to troubleshoot specific problems or during troubleshooting sessions with Cisco technical support staff. Moreover, it is best to use **debug** commands during periods of lower network flows and fewer users. Debugging during these periods reduces the effect these commands have on other users on the system.

**Examples**    The following example configures a debug session to check all IP IOS SLB parameters:

```
Router# debug ip slb all
SLB All debugging is on
Router#
```

The following example stops all debugging:

```
Router# no debug all
All possible debugging has been turned off
Router#
```

The following example configures debugging to check IP IOS SLB replication used with stateful backup and displays the output from the send or transmit virtual server:

```
Router# debug ip slb replication
*Mar  2 08:02:38.019:  SLB Replicate: (send) update vs: VS1 update_count 42
```

# delay (firewall farm TCP protocol)

To change the amount of time IOS SLB maintains TCP connection context after a connection has terminated, use the **delay** firewall farm TCP protocol configuration command. To restore the default delay timer, use the **no** form of this command.

**delay** *duration*

**no delay**

**Syntax Description**

| | |
|---|---|
| *duration* | Delay timer duration in seconds. The valid range is 1 to 600 seconds. The default value is 10 seconds. |

**Defaults**

Duration default: 10 seconds

**Command Modes**

Firewall farm TCP protocol configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**

The delay timer allows out-of-sequence packets and final acknowledgments (ACKs) to be delivered after a TCP connection ends.

Do not set this value to zero (0).

If you are configuring a delay timer for HTTP flows, choose a low number such as 5 seconds as a starting point.

**Examples**

The following example specifies that IOS SLB maintains TCP connection context for 30 seconds after a connection has terminated:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# tcp
Router(config-slb-fw-tcp)# delay 30
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **tcp** | Enters TCP protocol configuration mode. |

# delay (virtual server)

To change the amount of time IOS SLB maintains TCP connection context after a connection has terminated, use the **delay** virtual server configuration command. To restore the default delay timer, use the **no** form of this command.

**delay** *duration*

**no delay**

**Syntax Description**

| | |
|---|---|
| *duration* | Delay timer duration in seconds. The valid range is 1 to 600 seconds. The default value is 10 seconds. |

**Defaults**

Duration default: 10 seconds

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**

The delay timer allows out-of-sequence packets and final acknowledgments (ACKs) to be delivered after a TCP connection ends.

Do not set this value to zero (0).

If you are configuring a delay timer for HTTP flows, choose a low number such as 5 seconds as a starting point.

**Examples**

The following example specifies that IOS SLB maintains TCP connection context for 30 seconds after a connection has terminated:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# delay 30
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# expect

To configure a status code or regular expression to expect from the HTTP probe, use the **expect** HTTP probe configuration command. To restore the default settings, use the **no** form of this command.

**expect** [**status** *status-code*] [**regex** *regular-expression*]

**no expect** [**status** *status-code*] [**regex** *regular-expression*]

**Syntax Description**

| status | (Optional) Specifies that a status code is to be expected. |
|---|---|
| *status-code* | (Optional) Configures the expected HTTP status code. The valid range is 100 to 599. The default expected status code is 200. |
| regex | (Optional) Specifies that a regular expression is to be expected. |
| *regular-expression* | (Optional) Configures the regular expression expected in the HTTP response. |

**Defaults**

The default expected status code is 200.

There is no default expected regular expression.

**Command Modes**

HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |
| 12.1(3a)E | The **regex** keyword and *regular-expression* variable were added. |

**Usage Guidelines**

The **expect** command configures the expected status code or regular expression to be received from the servers. A real server is considered to have failed and is taken out of service if any of the following events occurs:

- A status number other than the expected one is received.
- The expected regular expression is not received in the first 2920 bytes of probe output. (IOS SLB searches only the first 2920 bytes for the expected status code or regular expression.)
- The server fails to respond.

For IOS SLB firewall load balancing, configure the HTTP probe to expect status code *40l*.

**Examples**

The following example configures an HTTP probe named *PROBE2* changes the CLI to HTTP submode, and configures the HTTP probe to expect the status code *40l* and the regular expression *Copyright*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# expect status 401 regex Copyright
```

**Related Commands**

| Command | Description |
| --- | --- |
| **ip slb probe (HTTP probe)** | Configures an HTTP probe name and changes to HTTP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# failaction (firewall farm)

To reset client TCP connections to failed firewalls in the firewall farm, use the **failaction** firewall farm configuration command.

**failaction purge**

**Syntax Description**

| | |
|---|---|
| **purge** | Resets client TCP connections to failed firewalls in the firewall farm without waiting for the TCP retransmission timeout. |

**Defaults**

No default behavior or values.

**Command Modes**

Firewall farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(9)E | This command was introduced. |

**Usage Guidelines**

There is not a **no** form of this command.

**Examples**

In the following example, IOS SLB resets all client TCP connections to failed firewalls in firewall farm FIRE1:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# failaction purge
```

# failaction (server farm)

To reset client TCP connections to failed real servers in the server farm, use the **failaction** server farm configuration command.

**failaction purge**

| Syntax Description | | |
|---|---|---|
| **purge** | | Resets client TCP connections to failed real servers in the server farm without waiting for the TCP retransmission timeout. |

**Defaults**

No default behavior or values.

**Command Modes**

Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(9)E | This command was introduced. |

**Usage Guidelines**

There is not a **no** form of this command.

**Examples**

In the following example, IOS SLB resets all client TCP connections to failed real servers in server farm PUBLIC:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# failaction purge
```

# faildetect (ping probe)

To specify the conditions that indicate a server failure, use the **faildetect** ping probe configuration command. To restore the default values that indicate a server failure, use the **no** form of this command.

**faildetect** *number-of-pings*

**no faildetect**

**Syntax Description**

| *number-of-pings* | Number of consecutive unacknowledged pings allowed before a real server is considered to have failed. Valid range is 1 to 255. The default is 10 unacknowledged pings. |
|---|---|

**Defaults**

The default value is 10 unacknowledged pings.

**Command Modes**

Ping probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

In the following example the unacknowledged ping threshold is set to 16:

```
Router(config)# ip slb probe PROBE1 ping
Router(config-slb-probe)# faildetect 16
```

**Related Commands**

| Command | Description |
|---|---|
| **real (firewall farm)** | Identifies a firewall by IP address as a member of a firewall farm and enters real server configuration mode. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# faildetect (real server)

To specify the conditions that indicate a real server failure, use the **faildetect** real server configuration command. To restore the default values that indicate a server failure, use the **no** form of this command.

**faildetect numconns** *number-conns* [**numclients** *number-clients*]

**no faildetect**

**Syntax Description**

| | |
|---|---|
| **numconns** | Number of consecutive connection failures allowed before IOS SLB fails the real server. |
| *number-conns* | Connection failure threshold value. The valid range is 1 to 255. The default value is 8. |
| **numclients** | (Optional) Number of unique client IP addresses that can experience connection failures before IOS SLB fails the real server. |
| *number-clients* | (Optional) Client connection failure threshold value. The valid range is 1 to 8. The default value is 2. |
| | If there is only one client in your network (for example, one SGSN in a GPRS load-balancing environment), then you must specify **numclients 1**. |

**Defaults**

If you do not specify the **faildetect** command, the default value of the connection failure threshold is 8.

If you specify the **faildetect** command but do not specify the **numclients** keyword, the default value of the client connection failure threshold is 2.

**Command Modes**

Real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(9)E | This command was modified to support GPRS load balancing. |

**Usage Guidelines**

IOS SLB does not fail the real server until *both* of the following conditions are met:

- There have been *number-conns* consecutive connection failures.
- There have been *number-clients* unique client connection failures.

That is, there can be many consecutive connection failures, but until there have also been *number-clients* unique client connection failures, IOS SLB does not fail the real server.

Similarly, there can be many unique client connection failures, but until there have also been *number-conns* consecutive connection failures, IOS SLB does not fail the real server.

In GPRS load balancing:

- The **numconns** keyword specifies the number of consecutive Create PDP requests allowed before IOS SLB fails the GGSN.

- The **numclients** keyword specifies the number of unique client Create PDP request failures allowed before IOS SLB fails the GGSN.

**Examples**    In the following example, **numconns** is set to 10 and **numclients** is set to 3:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.10.1.1
Router(config-slb-real)# faildetect numconns 10 numclients 3
```

With those settings, IOS SLB will not fail the real server until there have been 10 consecutive connection failures *and* there have been 3 unique client connection failures.

**Related Commands**

| Command | Description |
|---|---|
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# gprs dfp max-weight

To specify the maximum weight sent to a DFP manager by a GGSN acting as a DFP agent, use the **gprs dfp max-weight** global configuration command. To restore the default setting, use the **no** form of this command.

**gprs dfp max-weight** [*max-weight-value*]

**no gprs dfp max-weight** [*max-weight-value*]

**Syntax Description**

| *max-weight-value* | Specifies the maximum weight sent by the GGSN, acting as a DFP agent, to a DFP manager. The valid range is 1 to 100. The default value is 8. |
|---|---|

**Defaults**

The default maximum weight is 8.

**Command Modes**

Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(9)E | This command was introduced. |

**Usage Guidelines**

If you use DFP with GPRS load balancing, you must also specify a maximum number of PDP contexts for each GGSN, using the **gprs maximum-pdp-context-allowed** command. *Do not* accept the default value of 1000 PDP contexts. A value of **45000** is recommended. Significantly lower values, including the default value of 1000, can impact performance in a GPRS load-balancing environment.

**Examples**

The following example sets the maximum weight sent by GGSN to *43*:

```
Router(config)# gprs dfp max-weight 43
```

**Related Commands**

| Command | Description |
|---|---|
| **agent** | Identifies a DFP agent to which IOS SLB can connect. |
| **gprs maximum-pdp-context-allowed** | Specifies the maximum number of PDP contexts (mobile sessions) that can be activated on the GGSN. |
| **ip dfp agent** | Identifies a DFP agent subsystem and enters DFP agent configuration mode. |
| **ip slb dfp** | Configures DFP, supplies an optional password, and enters DFP configuration mode. |

# gprs slb cef

To identify the IP address of the GGSN virtual server to CEF, use the **gprs slb cef** global configuration command. To remove the IP address identification, use the **no** form of this command.

**gprs slb cef** *virtual-server-address*

**no gprs slb cef** *virtual-server-address*

**Syntax Description**

| *virtual-server-address* | IP address of the GGSN virtual server instance used by clients to connect to the server farm. (This virtual IP address is also a loopback address on the GGSN.) |
|---|---|

**Defaults**   No default behavior or values.

**Command Modes**   Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(9)E | This command was introduced. |

**Usage Guidelines**   This command is required if the GGSN is using CEF switching. If the GGSN is *not* using CEF switching, do not use this command.

**Examples**   The following example identifies the IP address of the GGSN virtual server, *10.0.0.13*, to CEF:

```
Router(config)# gprs slb cef 10.0.0.13
```

**Related Commands**

| Command | Description |
|---|---|
| **interface loopback** | Creates a loopback interface. |
| **ip cef** | Enables CEF on the RP card. |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# header

To configure the basic authentication values for the HTTP probe, use the **header** HTTP probe configuration command. To remove a **header** HTTP probe configuration, use the **no** form of this command.

> **header** *field-name* [*field-value*]

> **no header** *field-name* [*field-value*]

**Syntax Description**

| | |
|---|---|
| *field-name* | Configures the name of the HTTP probe header. The character string is limited to 15 characters. |
| *field-value* | (Optional) Configures the value of the HTTP probe header. |

**Defaults**

No default behavior or values, although the following headers are inserted in the request by default:

```
Accept: */*
Connection: close
User-Agent: cisco-slb-probe/1.0
Host: virtual IP address
```

**Command Modes**

HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Usage Guidelines**

The **header** HTTP probe configuration command configures the name and value parameters of the header.

**Note**  The colon ( : ) separating the field-name and field-value is automatically inserted if not provided. Multiple headers with the same name are not supported.

**Examples**

The following example configures an HTTP probe named *PROBE2*, changes the CLI to HTTP submode, and configures the HTTP probe header name as *Cookie* and value as *Monster*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# header Cookie Monster
```

**Related Commands**

| Command | Description |
| --- | --- |
| **ip slb probe (HTTP probe)** | Configures an HTTP probe name and changes to HTTP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# idle (firewall farm TCP protocol)

To specify the minimum amount of time IOS SLB maintains connection information in the absence of packet activity, use the **idle** firewall farm TCP protocol configuration command. To restore the default idle duration value, use the **no** form of this command.

**idle** *duration*

**no idle**

**Syntax Description**

| | |
|---|---|
| *duration* | Idle connection timer duration in seconds. Valid values range from 10 to 65535. The default is 3600 seconds (1 hour). |

**Defaults**

Duration default: 3600 seconds

**Command Modes**

Firewall farm TCP protocol configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**

TCP connections that do not send flows or keepalives before the idle timer expires are assumed to be inactive and are reset (RST).

If you are configuring an idle timer for HTTP flows, choose a low number such as 120 seconds as a starting point. A low number ensures that the IOS SLB connection database maintains a manageable size if problems at the server, client, or network result in a large number of connections. However, do not choose a value under 60 seconds; such a low value can reduce the efficiency of IOS SLB.

**Examples**

The following example instructs IOS SLB to maintain connection information for an idle connection for 120 seconds.

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# tcp
Router(config-slb-fw-tcp)# idle 120
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **tcp** | Enters TCP protocol configuration mode. |

# idle (firewall farm UDP protocol)

To specify the minimum amount of time IOS SLB maintains connection information in the absence of packet activity, use the **idle** firewall farm UDP protocol configuration command. To restore the default idle duration value, use the **no** form of this command.

**idle** *duration*

**no idle**

**Syntax Description**

| *duration* | Idle connection timer duration in seconds. Valid values range from 10 to 65535. The default is 3600 seconds (1 hour). |
|---|---|

**Defaults**

Duration default: 3600 seconds

**Command Modes**

Firewall farm UDP protocol configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example instructs IOS SLB to maintain connection information for an idle connection for 120 seconds.

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# udp
Router(config-slb-fw-udp)# idle 120
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **udp** | Enters UDP protocol configuration mode. |

# idle (virtual server)

To specify the minimum amount of time IOS SLB maintains connection information in the absence of packet activity, use the **idle** virtual server configuration command. To restore the default idle duration value, use the **no** form of this command.

**idle** *duration*

**no idle**

## Syntax Description

| | |
|---|---|
| *duration* | Idle connection timer duration in seconds. Valid values range from 10 to 65535. The default is 30 seconds in GPRS load balancing, or 3600 seconds (1 hour) in all other environments. |

## Defaults

The default idle duration is 30 seconds in GPRS load balancing, or 3600 seconds in all other environments.

## Command Modes

Virtual server configuration

## Command History

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(9)E | This command was modified to support GPRS load balancing. |

## Usage Guidelines

TCP connections that do not send flows or keepalives before the idle timer expires are assumed to be inactive and are reset (RST).

If you are configuring an idle timer for HTTP flows, choose a low number such as 120 seconds as a starting point. A low number ensures that the IOS SLB connection database maintains a manageable size if problems at the server, client, or network result in a large number of connections. However, do not choose a value under 60 seconds (except in GPRS load balancing); such a low value can reduce the efficiency of IOS SLB.

In most environments, the idle timer times out data paths. However, in GPRS load balancing, it times out the session context for signaling paths (not data paths).

In GPRS load balancing, you must specify an idle timer greater than the longest possible interval between PDP context requests on the SGSN. The longest interval can be expressed using the following algorithm:

**Longest interval = T3 x $2^{(N3-2)}$**

where **T3** is the value specified on the **gprs gtp t3-response** *response_interval* command, and **N3** is the value specified on the **gprs gtp n3-requests** *n3-requests* command (or the equivalent commands, if you are not using a Cisco SGSN).

For example, if *response_interval* is set to **3** and *n3-requests* is set to **6**, then:

**Longest interval = 3 x $2^{(6-2)}$ = 3 x $2^{(4)}$ = 3 x 16 = 48 seconds**

Given those values, you must specify an idle timer of at least 49 seconds.

**Examples**

The following example instructs IOS SLB to maintain connection information for an idle connection for 120 seconds.

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# idle 120
```

**Related Commands**

| Command | Description |
|---|---|
| show ip slb vserver | Displays information about the virtual servers defined to IOS SLB. |
| virtual (virtual server) | Configures the virtual server attributes. |

# inservice (firewall farm)

To enable the firewall farm for use by IOS SLB, use the **inservice** firewall farm configuration command. To remove the firewall farm from service, use the **no** form of this command.

**inservice** [**standby** *group-name*]

**no inservice** [**standby** *group-name*]

## Syntax Description

| | |
|---|---|
| **standby** | (Optional) Configures the HSRP standby firewall farm for use with stateless and stateful backup. |
| *group-name* | (Optional) Specifies the HSRP group name with which the IOS SLB firewall farm is associated. |

## Defaults

If the **inservice** command is not specified, the firewall farm is defined to IOS SLB but is not used.

## Command Modes

Firewall farm configuration

## Command History

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

## Usage Guidelines

When you use the **no** form of this command to remove a firewall farm from service, the firewall farm quiesces gracefully. No new connections are assigned, and existing connections are allowed to complete.

## Examples

In the following example, the firewall farm is enabled for use by the IOS SLB feature:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# inservice
```

## Related Commands

| Command | Description |
|---|---|
| **ip slb firewallfarm** | Identifies a firewall by IP address farm and enters firewall farm configuration mode. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |

# inservice (firewall farm real server)

To enable the firewall for use by IOS SLB, use the **inservice** firewall farm real server configuration command. To remove the firewall from service, use the **no** form of this command.

> **inservice**

> **no inservice**

**Syntax Description**    This command has no arguments or keywords.

**Defaults**    If the **inservice** command is not specified, the firewall is defined to IOS SLB but is not used.

**Command Modes**    Firewall farm real server configuration

**Command History**

| Release | Modification |
|---------|--------------|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**    IOS SLB firewall load balancing uses probes to detect failures. Therefore, if you have not configured a probe, the firewall is not placed in service.

When you use the **no** form of this command to remove a firewall from service, the firewall quiesces gracefully. No new connections are assigned, and existing connections are allowed to complete.

**Examples**    In the following example, the firewall is enabled for use by the IOS SLB feature:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# real 10.10.1.1
Router(config-slb-fw-real)# inservice
```

**Related Commands**

| Command | Description |
|---------|-------------|
| **real (firewall farm)** | Identifies a firewall by IP address as a member of a firewall farm and enters real server configuration mode. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# inservice (server farm real server)

To enable the real server for use by IOS SLB, use the **inservice** server farm real server configuration command. To remove the real server from service, use the **no** form of this command.

**inservice**

**no inservice**

**Syntax Description**    This command has no arguments or keywords.

**Defaults**    If the **inservice** command is not specified, the real server is defined to IOS SLB but is not used.

**Command Modes**    Server farm real server configuration

**Command History**

| Release | Modification |
| --- | --- |
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**    When you use the **no** form of this command to remove a real server from service, the real server quiesces gracefully. No new connections are assigned, and existing connections are allowed to complete.

**Examples**    In the following example, the real server is enabled for use by the IOS SLB feature:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.10.1.1
Router(config-slb-sfarm-real)# inservice
```

**Related Commands**

| Command | Description |
| --- | --- |
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# inservice (server farm virtual server)

To enable the virtual server for use by IOS SLB, use the **inservice** server farm virtual server configuration command. To remove the virtual server from service, use the **no** form of this command.

**inservice** [**standby** *group-name*]

**no inservice** [**standby** *group-name*]

**Syntax Description**

| | |
|---|---|
| **standby** | (Optional) Configures the HSRP standby virtual server for use with stateless and stateful backup. |
| *group-name* | (Optional) Specifies the HSRP group name with which the IOS SLB virtual server is associated. |

**Defaults**

If the **inservice** command is not specified, the virtual server is defined to IOS SLB but is not used.

**Command Modes**

Server farm virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(1)E | The **standby** keyword and *group-name* variable were added. |

**Usage Guidelines**

When you use the **no** form of this command to remove a virtual server from service, the virtual server quiesces gracefully. No new connections are assigned, and existing connections are allowed to complete.

**Examples**

In the following example, the virtual server is enabled for use by the IOS SLB feature:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# inservice
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# interval (HTTP probe)

To configure an HTTP probe interval, use the **interval** HTTP probe configuration command. To remove an HTTP probe **interval** configuration, use the **no** form of this command.

**interval** *seconds*

**no interval** *seconds*

| | |
|---|---|
| **Syntax Description** | |

| | |
|---|---|
| *seconds* | Designates the number of seconds to wait before reattempting the probe. Valid values range from 1–65535 seconds. The default interval is 8 seconds. |

**Defaults**  The default **interval** value is 8 seconds.

**Command Modes**  HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Examples**  The following example configures an HTTP probe named *PROBE2*, changes the CLI to HTTP submode, and configures the HTTP probe timer interval to transmit every *11* seconds:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# interval 11
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (HTTP probe)** | Configures an HTTP probe name and changes to HTTP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# interval (ping probe)

To configure a ping probe interval, use the **interval** ping probe configuration command. To remove a ping probe **interval** configuration, use the **no** form of this command.

**interval** *seconds*

**no interval** *seconds*

| | |
|---|---|
| **Syntax Description** | *seconds*      Designates the number of seconds to wait before reattempting the probe. Valid values range from 1–65535 seconds. The default interval is 1 second. |

**Defaults**  The default **interval** value is 1 second.

**Command Modes**  Ping probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**  The following example configures a ping probe named *PROBE1*, changes the CLI to ping submode, and configures the ping probe timer interval to transmit every *11* seconds:

```
Router(config)# ip slb probe PROBE1 ping
Router(config-slb-probe)# interval 11
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (ping probe)** | Configures a ping probe name and changes to ping probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# interval (WSP probe)

To configure a WSP probe interval, use the **interval** WSP probe configuration command. To remove a WSP probe **interval** configuration, use the **no** form of this command.

**interval** *seconds*

**no interval** *seconds*

| Syntax Description | *seconds* | Designates the number of seconds to wait before reattempting the probe. Valid values range from 1–65535 seconds. The default interval is 8 seconds. |
|---|---|---|

**Defaults**  The default **interval** value is 8 seconds.

**Command Modes**  WSP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(5a)E | This command was introduced. |

**Examples**  The following example configures a ping probe named *PROBE3*, changes the CLI to WSP probe submode, and configures the WSP probe timer interval to transmit every *11* seconds:

```
Router(config)# ip slb probe PROBE3 wsp
Router(config-slb-probe)# interval 11
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (WSP probe)** | Configures a WSP probe name and changes to WSP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# ip slb dfp

To configure DFP, supply an optional password, and initiate DFP configuration mode, use the **ip slb dfp** global configuration command. To remove the DFP configuration, use the **no** form of this command.

> **ip slb dfp** [**password** [**0** | **7**] *password* [*timeout*]]

> **no ip slb dfp**

**Syntax Description**

| password | (Optional) Specifies a password for MD5 authentication. |
|---|---|
| **0** | (Optional) Specifies that the password is unencrypted. This is the default setting. |
| **7** | (Optional) Specifies that the password is encrypted. |
| *password* | (Optional) Password value for MD5 authentication. This password must match the password configured on the host agent. |
| *timeout* | (Optional) Delay period, in seconds, during which both the old password and the new password are accepted. The valid range is 0 to 65535 seconds. The default value is 180 seconds, if a password is specified. |

**Defaults**

The password encryption default is 0 (unencrypted).

The password timeout default is 180 seconds, if a password is specified.

**Command Modes**

Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(3a)E | The **0** and **7** keywords were added. |

**Usage Guidelines**

The password specified on the **ip slb dfp** command in the DFP manager must match the password specified on the **password** command in the DFP agent.

The timeout option allows you to change the password without stopping messages between the DFP agent and its manager. The default value is 180 seconds.

During the timeout, the agent sends packets with the old password (or null, if there is no old password), and receives packets with either the old or new password. After the timeout expires, the agent sends and receives packets only with the new password; received packets that use the old password are discarded.

If you are changing the password for an entire load-balanced environment, set a longer timeout. This allows enough time for you to update the password on all agents and servers before the timeout expires. It also prevents mismatches between agents and servers that have begun running the new password and agents, and servers on which you have not yet changed the old password.

**Examples**

The following example configures DFP, sets the DFP password to *Cookies* and the timeout to *360* seconds, and changes the CLI to DFP configuration mode:

```
Router(config)# ip slb dfp password Cookies 360
Router(config-slb-dfp)#
```

**Related Commands**

| Command | Description |
| --- | --- |
| **agent** | Identifies a DFP agent to which IOS SLB can connect. |
| **ip dfp agent** | Identifies a DFP agent subsystem and enters DFP agent configuration mode. |

# ip slb entries

To configure an initial allocation and a maximum value for IOS SLB database entries, use the **ip slb entries** global configuration command. To restore the default values, use the **no** form of this command.

**ip slb entries** [**conn** [*init-conn* [*max-conn*]] | **frag** [*init-frag* [*max-frag*]] | **sticky** [*init-sticky* [*max-sticky*]]]

**no ip slb entries** [**conn** | **frag** | **sticky**]

**Syntax Description**

| | |
|---|---|
| **conn** | (Optional) Configures an initial allocation and a maximum value for IOS SLB connection database entries. |
| *init-conn* | (Optional) Initial allocation of connection database entries. The number of connection database entries can grow dynamically: When the number of available connection database entries is less than half of *init-conn*, IOS SLB allocates additional connection database entries. |
| | Valid range is 1 to 1000000 connection database entries. The default is 8000 connection database entries. |
| *max-conn* | (Optional) Maximum number of connection database entries that can be allocated. Valid range is 1 to 8000000 connection database entries. The default is 8000000 connection database entries. |
| **frag** | (Optional) Configures an initial allocation and a maximum value for IOS SLB fragment database entries. |
| *init-frag* | (Optional) Initial allocation of fragment database entries. The number of fragment database entries can grow dynamically: When the number of available fragment database entries is less than half of *init-frag*, IOS SLB allocates additional fragment database entries. |
| | Valid range is 1 to 1000000 fragment database entries. The default is 2000 fragment database entries. |
| *max-frag* | (Optional) Maximum number of fragment database entries that can be allocated. Valid range is 1 to 8000000 fragment database entries. The default is 32000 fragment database entries. |
| **sticky** | (Optional) Configures an initial allocation and a maximum value for IOS SLB sticky connection database entries. |
| *init-sticky* | (Optional) Initial allocation of sticky database entries. The number of sticky database entries can grow dynamically: When the number of available sticky database entries is less than half of *init-sticky*, IOS SLB allocates additional sticky database entries. |
| | Valid range is 1 to 1000000 sticky database entries. The default is 4000 sticky database entries. |
| *max-sticky* | (Optional) Maximum number of sticky database entries that can be allocated. Valid range is 1 to 8000000 sticky database entries. The default is 8000000 sticky database entries. |

**Defaults**                For connections, the default initial allocation is 8000 connections, and the default maximum is 8000000 connections.

For fragments, the default initial allocation is 4000 fragments, and the default maximum is 8000000 fragments.

For sticky connections, the default initial allocation is 2000 sticky connections, and the default maximum is 3200 sticky connections.

**Command Modes**           Global configuration

**Command History**

| Release | Modification |
|---------|--------------|
| 12.1(2)E | This command was introduced. |

**Usage Guidelines**        If you configure an initial allocation value that exceeds the amount of available memory, memory might not be available for other features. In extreme cases, the router or switch might not boot properly. Therefore, be careful when you configure initial allocation values.

**Examples**                The following example configures an initial allocation of 128,000 connections, which can grow dynamically to a limit of 512,000 connections:

```
Router(config)# ip slb entries conn 128000 512000
```

**Related Commands**

| Command | Description |
|---------|-------------|
| **show ip slb conns** | Displays all connections handled by IOS SLB, or, optionally, only those connections associated with a particular virtual server or client. |

# ip slb firewallfarm

To identify a firewall farm and initiate firewall farm configuration mode, use the **ip slb firewallfarm** global configuration command. To remove the firewall farm from the IOS SLB configuration, use the **no** form of this command.

**ip slb firewallfarm** *firewallfarm-name*

**no ip slb firewallfarm** *firewallfarm-name*

| Syntax Description | | |
|---|---|---|
| *firewallfarm-name* | Character string used to identify the firewall farm. The character string is limited to 15 characters. | |

**Defaults**    No default behavior or values.

**Command Modes**    Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**    Grouping real servers into firewall farms is an essential part of IOS SLB firewall load balancing. Using firewall farms enables IOS SLB to assign new connections to the real servers based on their weighted capacities, and on the load-balancing algorithms used.

**Examples**    The following example identifies a firewall farm named FIRE1:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)#?
```

**Related Commands**

| Command | Description |
|---|---|
| **real (firewall farm)** | Identifies a firewall by IP address as a member of a firewall farm and enters real server configuration mode. |

# ip slb natpool

To configure an IOS SLB NAT, use the **ip slb natpool** configuration command to create at least one client address pool. To remove an **ip slb natpool** configuration, use the **no** form of this command.

**ip slb natpool** *pool-name start-ip end-ip* [**netmask** *netmask* | **prefix-length** *leading_1_bits*] [**entries** *init-addr* [*max-addr*]]

**no ip slb natpool** *pool-name*

## Syntax Description

| | |
|---|---|
| *pool-name* | Configures a character string used to identify this client address pool. The character string is limited to 15 characters. |
| *start-ip* | Configures a starting IP address that defines the range of addresses in the address pool. |
| *end-ip* | Configures an ending IP address that defines the range of addresses in the address pool. |
| **netmask** | (Optional) Keyword used to configure the subnet mask. |
| *netmask* | (Optional) Mask for the associated IP subnet. |
| **prefix-length** | (Optional) Keyword used to configure the subnet mask. |
| *leading_1_bits* | (Optional) Mask for the associated IP subnet. |
| **entries** | (Optional) Configures an initial allocation and a maximum value for IOS SLB client NAT address entries for *pool-name*. |
| *init-addr* | (Optional) Initial allocation of client NAT address entries. The number of client NAT address entries can grow dynamically: When the number of available client NAT address entries is less than half of *init-addr*, IOS SLB allocates additional client NAT address entries. <br><br> Valid range is 1 to 1000000 client NAT address entries. The default is 8000 client NAT address entries. |
| *max-addr* | (Optional) Maximum number of client NAT address entries that can be allocated. Valid range is 1 to 8000000 client NAT address entries. <br><br> The default is the maximum number of ports that can be allocated within the IP address range specified for *pool-name*. For example, the following command: <br><br> **ip slb natpool 3.3.3.1 3.3.3.5 prefix-length 24 entries 8000** <br><br> has a default *max-addr* of (3.3.3.1-3.3.3.5)*54535, or 4*54535, or 218140. |

## Defaults

The default initial allocation is 8000 client NAT address entries.

The default maximum number of client NAT address entries that can be allocated is the maximum number of ports that can be allocated within the IP address range.

## Command Modes

Global configuration

| Command History | Release | Modification |
|---|---|---|
| | 12.1(2)E | This command was introduced. |

**Usage Guidelines**   If you want to use client NAT, you must create at least one client address pool.

**Examples**   The following example configures an IOS SLB NAT server farm pool of addresses with the name web-clients, the IP address range from 128.3.0.1 through 128.3.0.254, and a subnet mask of 255.255.0.0:

```
Router(config)# ip slb natpool web-clients 128.3.0.1 128.3.0.254 netmask
255.255.0.0
```

| Related Commands | Command | Description |
|---|---|---|
| | **show ip slb natpool** | Displays information about the IOS SLB NAT configuration. |
| | **show ip slb serverfarms** | Displays information about the server farm configuration. |

# ip slb probe (HTTP probe)

To configure an HTTP probe name and to change to HTTP probe configuration submode, use the **ip slb probe (HTTP probe)** configuration command. To remove an **ip slb probe** configuration, use the **no** form of this command.

**ip slb probe** *name* **http**

**no ip slb probe** *name*

**Syntax Description**

| | |
|---|---|
| *name* | Configures a name for the HTTP probe. The character string is limited to 15 characters. |

**Defaults**        No default behavior or values.

**Command Modes**        Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Usage Guidelines**        This command configures the HTTP probe name and application protocol, and changes the user interface to HTTP submode.

The HTTP probe cannot be unconfigured while it is being used by the server farm or firewall farm.

You can configure more than one probe, in any combination of types (HTTP, ping, or WSP), for each server farm, or for each firewall in a firewall farm.

**Note**        HTTP probes require a route to the virtual server. The route is not used, but it must exist in order for HTTP probes to function correctly. The route can be either a host route (advertised by the virtual server) or a default route (specified using the **ip route 0.0.0.0 0.0.0.0** command, for example).

**Examples**        The following example configures an IOS SLB probe named *PROBE2*, then changes to HTTP probe configuration submode:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)#
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# ip slb probe (ping probe)

To configure a ping probe name and to change to ping probe configuration submode, use the **ip slb probe (ping probe)** configuration command. To remove an **ip slb probe** configuration, use the **no** form of this command.

**ip slb probe** *name* **ping**

**no ip slb probe** *name*

**Syntax Description**

| | |
|---|---|
| *name* | Configures a name for the ping probe. The character string is limited to 15 characters. |

**Defaults**    No default behavior or values.

**Command Modes**    Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**    This command configures the ping probe name and application protocol, and changes the user interface to ping submode.

The ping probe cannot be unconfigured while it is being used by the server farm or firewall farm.

You can configure more than one probe, in any combination of types (HTTP, ping, or WSP), for each server farm, or for each firewall in a firewall farm.

**Examples**    The following example configures an IOS SLB probe named *PROBE1*, then changes to ping probe configuration submode:

```
Router(config)# ip slb probe PROBE1 ping
Router(config-slb-probe)#
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# ip slb probe (WSP probe)

To configure a WSP probe name and to change to WSP probe configuration submode, use the
**ip slb probe (WSP probe)** configuration command. To remove an **ip slb probe** configuration, use the
**no** form of this command.

> **ip slb probe** *name* **wsp**

> **no ip slb probe** *name*

**Syntax Description**

| | |
|---|---|
| *name* | Configures a name for the WSP probe. The character string is limited to 15 characters. |

**Defaults**

No default behavior or values.

**Command Modes**

Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(5a)E | This command was introduced. |

**Usage Guidelines**

This command configures the WSP probe name and application protocol, and changes the user interface to WSP probe configuration submode.

The WSP probe cannot be unconfigured while it is being used by the server farm or firewall farm.

You can configure more than one probe, in any combination of types (HTTP, ping, or WSP), for each server farm, or for each firewall in a firewall farm.

**Examples**

The following example configures an IOS SLB probe named *PROBE3*, then changes to WSP probe configuration submode:

```
Router(config)# ip slb probe PROBE3 wsp
Router(config-slb-probe)#
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# ip slb serverfarm

To identify a server farm and initiate server farm configuration mode, use the **ip slb serverfarm** global configuration command. To remove the server farm from the IOS SLB configuration, use the **no** form of this command.

**ip slb serverfarm** *serverfarm-name*

**no ip slb serverfarm** *serverfarm-name*

| Syntax Description | | |
|---|---|
| *serverfarm-name* | Character string used to identify the server farm. The character string is limited to 15 characters. |

**Defaults**  No default behavior or values.

**Command Modes**  Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**  Grouping real servers into server farms is an essential part of IOS SLB. Using server farms enables IOS SLB to assign new connections to the real servers based on their weighted capacities, and on the load-balancing algorithms used.

**Examples**  The following example identifies a server farm named PUBLIC:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)#?
```

**Related Commands**

| Command | Description |
|---|---|
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |

# ip slb vserver

To identify a virtual server and initiate virtual server configuration mode, use the **ip slb vserver** global configuration command. To remove a virtual server from the IOS SLB configuration, use the **no** form of this command.

**ip slb vserver** *virtual_server-name*

**no ip slb vserver** *virtual_server-name*

**Syntax Description**

| | |
|---|---|
| *virtual_server-name* | Character string used to identify the virtual server. The character string is limited to 15 characters. |

**Defaults**  No default behavior or values.

**Command Modes**  Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**  The following example identifies a virtual server named PUBLIC_HTTP:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)#
```

**Related Commands**

| Command | Description |
|---|---|
| **serverfarm** | Associates a real server farm with a virtual server. |
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

# maxconns (firewall farm TCP protocol)

To limit the number of active connections to the real server, use the **maxconns** firewall farm TCP protocol configuration command. To restore the default of no limit, use the **no** form of this command.

**maxconns** *maximum-number*

**no maxconns**

## Syntax Description

| *maximum-number* | Maximum number of simultaneous active TCP connections using the firewall farm. Valid values range from 1 to 4294967295. The default is 4294967295. |
|---|---|

## Defaults

Maximum_number default: 4294967295

## Command Modes

Firewall farm TCP protocol configuration

## Command History

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

## Examples

The following example limits the real server to a maximum of 1000 simultaneous active connections:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# tcp
Router(config-slb-fw-tcp)# maxconns 1000
```

## Related Commands

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |
| **tcp** | Enters TCP protocol configuration mode. |

# maxconns (firewall farm UDP protocol)

To limit the number of active connections to the real server, use the **maxconns** firewall farm UDP protocol configuration command. To restore the default of no limit, use the **no** form of this command.

**maxconns** *maximum-number*

**no maxconns**

**Syntax Description**

| | |
|---|---|
| *maximum-number* | Maximum number of simultaneous active UDP connections using the firewall farm. Valid values range from 1 to 4294967295. The default is 4294967295. |

**Defaults**

Maximum_number default: 4294967295

**Command Modes**

Firewall farm UDP protocol configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example limits the real server to a maximum of 1000 simultaneous active connections:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# udp
Router(config-slb-fw-udp)# maxconns 1000
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |
| **udp** | Enters UDP protocol configuration mode. |

# maxconns (server farm)

To limit the number of active connections to the real server, use the **maxconns** real server configuration command. To restore the default of no limit, use the **no** form of this command.

**maxconns** *maximum-number*

**no maxconns**

**Syntax Description**

| | |
|---|---|
| *maximum-number* | Maximum number of simultaneous active connections on the real server. Valid values range from 1 to 4294967295. The default is 4294967295. |

**Defaults**

The default maximum number of simultaneous active connections on the real server is 4294967295.

**Command Modes**

Real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**

In GPRS load balancing, allow this command to default to 4294967295. The impact of this command is minimal because sessions are very short-lived.

**Examples**

The following example limits the real server to a maximum of 1000 simultaneous active connections:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.10.1.1
Router(config-slb-real)# maxconns 1000
```

**Related Commands**

| Command | Description |
|---|---|
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# mls aging slb normal

To configure the aging time for flows, use the **mls aging slb normal** global configuration command. To restore the default setting, use the **no** form of this command.

**mls aging slb normal** *time*

**no mls aging slb normal** *time*

## Syntax Description

| | |
|---|---|
| *time* | Idle time, in milliseconds, before a flow is aged. The valid range is 1 milliseconds to 10000 milliseconds. The default setting is 2000 milliseconds. |
| | **Note** Heavier-than-normal loads can age flows more aggressively than this time. |

## Defaults

The default aging idle time is 2000 milliseconds.

## Command Modes

Global configuration

## Command History

| Release | Modification |
|---|---|
| 12.1(8)E | This command was introduced. |

## Usage Guidelines

This command is supported for Catalyst 6000 Family Switches only.

## Examples

The following example sets the idle time to 4000 milliseconds:

```
Router(config)# mls aging slb normal 4000
```

## Related Commands

| Command | Description |
|---|---|
| **ip slb firewallfarm** | Identifies a firewall farm and initiates firewall farm configuration mode. |
| **ip slb serverfarm** | Associates a real server farm with a virtual server. |
| **ip slb vserver** | Identifies a virtual server. |
| **mls aging slb process** | Controls how often the aging process runs. |

# mls aging slb process

To control how often the aging process runs, use the **mls aging slb process** global configuration command. To restore the default setting, use the **no** form of this command.

**mls aging slb process** *time*

**no mls aging slb process** *time*

**Syntax Description**

| | |
|---|---|
| *time* | Aging process interval, in milliseconds. The valid range is 1 millisecond to 10000 milliseconds. The default setting is 2000 seconds. |

**Defaults**

The default aging process interval is 2000 milliseconds.

**Command Modes**

Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(8)E | This command was introduced. |

**Usage Guidelines**

This command is supported for Catalyst 6000 Family Switches only.

**Examples**

The following example sets the aging process interval to 4000 milliseconds:

```
Router(config)# mls aging slb process 4000
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb firewallfarm** | Identifies a firewall farm and initiates firewall farm configuration mode. |
| **ip slb serverfarm** | Associates a real server farm with a virtual server. |
| **ip slb vserver** | Identifies a virtual server. |
| **mls aging slb normal** | Configures the aging time for flows. |

# mls ip slb search wildcard

To specify the behavior of IOS SLB wildcard searches, use the **mls ip slb search wildcard** global configuration command. To restore the default setting, use the **no** form of this command.

> **mls ip slb search wildcard** [**pfc** | **rp**]

> **no mls ip slb search wildcard** [**pfc** | **rp**]

**Syntax Description**

| | |
|---|---|
| **pfc** | Specifies that IOS SLB wildcard searches are to be performed by the Policy Feature Card (PFC). This is the default setting. |
| **rp** | Specifies that IOS SLB wildcard searches are to be performed by the route processor. |

**Defaults**

The default setting is for the PFC to perform IOS SLB wildcard searches.

**Command Modes**

Global configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(7)E | This command was introduced. |

**Usage Guidelines**

This command is supported for Catalyst 6000 Family Switches only.

If you configure IOS SLB and either input ACLs or firewall load balancing on the same Catalyst 6000 Family Switch, you can exceed the capacity of the TCAM on the PFC. To correct the problem, use the **mls ip slb search wildcard rp** command to reduce the amount of TCAM space used by IOS SLB, but be aware that this command can result in a slight increase in route processor utilization.

**Examples**

The following example limits wildcard searches to the route processor:

```
Router(config)# mls ip slb search wildcard rp
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb firewallfarm** | Identifies a firewall by IP address farm and enters firewall farm configuration mode. |
| **ip slb serverfarm** | Associates a real server farm with a virtual server. |
| **ip slb vserver** | Identifies a virtual server. |

# nat

To configure IOS SLB NAT and specify a NAT mode, use the **nat** server farm configuration command. To remove a NAT configuration, use the **no** form of this command.

**nat** {**client** *pool-name* | **server**}

**no nat** {**client** | **server**}

| Syntax Description | | |
|---|---|---|
| | **client** | Configures the client address in load-balanced packets using addresses from the client address pool. |
| | *pool-name* | Configures the pool name and must match the *pool-name* parameter from a previous **ip slb natpool** command. |
| | **server** | Configures the destination address in load-balanced packets sent to the real server as the address of the real server chosen by the server farm load-balancing algorithm. |

**Defaults**          No default behavior or values.

**Command Modes**     Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(1)E | This command was introduced. |
| 12.1(2)E | The **client** keyword and *pool-name* variable were added. |

**Usage Guidelines**  The **no nat** command is allowed only if the virtual server was removed from service with the **no inservice** command.

In GPRS load balancing, NAT is not supported, therefore the **nat** command is not supported.

**Examples**          The following example changes to IOS SLB server farm configuration mode and configures NAT mode as server address translation on server farm FARM2:

```
Router# ip slb serverfarm FARM2
Router(config-slb-sfarm)# nat server
```

The following example configures the NAT mode on server farm FARM2 to client translation mode and, using the **real (server farm)** command, configures the real server IP address as 10.3.1.1:

```
Router(config-slb-sfarm)# nat client web-clients
Router(config-slb-sfarm)# real 10.3.1.1
```

| Related Commands | Command | Description |
|---|---|---|
| | **ip slb serverfarm** | Associates a real server farm with a virtual server. |
| | **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| | **show ip slb serverfarms** | Displays information about the server farm configuration. |

# port (HTTP probe)

To specify the port to which an HTTP probe is to connect, use the **port (HTTP probe)** HTTP probe configuration command. To restore the default settings, use the **no** form of this command.

**port** *port-number*

**no port** *port-number*

| Syntax Description | *port-number* | Configures the TCP or UDP port number to which the HTTP probe is to connect. |
|---|---|---|

**Defaults**

In dispatched mode, the port number is inherited from the virtual server.

If port translation is configured for the real server, that port number is used. See the **real (server farm)** command for more details.

**Command Modes**

HTTP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example configures an HTTP probe named *PROBE2*, changes the CLI to IOS SLB HTTP probe submode, and configures the probe to connect to port number *8*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# port 8
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (HTTP probe)** | Configures an HTTP probe name and changes to HTTP probe configuration submode. |
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# predictor (server farm)

To specify the load-balancing algorithm for selecting a real server in the server farm, use the **predictor** server farm configuration command. To restore the default load-balancing algorithm of weighted round robin, use the **no** form of this command.

**predictor [roundrobin | leastconns]**

**no predictor**

**Syntax Description**

| | |
|---|---|
| **roundrobin** | (Optional) Uses the weighted round robin algorithm for selecting the real server to handle the next new connection for the server farm. See the "Weighted Round Robin" section on page 5 for a detailed description of this algorithm. This is the default value. |
| **leastconns** | (Optional) Uses the weighted least connections algorithm for selecting the real server to handle the next new connection for this server farm. See the "Weighted Least Connections" section on page 5 for a detailed description of this algorithm. |

**Defaults**
If you do not enter a **predictor** command, or if you enter the **predictor** command without specifying a load-balancing algorithm, the weighted round robin algorithm is used.

**Command Modes**
Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**
GPRS load balancing requires the weighted round robin algorithm. A server farm that uses weighted least connections can be bound to a virtual server providing GPRS load balancing, but you cannot place the virtual server **INSERVICE**. If you try to do so, IOS SLB issues an error message.

**Examples**
The following example specifies the weighted least connections algorithm:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# predictor leastconns
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **weight (server farm)** | Specifies the real server's capacity, relative to other real servers in the server farm. |

# predictor hash address (firewall farm)

To specify the load-balancing algorithm for selecting a firewall in the firewall farm, use the **predictor hash address** firewall farm configuration command. To restore the default load-balancing algorithm, use the **no** form of this command.

**predictor hash address** [**port**]

**no predictor**

| | |
|---|---|
| **Syntax Description** | **port** | (Optional) Uses the source and destination TCP or UDP port numbers, in addition to the source and destination IP addresses, when selecting a firewall. |

**Defaults**   Uses the source and destination IP addresses when selecting a firewall.

**Command Modes**   Firewall farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**   The following example specifies that source and destination IP addresses are to be used when selecting a firewall:

```
Router(config)# ip slb firewall FIRE1
Router(config-slb-fw)# predictor hash address
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **weight (firewall farm real server)** | Specifies the firewall's capacity, relative to other firewalls in the firewall farm. |

# probe (firewall farm real server)

To associate a probe with a firewall farm, use the **probe** firewall farm real server configuration command. To remove the association, use the **no** form of this command.

**probe** *name*

**no probe** *name*

| | |
|---|---|
| **Syntax Description** | |

| *name* | Name of the probe to associate with this firewall farm. |
|---|---|

**Defaults**

No default behavior or values.

**Command Modes**

Firewall farm real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**

You can configure more than one probe for each firewall in a firewall farm.

If you configure probes in your network, you must also do one of the following:

- Configure the **exclude** keyword on the **client** command on the virtual server, to exclude connections initiated by the client IP address from the load-balancing scheme.
- Configure IP addresses on the IOS SLB device that are Layer 3-adjacent to the real servers used by the virtual server.

**Examples**

The following example associates probe DAWN with server farm FIRE1:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw-real)# probe DAWN
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the server farm configuration. |

# probe (server farm)

To associate a probe with a server farm, use the **probe** server farm configuration command. To remove the association, use the **no** form of this command.

**probe** *name*

**no probe** *name*

| Syntax Description | *name* | Name of the probe to associate with this server farm. |
|---|---|---|

**Defaults**      No default behavior or values.

**Command Modes**      Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Usage Guidelines**      You can configure more than one probe for each server farm.

If you configure probes in your network, you must also do one of the following:

- Configure the **exclude** keyword on the **client** command on the virtual server, to exclude connections initiated by the client IP address from the load-balancing scheme.
- Configure IP addresses on the IOS SLB device that are Layer 3-adjacent to the real servers used by the virtual server.

**Examples**      The following example associates probe PROBE1 with server farm PUBLIC:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# probe PROBE1
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb serverfarms** | Displays information about the server farm configuration. |

# real (firewall farm)

To identify a firewall as a member of a firewall farm and initiate real server configuration mode, use the **real** firewall farm configuration command. To remove the firewall from the IOS SLB configuration, use the **no** form of this command.

**real** *ip-address*

**no real** *ip-address*

**Syntax Description**

| ip-address | Real server IP address. |
|---|---|

**Defaults**
No default behavior or values.

**Command Modes**
Firewall farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**
A firewall farm comprises a number of firewalls. The firewalls are the physical devices that provide the firewall load-balanced services.

**Examples**
The following example identifies a firewall as a member of firewall farm FIRE1:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# real 10.1.1.1
Router(config-slb-real)#
```

**Related Commands**

| Command | Description |
|---|---|
| **inservice (firewall farm real server)** | Enables the firewall for use by IOS SLB. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# real (server farm)

To identify a real server as a member of a server farm and initiate real server configuration mode, use the **real** server farm configuration command. To remove the real server from the IOS SLB configuration, use the **no** form of this command.

**real** *ip-address* [*port_number*]

**no real** *ip-address* [*port_number*]

| Syntax Description | | |
|---|---|---|
| *ip-address* | Real server IP address. | |
| *port_number* | (Optional) Port translation for the server. Valid values range from 1 to 65535. | |

**Defaults**      No default behavior or values.

**Command Modes**      Server farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(2)E | The *port-number* variable was added. |

**Usage Guidelines**      A server farm comprises a number of real servers. The real servers are the physical devices that provide the load-balanced services.

In GPRS load balancing, this command identifies a GGSN that is a member of the server farm. Also, remember that the GGSN IP addresses are virtual template IP addresses, not real interface IP addresses.

**Examples**      The following example identifies a real server as a member of the server farm:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.1.1.1
Router(config-slb-real)#
```

**Related Commands**

| Command | Description |
|---|---|
| **inservice (server farm real server)** | Enables the real server for use by IOS SLB. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# reassign

To specify the threshold of consecutive unacknowledged synchronizations or Create PDP requests that, if exceeded, result in an attempted connection to a different real server, use the **reassign** real server configuration command. To restore the default reassignment threshold, use the **no** form of this command.

**reassign** *threshold*

**no reassign**

| Syntax Description | | |
|---|---|---|
| *threshold* | Number of unacknowledged TCP SYNs (or Create PDP requests, in GPRS load balancing) that are directed to a real server before the connection is reassigned to a different real server. An unacknowledged SYN is one for which no SYN or ACK is detected before the next SYN arrives from the client. IOS SLB allows 30 seconds for the connection to be established or for a new SYN to be received. If neither of these occurs within that time, the connection is removed from the IOS SLB database. | |
| | The 30-second timer is restarted for each SYN as long as the number of connection reassignments specified on the **faildetect (real server)** command's **numconns** keyword is not exceeded. See the **faildetect (real server)** command for more information. | |
| | Valid threshold values range from 1 to 4 SYNs. The default value is 3. | |

**Defaults**  Threshold default: 3 SYNs

**Command Modes**  Real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(9)E | This command was modified to support GPRS load balancing. |

**Usage Guidelines**  IOS SLB does not reassign sticky connections if either of the following conditions is true:

- The real server is not **OPERATIONAL** or **MAXCONNS_THROTTLED**.
- The connection is the first for this sticky.

In GPRS load balancing, this command specifies the number of consecutive unacknowledged Create PDP requests (not TCP SYNs) that are directed to a GGSN before the connection is reassigned to a different GGSN. You must specify a reassign threshold less than the value specified on the **gprs gtp n3-requests** *n3-requests* command (or the equivalent command, if you are not using a Cisco SGSN):

*threshold < n3-requests*

**Examples**      The following example sets the threshold of unacknowledged SYNs to 2:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.10.1.1
Router(config-slb-real)# reassign 2
```

**Related Commands**

| Command | Description |
| --- | --- |
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# replicate casa (firewall farm)

To configure a stateful backup of IOS SLB decision tables to a backup switch, use the **replicate casa** firewall farm configuration command. To remove a **replicate casa** configuration, use the **no** form of this command.

**replicate casa** *listening-ip remote-ip port-number* [*interval*] [**password** [**0** | **7**] *password* [*timeout*]]

**no replicate casa** *listening-ip remote-ip port-number*

**Syntax Description**

| | |
|---|---|
| *listening-ip* | Specifies the listening IP address for state exchange messages that are advertised. |
| *remote-ip* | Specifies the destination IP address for all state exchange signals. |
| *port-number* | The ports and the valid name or number for the *port-number* argument are as follows:<br><br>• Domain Name System: **dns**  **53**<br>• File Transfer Protocol: **ftp**  **21**<br>• HTTP over Secure Socket Layer: **https**  **443**<br>• Mapping of Airline Traffic over IP, Type A: **matip-a**  **350**<br>• Network News Transport Protocol: **nntp**  **119**<br>• Post Office Protocol v2: **pop2**  **109**<br>• Post Office Protocol v3: **pop3**  **110**<br>• Simple Mail Transport Protocol: **smtp**  **25**<br>• Telnet: **telnet**  **23**<br>• World Wide Web (HTTP): **www**  **80** |
| *interval* | (Optional) Maximum replication delivery interval from 1 to 300 seconds. The default value is 10 seconds. |
| **password** | (Optional) Specifies a password for MD5 authentication. |
| **0** | (Optional) Specifies that the password is unencrypted. This is the default setting. |
| **7** | (Optional) Specifies that the password is encrypted. |
| *password* | (Optional) Password value for MD5 authentication. This password must match the password configured on the host agent. |
| *timeout* | (Optional) Delay period, in seconds, during which both the old password and the new password are accepted. The default value is 180 seconds. |

**Defaults**

The interval default is 10 seconds.

The password encryption default is 0 (unencrypted).

The password timeout default is 180 seconds.

**Command Modes**  Firewall farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**  The timeout option allows you to change the password without stopping messages between the backup and primary Layer 3 switches. The default value is 180 seconds.

During the timeout, the backup sends packets with the old password (or null, if there is no old password), and receives packets with either the old or new password. After the timeout expires, the backup sends and receives packets only with the new password.

When setting a new password timeout, keep the following in mind:

- If you are configuring a new backup, set the timeout to **0** (send packets with the new password immediately). This prevents password mismatches between the new backup and its primary.

- If you are changing the password for an existing backup, set a longer timeout. This allows enough time for you to update the password on the primary before the timeout expires. It also prevents mismatches between the backup and primary.

**Examples**  The following example configures a stateful backup Layer 3 switch with a listening IP address of *10.10.10.11*, a remote IP address of *10.10.11.12*, over HTTP port *4231*:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# replicate casa 10.10.10.11 10.10.11.12 4231
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb replicate** | Displays the configuration of IOS SLB IP replication. |
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |

# replicate casa (virtual server)

To configure a stateful backup of IOS SLB decision tables to a backup switch, use the **replicate casa** virtual server configuration command. To remove a **replicate casa** configuration, use the **no** form of this command.

> **replicate casa** *listening-ip remote-ip port-number* [*interval*] [**password** [**0** | **7**] *password* [*timeout*]]

> **no replicate casa** *listening-ip remote-ip port-number*

| Syntax Description | | |
|---|---|---|
| *listening-ip* | | Specifies the listening IP address for state exchange messages that are advertised. |
| *remote-ip* | | Specifies the destination IP address for all state exchange signals. |
| *port-number* | | Specifies the TCP or UDP port number or port name for all state exchange signals. |
| *interval* | | (Optional) Maximum replication delivery interval from 1 to 300 seconds. The default value is 10 seconds. |
| **password** | | (Optional) Specifies a password for MD5 authentication. |
| **0** | | (Optional) Specifies that the password is unencrypted. This is the default setting. |
| **7** | | (Optional) Specifies that the password is encrypted. |
| *password* | | (Optional) Password value for MD5 authentication. This password must match the password configured on the host agent. |
| *timeout* | | (Optional) Delay period, in seconds, during which both the old password and the new password are accepted. The default value is 180 seconds. |

**Defaults**

The interval default is 10 seconds.

The password encryption default is 0 (unencrypted).

The password timeout default is 180 seconds.

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |
| 12.1(3a)E | The **0** and **7** keywords were added. |

**Usage Guidelines**    The timeout option allows you to change the password without stopping messages between the backup and primary Layer 3 switches. The default value is 180 seconds.

During the timeout, the backup sends packets with the old password (or null, if there is no old password), and receives packets with either the old or new password. After the timeout expires, the backup sends and receives packets only with the new password.

When setting a new password timeout, keep the following in mind:

- If you are configuring a new backup, set the timeout to **0** (send packets with the new password immediately). This prevents password mismatches between the new backup and its primary.

- If you are changing the password for an existing backup, set a longer timeout. This allows enough time for you to update the password on the primary before the timeout expires. It also prevents mismatches between the backup and primary.

In GPRS load balancing, the **replicate casa (virtual server)** command is not supported.

**Examples**    The following example configures a stateful backup Layer 3 switch with a listening IP address of *10.10.10.11*, a remote IP address of *10.10.11.12*, over HTTP port *4231*:

```
Router(config)# ip slb vserver VS1
Router(config-slb-vserver)# replicate casa 10.10.10.11 10.10.11.12 4231
```

**Related Commands**

| Command | Description |
| --- | --- |
| **show ip slb replicate** | Displays the configuration of IOS SLB IP replication. |
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

# request method, request url

To configure an HTTP probe to check the status of the real servers, use the **request method** or **request url** configuration command. To remove a **request method** or **request url** configuration, use the **no** form of this command.

**request** [**method** {**get** | **post** | **head** | **name** *name*}] [**url** *path*]

**no request** [**method** {**get** | **post** | **head** | **name** *name*}] [**url** *path*]

**Syntax Description**

| | |
|---|---|
| **method** | Configures the way the data is requested from the server. |
| **get** | Configures the Get method to request data from the server. |
| **post** | Configures the Post method to request data from the server. |
| **head** | Configures the header data type to request data from the server. |
| **name** | Name keyword to be followed by the name of the data to request data from the server. |
| *name* | Configures the name string of the data to send to the servers. The character string is limited to 15 characters. |
| **url** | (Optional) Configures the path from the server. |
| *path* | (Optional) Path from the server. |

**Defaults**

If no values are configured following the method keyword, the default is Get.

If no URL path is set to the server, the default is /.

**Command Modes**

HTTP IOS SLB probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(2)E | This command was introduced. |

**Usage Guidelines**

The **request method/url** command configures the IOS SLB HTTP probe method used to receive data from the server. Only one IOS SLB HTTP probe can be configured for each server farm.

**Examples**

The following example configures an IOS SLB HTTP probe named *PROBE2*, changes the CLI to IOS SLB probe submode, and configures HTTP requests to use the *post* method and the URL */probe.cgi?all*:

```
Router(config)# ip slb probe PROBE2 http
Router(config-slb-probe)# request method post url /probe.cgi?all
```

■ request method, request url

**Related Commands**

| Command | Description |
| --- | --- |
| **ip slb probe (HTTP probe)** | Configures the IOS SLB IP probe name. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

   retry

# retry

To specify how long to wait before a new connection is attempted to a failed server, use the **retry** real server configuration command. To restore the default retry value, use the **no** form of this command.

**retry** *retry-value*

**no retry**

| Syntax Description | | |
|---|---|---|
| *retry-value* | Time, in seconds, to wait after the detection of a server failure before a new connection to the server is attempted. | |
| | If the new connection attempt succeeds, the real server is placed in **OPERATIONAL** state. If the connection attempt fails, the timer is reset, the connection is reassigned, and the process repeats until it is successful or until the server is placed **OUTOFSERVICE** by the network administrator. | |
| | Valid values range from 1 to 3600. The default value is 60 seconds. | |
| | A value of 0 means do not attempt a new connection to the server when it fails. | |

**Defaults**

The retry-value default is 60 seconds.

**Command Modes**

Real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**

The following example specifies that 120 seconds must elapse after the detection of a server failure before a new connection is attempted:

```
Router(config)# ip slb serverfarm PUBLIC
Router(config-slb-sfarm)# real 10.10.1.1
Router(config-slb-real)# retry 120
```

**Related Commands**

| Command | Description |
|---|---|
| **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| **show ip slb serverfarms** | Displays information about the server farm configuration. |
| **show ip slb reals** | Displays information about the real servers. |

# serverfarm

To associate a real server farm with a virtual server, or to configure a backup server farm, use the **serverfarm** virtual server configuration command. To remove the server farm association from the virtual server configuration, use the **no** form of this command.

**serverfarm** *primary-serverfarm-name* [**backup** *backup-serverfarm-name* [**sticky**]]

**no serverfarm** *primary-serverfarm-name* [**backup** *backup-serverfarm-name* [**sticky**]]

| Syntax Description | | |
|---|---|---|
| | *primary-serverfarm-name* | Name of a server farm that has already been defined using the **ip slb serverfarm** command. |
| | **backup** | (Optional) Specifies that a backup server farm is to be configured. |
| | *backup-serverfarm-name* | (Optional) Name of a server farm that has already been defined using the **ip slb serverfarm** command. |
| | **sticky** | (Optional) Specifies that sticky connections are to be used in the backup server farm. |

**Defaults**

If **backup** *backup-serverfarm-name* is not specified, no backup server farm is configured.

If a backup server farm is configured but **sticky** is not specified, sticky connections are not used in the backup server farm.

**Command Modes**

Virtual server configuration

| Command History | Release | Modification |
|---|---|---|
| | 12.0(7)XE | This command was introduced. |
| | 12.1(8a)E | The **backup** and **sticky** keywords and the *backup-serverfarm-name* arguments were added. |

**Usage Guidelines**

In GPRS load balancing, the **sticky** keyword is not supported.

**Examples**

The following example shows how the **ip slb vserver**, **virtual**, and **serverfarm** commands are used to associate the real server farm named PUBLIC with the virtual server named PUBLIC_HTTP.

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# virtual 10.0.0.1 tcp www
Router(config-slb-vserver)# serverfarm PUBLIC
```

| Related Commands | Command | Description |
|---|---|---|
| | **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

| Command | Description |
| --- | --- |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# show ip slb conns

To display the active IOS SLB connections, use the **show ip slb conns** privileged EXEC command.

> **show ip slb conns [vserver** *virtual_server-name* | **client** *ip-address* | **firewall** *firewallfarm-name*]
> **[detail]**

**Syntax Description**

| | |
|---|---|
| **vserver** | (Optional) Displays only those connections associated with a particular virtual server. |
| *virtual_server-name* | (Optional) Name of the virtual server to be monitored. |
| **client** | (Optional) Displays only those connections associated with a particular client IP address. |
| *ip-address* | (Optional) IP address of the client to be monitored. |
| **firewall** | (Optional) Displays only those connections associated with a particular firewall farm. |
| *firewallfarm-name* | (Optional) Name of the firewall farm to be monitored. |
| **detail** | (Optional) Displays detailed connection information. |

**Defaults**

If no options are specified, the command displays output for all active IOS SLB connections.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(7)E | The **firewall** keyword and *firewallfarm-name* variable were added. |

**Examples**

The following example shows IOS SLB active connection data:

```
Router# show ip slb conns

vserver          prot   client             real                   state
-------------------------------------------------------------------------
TEST             TCP    7.150.72.183:328   80.80.90.25:80         INIT
TEST             TCP    7.250.167.226:423  80.80.90.26:80         INIT
TEST             TCP    7.234.60.239:317   80.80.90.26:80         ESTAB
TEST             TCP    7.110.233.96:747   80.80.90.26:80         ESTAB
TEST             TCP    7.162.0.201:770    80.80.90.30:80         CLOSING
TEST             TCP    7.22.225.219:995   80.80.90.26:80         CLOSING
TEST             TCP    7.2.170.148:169    80.80.90.30:80         ZOMBIE
```

*Table 1     show ip slb conns Field Descriptions*

| Field | Description |
|---|---|
| vserver | Name of the virtual server whose connections are being monitored and displayed. Information about each connection is displayed on a separate line. |
| prot | Protocol being used by the connection. |
| client | Client IP address being used by the connection. |
| real | Real IP address of the connection. |
| state | Current state of the connection:<br><br>• CLOSING—IOS SLB TCP connection deactivated (awaiting a delay timeout before cleaning up the connection)<br><br>• ESTAB—IOS SLB TCP connection processed a SYN-SYN/ACK exchange between the client and server<br><br>• FINCLIENT—IOS SLB TCP connection processed a FIN from the client<br><br>• FINSERVER—IOS SLB TCP connection processed a FIN from the server<br><br>• INIT—Initial state of the IOS SLB TCP connection<br><br>• SYNBOTH—IOS SLB TCP connection processed one or more TCP SYNs from both the client and the server<br><br>• SYNCLIENT—IOS SLB TCP connection processed one or more client TCP SYNs<br><br>• SYNSERVER—IOS SLB TCP connection processed one or more server 1 TCP SYNs<br><br>• ZOMBIE—Destruction of the IOS SLB TCP connection failed, possibly because of bound flows. Destruction will proceed when the flows are unbound. |

# show ip slb dfp

To display DFP manager and agent information, such as passwords, timeouts, retry counts, and weights, use the **show ip slb dfp** privileged EXEC command.

> **show ip slb dfp [agent** *agent_ip_address port-number* | **manager** *manager_ip_address* | **detail** | **weights]**

**Syntax Description**

| | |
|---|---|
| **agent** | (Optional) Displays information about an agent. |
| *agent_ip_address* | (Optional) Agent IP address. |
| *port-number* | (Optional) Agent TCP or UDP port number. |
| **manager** | (Optional) Displays information about a manager. |
| *manager_ip_address* | (Optional) Manager IP address. |
| **detail** | (Optional) Displays all data available. |
| **weights** | (Optional) Displays information about weights assigned to real servers for load balancing. |

**Defaults**

If no options are specified, the command displays summary information.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(5a)E | The **manager** keyword and *manager_ip_address* variable were added. |

**Usage Guidelines**

The following example displays high-level information about all DFP agents and managers:

```
Router# show ip slb dfp
DFP Manager:
     Current passwd:NONE Pending passwd:NONE
     Passwd timeout:0 sec

Agent IP       Port    Timeout   Retry Count   Interval
-----------------------------------------------------------
161.44.2.34    61936   0         0             180 (Default)
```

*Table 2     show ip slb dfp Field Descriptions*

| Field | Description |
|---|---|
| Agent IP | IP address of the agent about which information is being displayed. |
| Port | TCP or UDP port number of the agent. |

*Table 2        show ip slb dfp Field Descriptions (continued)*

| Timeout | Time period, in seconds, during which the DFP manager must receive an update from the DFP agent. A value of 0 means there is no timeout. |
|---|---|
| Retry Count | Number of times the DFP manager attempts to establish the TCP connection to the DFP agent. A value of 0 means there are infinite retries. |
| Interval | Interval, in seconds, between retries. |
| State | Current state of the connection. <br><br> • CONNING—This state is set after the Agent opens the socket, and is connecting. <br><br> • FAILED—This state is set when there is an error in the data, or when the maximum number of retries is reached. <br><br> • NOT_OPEN—This is the initial state when the Agent block is created. <br><br> • OPEN—The Agent is talking to the Host. This is the functioning state. <br><br> • SEC_ERR—This state is set when there is a security error in the received message. Either there is no security vector when one was required, or the MD5 string in the message did not match the expected result (password not correct). <br><br> • TRYING—When the Agent tries to open the socket, it is moved to this state. |
| Timeout | Time period, in seconds, during which the DFP manager must receive an update from the DFP agent. A value of 0 means there is no timeout. |

The following example displays detailed information about DFP agents and managers:

```
Router# show ip slb dfp detail
DFP Manager
      Current passwd <none> Pending passwd <none>
      Passwd timeout 0 sec
      Unexpected errors 0
% No DFP Agents configured
```

The following example displays detailed information about DFP manager 55.55.55.2:

```
Router# show ip slb dfp manager 55.55.55.2
DFP Manager 55.55.55.2 Connection state Connected
   Timeout = 20
   Last message sent 033537 UTC 01/02/00
```

The following example displays detailed information about weights assigned to real servers for load balancing:

```
Router# show ip slb dfp weights
Real IP Address 17.17.17.17 Protocol TCP Port 22 Bind_ID 111 Weight 111
      Set by Agent 161.44.2.3458490 at 132241 UTC 12/03/99
Real IP Address 17.17.17.17 Protocol TCP Port www Bind_ID 1 Weight 1
      Set by Agent 161.44.2.3458490 at 132241 UTC 12/03/99
Real IP Address 68.68.68.68 Protocol TCP Port www Bind_ID 4 Weight 4
      Set by Agent 161.44.2.3458490 at 132241 UTC 12/03/99
Real IP Address 85.85.85.85 Protocol TCP Port www Bind_ID 5 Weight 5
      Set by Agent 161.44.2.3458490 at 132241 UTC 12/03/99
```

# show ip slb firewallfarm

To display firewall farm information, use the **show ip slb firewallfarm** configuration command.

**show ip slb firewallfarm** [**detail**]

| Syntax Description | **detail** | (Optional) Displays detailed information. |
|---|---|---|

**Defaults**   No default behavior or values.

**Command Modes**   Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**   The following example shows firewall farm data:

```
Router# show ip slb firewallfarm

firewall farm    hash       state         reals
-----------------------------------------------
FIRE1            IPADDR     OPERATIONAL   2
```

*Table 3        show ip slb firewallfarm Field Descriptions*

| Field | Description |
|---|---|
| firewall farm | Name of the firewall farm. |
| hash | Load-balancing algorithm used to select a firewall for the firewall farm:<br><br>• IPADDR—Uses the source and destination IP addresses in the algorithm.<br><br>• IPADDRPORT—Uses the source and destination TCP or UDP port numbers, in addition to the source and destination IP addresses, in the algorithm.<br><br>See the **predictor hash address (firewall farm)** command for more details. |

*Table 3      show ip slb firewallfarm Field Descriptions (continued)*

| state | Current state of the firewall farm. |
|-------|-------------------------------------|
|       | • OPERATIONAL—Functioning properly |
|       | • OUTOFSERVICE—Removed from the load-balancing predictor lists |
|       | • STANDBY—Backup firewall farm, ready to become operational if active firewall farm fails |
| reals | Number of firewalls that are members of the firewall farm. |

# show ip slb natpool

To display the IP IOS SLB NAT configuration, use the **show ip slb natpool** command.

**show ip slb natpool** [**name** *pool-name*] [**detail**]

**Syntax Description**

| name | (Optional) Keyword to display a specific NAT pool. |
| --- | --- |
| *pool-name* | (Optional) NAT pool name string to display. |
| detail | (Optional) Lists all the interval ranges currently allocated in the client NAT pool. |

**Defaults**

No default behavior or values.

**Command Modes**

EXEC configuration

**Command History**

| Release | Modification |
| --- | --- |
| 12.1(2)E | This command was introduced. |

**Examples**

The following example displays the default **show ip slb natpool** command:

```
Router# show ip slb natpool
nat client B  1.1.1.6  1.1.1.8  Netmask 255.255.255.0
     nat client A  1.1.1.1  1.1.1.5  Netmask 255.255.255.0
```

The following example displays the **show ip slb natpool** command with the additional detail parameter:

```
Router# show ip slb natpool detail
nat client A  1.1.1.1  1.1.1.5  Netmask 255.255.255.0
    Start NAT        Last NAT          Count        ALLOC/FREE
    --------------------------------------------------------
    1.1.1.1:11001    1.1.1.1:16333     0005333      ALLOC
    1.1.1.1:16334    1.1.1.1:19000     0002667      ALLOC
    1.1.1.1:19001    1.1.1.5:65535     0264675      FREE

nat client B  1.1.1.6  1.1.1.8  Netmask 255.255.255.0
    Start NAT        Last NAT          Count        ALLOC/FREE
    --------------------------------------------------------
    1.1.1.6:11001    1.1.1.6:16333     0005333      ALLOC
    1.1.1.6:16334    1.1.1.6:19000     0002667      ALLOC
    1.1.1.6:19001    1.1.1.8:65535     0155605      FREE
```

**Related Commands**

| Command | Description |
| --- | --- |
| **ip slb natpool** | Configures the IOS SLB NAT. |

# show ip slb probe

To display information about an IOS SLB probe, use the **show ip slb probe** configuration command.

**show ip slb probe** [**name** *probe_name*] [**detail**]

**Syntax Description**

| name | (Optional) Displays information about the specific probe named. |
|------|----------------------------------------------------------------|
| *probe_name* | (Optional) Probe name to display. |
| detail | (Optional) Displays detailed information, including the SA Agent operation ID, which you can use to correlate the output of the **show rtr operational-state** EXEC command. |

**Defaults**

No default behavior or values.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---------|--------------|
| 12.1(2)E | This command was introduced. |

**Examples**

The following example shows IOS SLB probe data:

```
Router# show ip slb probe

Server:Port          State          Outages  Current  Cumulative
---------------------------------------------------------------
10.10.4.1:0          OPERATIONAL         0  never     00:00:00
10.10.5.1:0          FAILED              1  00:00:06 00:00:06
```

*Table 4       show ip slb probe Field Descriptions*

| Field | Description |
|-------|-------------|
| Server:Port | IP address and port of the real server. |
| State | Operational state of the probe: <br>• FAILED—The probe has succeeded in the past but has currently failed. <br>• OPERATIONAL—The probe is functioning normally. <br>• TESTING—The probe has never succeeded, due to no response. IOS SLB keeps no counters or timers for this state. |
| Outages | Number of intervals between successful probes. |

*Table 4 show ip slb probe Field Descriptions (continued)*

| | |
|---|---|
| Current | Time since the last probe success. That is, the duration (so far) of the current outage. |
| Cumulative | Total time the real server has been under test by the probe and has failed the probe test. This value is the sum of the **Current** time plus the total time of all previous outages. |

# show ip slb reals

To display information about the real servers, use the **show ip slb reals** privileged EXEC command.

**show ip slb reals [vserver** *virtual_server-name***] [detail]**

**Syntax Description**

| | |
|---|---|
| **vserver** | (Optional) Displays information about only those real servers associated with a particular virtual server. |
| *virtual_server-name* | (Optional) Name of the virtual server. |
| **detail** | (Optional) Displays detailed information. |

**Defaults**

If no options are specified, the command displays information about all real servers.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**

The following example shows IOS SLB real server data:

```
Router# show ip slb reals

real            farm name       weight  state           conns
------------------------------------------------------------------
80.80.2.112     FRAG            8       OUTOFSERVICE    0
80.80.5.232     FRAG            8       OPERATIONAL     0
80.80.15.124    FRAG            8       OUTOFSERVICE    0
80.254.2.2      FRAG            8       OUTOFSERVICE    0
80.80.15.124    LINUX           8       OPERATIONAL     0
80.80.15.125    LINUX           8       OPERATIONAL     0
80.80.15.126    LINUX           8       OPERATIONAL     0
80.80.90.25     SRE             8       OPERATIONAL     220
80.80.90.26     SRE             8       OPERATIONAL     216
80.80.90.27     SRE             8       OPERATIONAL     216
80.80.90.28     SRE             8       TESTING         1
80.80.90.29     SRE             8       OPERATIONAL     221
80.80.90.30     SRE             8       OPERATIONAL     224
80.80.30.3      TEST            100     READY_TO_TEST   0
80.80.30.4      TEST            100     READY_TO_TEST   0
80.80.30.5      TEST            100     READY_TO_TEST   0
80.80.30.6      TEST            100     READY_TO_TEST   0
```

*Table 5      show ip slb reals Field Descriptions*

| Field | Description |
|-------|-------------|
| real | IP address of the real server about which information is being displayed. Used to identify each real server. Information about each real server is displayed on a separate line. |
| farm name | Name of the server farm or firewall farm with which the real server is associated. |
| weight | Weight assigned to the real server. The weight identifies the real server's capacity, relative to other real servers in the server farm. |
| state | Current state of the real server.<br><br>• DFP_THROTTLED—The DFP agent sent a weight of **0** for this real server (send no further connections to this real server).<br><br>• FAILED—The real server has failed due as a result of either no response or RST responses to client traffic. (See the **faildetect (real server)** command for more information about controlling tolerance for no-responses and RSTs). The real server has been removed from use by the predictor algorithms. The retry timer has started.<br><br>• MAXCONNS_THROTTLE—The number of connections on the real server exceeds the configured maximum number of simultaneous active connections (**maxconns**).<br><br>• OPERATIONAL—The real server is functioning properly and is being used for load-balancing.<br><br>• OPER_WAIT—The real server is waiting to become operational (waiting for a timeout or some other condition to be met).<br><br>• OUTOFSERVICE—The real server was configured with **no inservice** and has been removed from the load-balancing predictor lists.<br><br>• PROBE_FAILED—The probe has succeeded in the past but has currently failed. This might occur at the same time user connections fail, or it might not.<br><br>• PROBE_TESTING—The probe has never succeeded, due to no response. The initial probe timed out waiting for a success.<br><br>• READY_TO_TEST—The real server is queued for testing after being in FAILED state until the retry timer expired.<br><br>• TESTING—The real server is queued for assignment. When a single user connection is assigned to a real server that is in READY_TO_TEST state, the real server is placed in TESTING state. If the test succeeds, the real server is placed back in OPERATIONAL state.<br><br>• TEST_WAIT—The real server is waiting to begin testing (waiting for a timeout or some other condition to be met). |
| conns | Number of connections (or sessions, in GPRS load balancing) associated with the real server. |

The following is sample output from the **show ip slb reals detail** command for a real server in a server farm:

```
Router# show ip slb reals detail

10.10.1.7, S, state = OPERATIONAL, type = server
  conns = 0, dummy_conns = 0, maxconns = 4294967295
  weight = 8, weight(admin) = 8, metric = 0, remainder = 0
  reassign = 3, retry = 60
  failconn threshold = 8, failconn count = 0
  failclient threshold = 2, failclient count = 0
  total conns established = 0, total conn failures = 0
  server failures = 0
```

The following is sample output from the **show ip slb reals detail** command for a real server in a firewall farm:

```
Router# show ip slb reals detail

10.10.3.2, F, state = OPERATIONAL, type = firewall
  conns = 0, dummy_conns = 0, maxconns = 4294967295
  weight = 8, weight(admin) = 8, metric = 0, remainder = 0
  total conns established = 8377, hash count = 0
  server failures = 0
  interface FastEthernet1/0, MAC 0000.0c41.1063
```

Table 6 describes the fields shown in the above detail displays.

*Table 6      show ip slb reals detail Field Descriptions*

| Field | Description |
|-------|-------------|
| IP address | IP address of the real server about which information is being displayed. Used to identify each real server. Information about each real server is displayed on a separate line. |
| farm name | Name of the server farm or firewall farm with which the real server is associated. |

*Table 6       show ip slb reals detail Field Descriptions (continued)*

| state | Current state of the real server. |
|---|---|
| | • DFP_THROTTLED—The Dynamic Feedback Protocol (DFP) agent sent a weight of 0 for this real server (send no further connections to this real server). |
| | • FAILED—The real server has failed as a result of either no response or reset (RST) responses to client traffic. (See the **faildetect (real server)** command for more information about controlling tolerance for no responses and RSTs.) The real server has been removed from use by the predictor algorithms. The retry timer has started. |
| | • MAXCONNS_THROTTLE—The number of connections on the real server exceeds the configured maximum number of simultaneous active connections (**maxconns**). |
| | • OPERATIONAL—The real server is functioning properly and is being used for load-balancing. |
| | • OPER_WAIT—The real server is waiting to become operational (waiting for a timeout or some other condition to be met). |
| | • OUTOFSERVICE—The real server was configured with **no inservice** and has been removed from the load-balancing predictor lists. |
| | • PROBE_FAILED—The probe has succeeded in the past but has currently failed. This failure might occur at the same time user connections fail, or it might not. |
| | • PROBE_TESTING—The probe has never succeeded, due to no response. The initial probe timed out waiting for a success. |
| | • READY_TO_TEST—The real server is queued for testing after being in FAILED state until the retry timer expired. |
| | • TESTING—The real server is queued for assignment. When a single user connection is assigned to a real server that is in READY_TO_TEST state, the real server is placed in TESTING state. If the test succeeds, the real server is placed back in OPERATIONAL state. |
| | • TEST_WAIT—The real server is waiting to begin testing (waiting for a timeout or some other condition to be met). |
| type | Indicates whether the real server is associated with a server farm (server) or firewall farm (firewall). |
| conns | Number of connections associated with the real server. In general packet radio service (GPRS) load balancing, number of sessions associated with the real server. In per-packet server load balancing, number of request packets that have been load balanced to each real server, using the connection count. |
| dummy_conns | Internal counter used in debugging. |
| maxconns | Maximum number of active connections allowed on the real server at one time. |
| weight | Weight assigned to the real server. The weight identifies the real server's capacity, relative to other real servers in the server farm. This value could be changed by DFP. |

*Table 6    show ip slb reals detail Field Descriptions (continued)*

| | |
|---|---|
| weight(admin) | Configured (or default) weight assigned to the real server. |
| metric | Internal counter used in debugging. |
| remainder | Internal counter used in debugging. |
| reassign | Total number of consecutive unacknowledged SYNchronize sequence numbers (SYNs) or Create Packet Data Protocol (PDP) requests since the last time the **clear ip slb counters** command was issued. |
| retry | Interval, in seconds, to wait between the detection of a failure on the real server and the next attempt to connect to the server. |
| failconn threshold | Maximum number of consecutive connection failures allowed before the real server is considered to have failed. |
| failconn count | Total number of consecutive connection failures since the last time the **clear ip slb counters** command was issued. |
| failclient threshold | Maximum number of unique client connection failures allowed before the real server is considered to have failed. |
| failclient count | Total number of unique client connection failures since the last time the **clear ip slb counters** command was issued. |
| total conns established | Total number of successful connection assignments since the last time the **clear ip slb counters** command was issued. |
| total conn failures | Total number of unsuccessful connection assignments since the last time the **clear ip slb counters** command was issued. |
| server failures | Total number of times this real server has been marked failed. |
| hash count | Total number of times the hash algorithm has been called. |
| interface | Type of interface. |
| MAC | MAC address of the firewall. |

# show ip slb replicate

To display the IOS SLB replication configuration, use the **show ip slb replicate** privileged EXEC command.

**show ip slb replicate**

**Syntax Description**    This command has no arguments or keywords.

**Defaults**    No default behavior or values.

**Command Modes**    Privileged EXEC

**Command History**

| Release | Modification |
|---------|--------------|
| 12.1(2)E | This command was introduced. |

**Examples**    The following example displays the IOS SLB replication configuration:

```
Router# show ip slb replicate
VS1, local = 10.10.99.132 remote = 10.10.99.99 port = 1024
  current password = none pending password = none
  password timeout = 180 sec (Default)
  unsent conn updates:        0
  conn updates received:      32
  conn updates transmitted:   471
  update packets received:    12
  update packets transmitted: 34
  failovers:                  0
Router#
```

Table 7 describes the fields shown in the display.

*Table 7        show ip slb replicate Field Descriptions*

| Field | Description |
|-------|-------------|
| local | Listening IP address for state exchange messages that are advertised. |
| remote | Destination IP address for all state exchange signals. |
| port | TCP or User Datagram Protocol (UDP) port number or port name for all state exchange signals. |
| current password | Current password for Message Digest Algorithm Version 5 (MD5) authentication, if any. |
| pending password | Pending password for MD5 authentication, if any. |
| password timeout | Delay period during which both the current (old) password and the pending (new) password are accepted. |

*Table 7        show ip slb replicate Field Descriptions (continued)*

| | |
|---|---|
| unsent conn updates | Number of connection updates (changes in connection status) waiting to be sent to the remote switch. |
| conn updates received | Number of connection updates (changes in connection status) received from the remote switch. |
| conn updates transmitted | Number of connection updates (changes in connection status) sent to the remote switch. |
| update packets received | Number of packets containing connection updates (changes in connection status) received from the remote switch. |
| update packets transmitted | Number of packets containing connection updates (changes in connection status) sent to the remote switch. |
| failovers | Number of failovers detected. That is, the number of times the Hot Standby Router Protocol (HSRP) state has changed from Active to Standby, or from Standby to Active. |

**Related Commands**

| Command | Description |
|---|---|
| **request method, request url** | Configures an HTTP probe to check the status of the real servers. |

# show ip slb serverfarms

To display information about the server farms, use the **show ip slb serverfarms** privileged EXEC command.

**show ip slb serverfarms** [**name** *serverfarm-name*] [**detail**]

**Syntax Description**

| name | (Optional) Displays information about only a particular server farm. |
|---|---|
| *serverfarm-name* | (Optional) Name of the server farm. |
| **detail** | (Optional) Displays detailed server farm information. |

**Defaults**

No default behavior or values.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**

The following example shows IOS SLB server farm data:

```
Router# show ip slb serverfarms

server farm     predictor      reals   bind id
------------------------------------------------
FRAG            ROUNDROBIN     4       0
LINUX           ROUNDROBIN     3       0
SRE             ROUNDROBIN     6       0
TEST            ROUNDROBIN     4       0
```

*Table 8        show ip slb serverfarms Field Descriptions*

| Field | Description |
|---|---|
| server farm | Name of the server farm about which information is being displayed. Information about each server farm is displayed on a separate line. |
| predictor | Type of load-balancing algorithm (ROUNDROBIN or LEASTCONNS) used by the server farm. |
| reals | Number of real servers configured in the server farm. |
| bind id | Bind ID configured on the server farm. |

# show ip slb stats

To display IOS SLB statistics, use the **show ip slb stats** privileged EXEC command.

> **show ip slb stats**

**Syntax Description**     This command has no arguments or keywords.

**Defaults**     No default behavior or values.

**Command Modes**     Privileged EXEC

**Command History**

| Release | Modification |
|---------|-------------|
| 12.0(7)XE | This command was introduced. |
| 12.1(9)E | This command was modified to support GPRS load balancing. |

**Examples**     The following example shows IOS SLB statistics:

```
Router# show ip slb stats
 Pkts via normal switching:   0
 Pkts via special switching:  12534087
 Connections Created:         12466960
 Connections Established:     12240100
 Connections Destroyed:       12466960
 Connections Reassigned:      0
 Zombie Count:                0
 Connections Reused:          0
 Connection Flowcache Purges: 0
 Failed Connection Allocs:    0
 Failed Real Assignments:     0
 Total indications:           0
   Unknown conn indications:  0
   SYNACK indications:        0
   RST client indications:    0
   RST server indications:    0
   Two FINs indications:      0
   Flow age indications:      0
```

*Table 9     show ip slb stats Field Descriptions*

| Field | Description |
|-------|-------------|
| Pkts via normal switching | Number of packets handled by IOS SLB via normal switching since the last time counters were cleared. Normal switching is when IOS SLB packets are handled on normal IOS switching paths (CEF, fastswitching, and process level switching). |

*Table 9    show ip slb stats Field Descriptions (continued)*

| | |
|---|---|
| Pkts via special switching | Number of packets handled by IOS SLB via special switching since the last time counters were cleared. Special switching is when IOS SLB packets are handled on hardware-assisted switching paths. |
| Connections Created | Number of connections (or sessions, in GPRS load balancing) created since the last time counters were cleared. |
| Connections Established | Number of connections (or sessions, in GPRS load balancing) created and have become established since the last time counters were cleared. |
| Connections Destroyed | Number of connections (or sessions, in GPRS load balancing) destroyed since the last time counters were cleared. |
| Connections Reassigned | Number of connections (or sessions, in GPRS load balancing) reassigned to a different real server since the last time counters were cleared. |
| Zombie Count | Number of connections (or sessions, in GPRS load balancing) that are currently pending destruction (awaiting a timeout or some other condition to be met). |
| Connections Reused | Number of zombie connections (or sessions, in GPRS load balancing) reused since the last time counters were cleared. A zombie connection is reused if it receives a TCP SYN or UDP packet and succeeds in connecting to a real server. The zombie connection becomes a real connection and the zombie count is decremented. |
| Connection Flowcache Purges | Number of times the connection flow cache was purged since the last time counters were cleared. |
| Failed Connection Allocs | Number of times the allocation of a connection (or session, in GPRS load balancing) failed since the last time counters were cleared. |
| Failed Real Assignments | Number of times the assignment of a real server failed since the last time counters were cleared. |
| Total indications | Total number of indications of all types since the last time counters were cleared. |
| Unknown conn indications | Number of times unknown connections (or sessions, in GPRS load balancing) were detected since the last time counters were cleared. |
| SYNACK indications | Number of TCP SYN/ACK events since the last time counters were cleared. |
| RST client indications | Number of TCP RST client events since the last time counters were cleared. |
| RST server indications | Number of TCP RST server events since the last time counters were cleared. |
| Two FINs indications | Number of times two FINs (one in each direction) were detected on the connection since the last time counters were cleared. |
| Flow age indications | Number of times TCP flows aged out since the last time counters were cleared. |

# show ip slb sticky

To display the IOS SLB sticky database, use the **show ip slb sticky** privileged EXEC command.

> **show ip slb sticky [client** *ip_address*]

| Syntax Description | | |
|---|---|---|
| **client** | | (Optional) Displays only those sticky database entries associated with a particular client IP address or subnet. |
| *ip-address* | | (Optional) IP address of the client. |

**Defaults**

If no options are specified, the command displays information about all virtual servers or firewall farms.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**

The following example shows the IOS SLB sticky database:

```
Router# show ip slb sticky

client          netmask        group  real                conns
------------------------------------------------------------------
10.10.2.12      255.255.0.0    4097   10.10.3.2           1
```

*Table 10     show ip slb sticky Field Descriptions*

| Field | Description |
|---|---|
| client | Client IP address or subnet which is bound to this sticky assignment. |
| netmask | Subnet mask for this sticky assignment. |
| group | Group ID for this sticky assignment. |
| real | Real server used by all clients connecting with the client IP address or subnet detailed on this line. |
| conns | Number of connections currently sharing this sticky assignment. |

# show ip slb vserver

To display information about the virtual servers, use the **show ip slb vserver** privileged EXEC command.

**show ip slb vserver [name** *virtual_server-name*] **[detail]**

**Syntax Description**

| name | (Optional) Displays information about only this virtual server. |
|---|---|
| *virtual_server-name* | (Optional) Name of the virtual server. |
| **detail** | (Optional) Displays detailed information. |

**Defaults**

If no options are specified, the command displays information about all virtual servers.

**Command Modes**

Privileged EXEC

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Examples**

The following example shows virtual server data:

```
Router# show ip slb vserver

slb vserver     prot   virtual                state        conns
----------------------------------------------------------------
TEST            TCP    80.80.254.3:80         OPERATIONAL  1013
TEST21          TCP    80.80.254.3:21         OUTOFSERVICE 0
TEST23          TCP    80.80.254.3:23         OUTOFSERVICE 0
```

*Table 11     show ip slb vserver Field Descriptions*

| Field | Description |
|---|---|
| slb vserver | Name of the virtual server about which information is being displayed. Information about each virtual server is displayed on a separate line. |
| prot | Protocol being used by the virtual server. |
| virtual | Virtual IP address of the virtual server, including the network mask, if configured. |

*Table 11     show ip slb vserver Field Descriptions (continued)*

| state | Current state of the virtual server: |
|---|---|
| | • FAILED—Real server represented by this virtual server has been removed from use by the predictor algorithms; retry timer started |
| | • OPERATIONAL—Functioning properly |
| | • OUTOFSERVICE—Removed from the load-balancing predictor lists |
| | • STANDBY—Backup virtual server, ready to become operational if active virtual server fails |
| | • TESTING_REAL—Real server represented by this virtual server is queued for assignment |
| conns | Number of connections (or sessions, in GPRS load balancing) associated with the virtual server. |

The following example shows detailed data for a virtual server with route health injection (**advertise=TRUE**):

```
Router#show ip slb vserver detail
RH1, state = OPERATIONAL, v_index = 6
  virtual = 5.5.5.5/32:80, TCP, service = NONE, advertise = TRUE
  server farm = RHSF, delay = 10, idle = 3600
  backup server farm = BACKUP, use count = 0, backup sticky = FALSE
  sticky timer = 0, sticky subnet = 255.255.255.255
  sticky group id = 0
  synguard counter = 0, synguard period = 0
  conns = 1, total conns = 31484, syns = 0, syn drops = 0
  standby group = None
```

*Table 12     show ip slb vserver detail Field Descriptions*

| Field | Description |
|---|---|
| RH1 | Name of the virtual server about which information is being displayed (in this case, RH1). Information about each virtual server is displayed on a separate line. |
| state | Current state of the virtual server: |
| | FAILED—Real server represented by this virtual server has been removed from use by the predictor algorithms; retry timer started |
| | OPERATIONAL—Functioning properly |
| | OUTOFSERVICE—Removed from the load-balancing predictor lists |
| | STANDBY—Backup virtual server, ready to become operational if active virtual server fails |
| | TESTING_REAL—Real server represented by this virtual server is queued for assignment |
| v_index | Virtual index, out of a maximum of 127. |
| virtual | Virtual IP address of the virtual server, including the network mask, if configured. |
| TCP | Protocol being used by the virtual server (in this case, TCP). |

*Table 12      show ip slb vserver detail Field Descriptions (continued)*

| | |
|---|---|
| service | Service, such as HTTP or Telnet, associated with the virtual server. |
| advertise | Current state of host route advertisement for this virtual server: TRUE—Host route is being advertised. FALSE—Host route is not being advertised. |
| server farm | Name of the server farm associated with the virtual server. |
| delay | Delay timer duration, in seconds, for this virtual server. |
| idle | Idle connection timer duration, in seconds, for this virtual server. |
| backup server farm | Name of the backup server farm associated with the virtual server. |
| use count | Number of times the backup server farm has taken over for the primary server farm in this period. |
| backup sticky | Indicates whether sticky connections are used in the backup server farm: TRUE—Sticky connections are used in the backup server farm. FALSE—Sticky connections are not used in the backup server farm. |
| sticky timer | Sticky timer duration, in seconds, for this virtual server. |
| sticky subnet | Sticky subnet in which this virtual server is placed, for coupling of services. |
| sticky group id | Sticky group in which this virtual server is placed, for coupling of services. |
| synguard counter | Number of unacknowledged SYNs that are allowed to be outstanding to this virtual server. |
| synguard period | Interval, in milliseconds, for SYN threshold monitoring for this virtual server. |
| conns | Number of active connections currently associated with the virtual server. |
| total conns | Total number of connections that have been associated with the virtual server since coming **INSERVICE**. |
| syns | Number of SYNs handled by the virtual server in this period. |
| syn drops | Number of SYNs dropped by the virtual server in this period. |
| standby group | HSRP group name with which the virtual server is associated. |

# sticky (firewall farm TCP protocol)

To assign all connections from a client to the same firewall, use the **sticky** firewall farm TCP protocol configuration command. To remove the client/server coupling use the **no** form of this command.

**sticky** *duration* [**netmask** *netmask*]

**no sticky**

| Syntax Description | | |
|---|---|---|
| *duration* | Sticky timer duration in seconds. Valid values range from 0 to 65535. | |
| **netmask** | (Optional) Places the virtual server as part of a sticky subnet, for coupling of services. | |
| *netmask* | (Optional) Sticky subnet mask number. | |

**Defaults**    Virtual servers are not associated with any groups.

**Command Modes**    Firewall farm TCP protocol configuration

| Command History | Release | Modification |
|---|---|---|
| | 12.1(3a)E | This command was introduced. |

**Examples**    The following example specifies that if a client's subsequent request for a firewall farm is made within 60 seconds of the previous request, then the same firewall is used for the connection:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# tcp
Router(config-slb-fw-tcp)# sticky 60
```

| Related Commands | Command | Description |
|---|---|---|
| | **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| | **show ip slb sticky** | Displays information about the IOS SLB database. |
| | **tcp** | Enters TCP protocol configuration mode. |

# sticky (firewall farm UDP protocol)

To assign all connections from a client to the same firewall, use the **sticky** firewall farm UDP protocol configuration command. To remove the client/server coupling use the **no** form of this command.

**sticky** *duration* [**netmask** *netmask*]

**no sticky**

**Syntax Description**

| | |
|---|---|
| *duration* | Sticky timer duration in seconds. Valid values range from 0 to 65535. |
| **netmask** | (Optional) Places the virtual server as part of a sticky subnet, for coupling of services. |
| *netmask* | (Optional) Sticky subnet mask number. |

**Defaults**

Virtual servers are not associated with any groups.

**Command Modes**

Firewall farm UDP protocol configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example specifies that if a client's subsequent request for a firewall farm is made within 60 seconds of the previous request, then the same firewall is used for the connection:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# udp
Router(config-slb-fw-udp)# sticky 60
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| **show ip slb sticky** | Displays information about the IOS SLB database. |
| **udp** | Enters UDP protocol configuration mode. |

# sticky (virtual server)

To assign all connections from a client to the same real server, use the **sticky** virtual server configuration command. To remove the client/server coupling use the **no** form of this command.

**sticky** *duration* [**group** group-*id*] [**netmask** *netmask*]

**no sticky**

**Syntax Description**

| | |
|---|---|
| *duration* | Sticky timer duration in seconds. Valid values range from 0 to 65535. |
| **group** | (Optional) Places the virtual server in a sticky group, for coupling of services. In essence, the **group** keyword and *group-id* argument tie multiple virtual servers together. |
| *group-id* | (Optional) Number identifying the sticky group to which the virtual server belongs. Valid values range from 0 to 255. |
| **netmask** | (Optional) Places the virtual server as part of a sticky subnet, for coupling of services. Client sessions whose source IP addresses fall within the *netmask* are directed to the same real server. |
| *netmask* | (Optional) Sticky subnet mask number. |

**Defaults**

Sticky connections are not tracked.

Virtual servers are not associated with any groups.

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(2)E | The **netmask** keyword and *netmask* variable were added. |

**Usage Guidelines**

The last real server that was used for a connection from a client is stored for the set *duration* seconds. If a new connection from the client to the virtual server is initiated during that time, the same real server that was used for the previous connection is chosen for the new connection. If two virtual servers are placed in the same group, coincident connection requests for those services from the same IP address are handled by the same real server.

In GPRS load balancing, the **sticky** command is not supported.

**Examples**

The following example specifies that if a client's subsequent request for a virtual server is made within 60 seconds of the previous request, then the same real server is used for the connection. This example also places the virtual server in group 10.

```
Router(config)# ip slb vserver VS1
Router(config-slb-vserver)# sticky 60 group 10
```

**Related Commands**

| Command | Description |
| --- | --- |
| **show ip slb sticky** | Displays information about the IOS SLB database. |
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# synguard (virtual server)

To limit the rate of TCP SYNs handled by a virtual server to prevent a SYN flood denial-of-service attack, use the **synguard** virtual server configuration command. To remove the threshold, use the **no** form of this command.

**synguard** *syn-count* [*interval*]

**no synguard**

| Syntax Description | | |
|---|---|---|
| | *syn-count* | Number of unacknowledged SYNs that are allowed to be outstanding to a virtual server. Valid values range from 0 (off) to 4294967295. The default is 0. |
| | *interval* | (Optional) Interval, in milliseconds, for SYN threshold monitoring. Valid values range from 50 to 5000. The default is 100 ms. |

**Defaults**

Syn-count default: 0 (off)

Interval default: 100 ms

**Command Modes**

Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

**Usage Guidelines**

In GPRS load balancing, the **synguard** command has no meaning.

**Examples**

The following example sets the threshold of unacknowledged SYNs to 50:

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# synguard 50
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |
| **virtual (virtual server)** | Configures the virtual server attributes. |

# tcp

To initiate TCP protocol configuration mode, use the **tcp** firewall farm configuration command.

**tcp**

**Syntax Description**    This command has no arguments or keywords.

**Defaults**    No default behavior or values.

**Command Modes**    Firewall farm configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Usage Guidelines**    There is not a **no** form of this command.

**Examples**    The following example initiates TCP protocol configuration mode:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# tcp
Router(config-slb-fw-tcp)# exit
```

**Related Commands**

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |

# udp

To initiate UDP protocol configuration mode, use the **udp** firewall farm configuration command.

**udp**

## Syntax Description

This command has no arguments or keywords.

## Defaults

No default behavior or values.

## Command Modes

Firewall farm configuration

## Command History

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

## Usage Guidelines

There is not a **no** form of this command.

## Examples

The following example initiates UDP protocol configuration mode:

```
Router(config)# ip slb firewallfarm FIRE1
Router(config-slb-fw)# udp
Router(config-slb-fw-udp)# exit
```

## Related Commands

| Command | Description |
|---|---|
| **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |

# url (WSP probe)

To specify the URL path that a WSP probe is to request from the server, use the **url** WSP probe configuration command. To restore the default settings, use the **no** form of this command.

**url** [*path*]

**no url** [*path*]

| Syntax Description | *path* | (Optional) Path from the server. This argument is case-sensitive. |
|---|---|---|

**Defaults**    If no URL path is specified, the default is /.

**Command Modes**    WSP probe configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(5a)E | This command was introduced. |

**Examples**    The following example configures a ping probe named *PROBE3*, changes the CLI to IOS SLB WSP probe submode, and configures the probe to request URL path *http://localhost/test.txt*:

```
Router(config)# ip slb probe PROBE3 wsp
Router(config-slb-probe)# url http://localhost/test.txt
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb probe (WSP probe)** | Configures a WSP probe name and changes to WSP probe configuration submode. |
| **show ip slb probe** | Displays information about an IOS SLB probe. |

# virtual (virtual server)

To configure virtual server attributes, use the **virtual** virtual server configuration command. To remove the attributes, use the **no** form of this command.

> **virtual** *ip-address* [*network-mask*] {**tcp** | **udp**} [*port-number* | **wsp** | **wsp-wtp** | **wsp-wtls** | **wsp-wtp-wtls**] [**service** *service-name*]

> **no virtual**

**Syntax Description**

| | |
|---|---|
| *ip-address* | IP address for this virtual server instance, used by clients to connect to the server farm. |
| *network-mask* | (Optional) Specifies the IP network mask for transparent webcache load balancing. The default is 0.0.0.0 (all subnets). |
| **tcp** | Performs load balancing for only TCP connections. |
| **udp** | Performs load balancing for only UDP connections. |
| *port-number* | (Optional) IOS SLB virtual port (the TCP or UDP port number or port name). If specified, only the connections for the specified port on the server are load-balanced. The ports and the valid name or number for the *port-number* argument are as follows:<br><br>• Domain Name System: **dns   53**<br>• File Transfer Protocol: **ftp   21**<br>• GPRS Tunneling Protocol: **gtp   3386**<br>• HTTP over Secure Socket Layer: **https   443**<br>• Mapping of Airline Traffic over IP, Type A: **matip-a   350**<br>• Network News Transport Protocol: **nntp   119**<br>• Post Office Protocol v2: **pop2   109**<br>• Post Office Protocol v3: **pop3   110**<br>• Simple Mail Transport Protocol: **smtp   25**<br>• Telnet: **telnet   23**<br>• World Wide Web (HTTP): **www   80**<br><br>Specify a port number of **0** to configure an all-port virtual server (that is, a virtual server that accepts flows destined for all ports except GTP ports). |
| **wsp** | (Optional) Performs load balancing in connectionless WSP mode on port 9200. |
| **wsp-wtp** | (Optional) Performs load balancing in connection-oriented WSP mode on port 9201, and activates the WAP FSM, which monitors WSP and drives the session FSM accordingly. |
| **wsp-wtls** | (Optional) Performs load balancing in connectionless secure WSP mode on port 9202. |
| **wsp-wtp-wtls** | (Optional) Performs load balancing in connection-oriented secure WSP mode on port 9203. |

| service | (Optional) Couples connections associated with a given service, such as HTTP or Telnet, so all related connections from the same client use the same real server. |
|---|---|
| *service-name* | (Optional) Type of connection coupling: |
| | • **ftp**—Couples FTP data connections with the control session that created them. |
| | • **gtp**—Allows load-balancing decisions to be made using Layer 5 information. You can balance UDP flows without awareness of GTP by omitting **service gtp**. |
| | • **wsp-wtp**—Couples WSP data connections with the control session that created them. |

**Defaults**    No default behavior or values.

**Command Modes**    Virtual server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |
| 12.1(5a)E | The **wsp**, **wsp-wtp**, **wsp-wtls**, and **wsp-wtp-wtls** keywords were added. |
| 12.1(9)E | The **gtp** option was added as new value on the *service-name* argument. |

**Usage Guidelines**    The **no virtual** command is allowed only if the virtual server was removed from service by the **no inservice** command.

For some applications, it is not feasible to configure all the virtual server TCP or UDP port numbers for IOS SLB. To support such applications, you can configure IOS SLB virtual servers to accept flows destined for all ports (except GTP ports). To configure an all-port virtual server, specify a port number of **0**.

On Catalyst 6000 Family Switches, if you use FTP sessions with MNLB you must configure a port-bound virtual server bound to port 21 on the MNLB Services Manager (the LocalDirector).

In general, you should use port-bound virtual servers instead of all-port virtual servers. When you use all-port virtual servers, flows can be passed to servers for which no application port exists. When the servers reject these flows, IOS SLB might fail the servers and remove them from load balancing. To prevent this problem, use one of the following procedures:

- Configure one or more port-bound virtual servers.
- Configure access control lists on the IOS SLB ingress interface to permit only ports for supported applications on the real server.
- Configure **faildetect numconns 255 numclients 8** on the real server.

**Examples**  The following example specifies that the virtual server with the IP address *10.0.0.1* performs load balancing for *TCP* connections for the port named *www*. The virtual server processes HTTP requests.

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# virtual 10.0.0.1 tcp www
```

The following example specifies that the virtual server with the IP address *10.0.0.13* performs load balancing for *UDP* connections for all ports. The virtual server processes HTTP requests.

```
Router(config)# ip slb vserver PUBLIC_HTTP
Router(config-slb-vserver)# virtual 10.0.0.13 udp 0
```

**Related Commands**

| Command | Description |
|---|---|
| **ip slb vserver** | Identifies a virtual server. |
| **show ip slb vserver** | Displays information about the virtual servers defined to IOS SLB. |

# weight (firewall farm real server)

To specify a real server's capacity, relative to other real servers in the firewall farm, use the **weight** firewall farm real server configuration command. To restore the default weight value, use the **no** form of this command.

**weight** *weighting-value*

**no weight**

**Syntax Description**

| | |
|---|---|
| *weighting-value* | Weighting value to use for real server predictor algorithm. Valid values range from 1 to 255. The default weighting value is 8. |

**Defaults**

Weighting-value default: 8

**Command Modes**

Firewall farm real server configuration

**Command History**

| Release | Modification |
|---|---|
| 12.1(3a)E | This command was introduced. |

**Examples**

The following example specifies the relative weighting values of three real servers as 16, 8 (by default), and 24, respectively:

```
Router(config)# ip slb firewallfarm FIRE1
!-----First real server
Router(config-slb-fw)# real 10.10.1.1
!-----Assigned weight of 16
Router(config-slb-fw-real)# weight 16
!-----Enabled
Router(config-slb-fw-real)# inservice
Router(config-slb-fw-real)# exit
!-----Second real server
Router(config-slb-fw)# real 10.10.1.2
!-----Enabled with default weight
Router(config-slb-fw-real)# inservice
Router(config-slb-fw-real)# exit
!-----Third real server
Router(config-slb-fw)# real 10.10.1.3
!-----Assigned weight of 24, not enabled
Router(config-slb-fw-real)# weight 24
```

| Related Commands | Command | Description |
|---|---|---|
| | **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| | **show ip slb firewallfarm** | Displays information about the firewall farm configuration. |
| | **show ip slb reals** | Displays information about the real servers. |

# weight (server farm)

To specify a real server's capacity, relative to other real servers in the server farm, use the **weight** real server configuration command. To restore the default weight value, use the **no** form of this command.

**weight** *weighting-value*

**no weight**

## Syntax Description

| | |
|---|---|
| *weighting-value* | Weighting value to use for real server predictor algorithm. Valid values range from 1 to 255. The default weighting value is 8. |

## Defaults

Weighting-value default: 8

## Command Modes

Real server configuration

## Command History

| Release | Modification |
|---|---|
| 12.0(7)XE | This command was introduced. |

## Usage Guidelines

The static weights you define using this command are overridden by the weights calculated by DFP. If DFP is removed from the network, IOS SLB reverts to these static weights.

## Examples

The following example specifies the relative weighting values of three real servers as 16, 8 (by default), and 24, respectively:

```
Router(config)# ip slb serverfarm PUBLIC
!-----First real server
Router(config-slb-sfarm)# real 10.10.1.1
!-----Assigned weight of 16
Router(config-slb-real)# weight 16
!-----Enabled
Router(config-slb-real)# inservice
Router(config-slb-real)# exit
!-----Second real server
Router(config-slb-sfarm)# real 10.10.1.2
!-----Enabled with default weight
Router(config-slb-real)# inservice
Router(config-slb-real)# exit
!-----Third real server
Router(config-slb-sfarm)# real 10.10.1.3
!-----Assigned weight of 24, not enabled
Router(config-slb-real)# weight 24
```

| Related Commands | Command | Description |
|---|---|---|
| | **real (server farm)** | Identifies a real server by IP address and optional port number as a member of a server farm and enters real server configuration mode. |
| | **show ip slb serverfarms** | Displays information about the server farm configuration. |
| | **show ip slb reals** | Displays information about the real servers. |

# FAQ (Frequently Asked Questions)

The following questions and answers can help you troubleshoot IOS SLB, if you have problems.

| Question | Answer |
|---|---|
| Can I use IOS SLB to load balance clients and real servers that are on the same LAN or VLAN? | **NO!**<br><br>**IOS SLB does not support load balancing of flows between clients and real servers that are on the same LAN or VLAN. The packets being load balanced cannot enter and leave the load-balancing device on the same interface.** |
| Why is IOS SLB not marking my connections as **ESTABLISHED** even though I'm transferring data? | If you are using dispatched mode, make sure there are no alternate paths that allow outbound flows to bypass IOS SLB. Also, make sure the clients and real servers are not on the same IP subnet. |
| Why am I able to connect to real servers directly, but unable to connect to the virtual server? | Make sure that the virtual IP address is configured as a loopback in each of the real servers (if you are running in dispatched mode). |
| Why is IOS SLB not marking my real server as failed when I disconnect it from the network? | Tune the values for the **numclients**, **numconns**, and **delay** keywords.<br><br>If you have a very small client population (for example, in a test environment), the **numclients** keyword could be causing the problem. This parameter prevents IOS SLB from mistaking the failure of a small number of clients for the failure of a real server. |
| Why does IOS SLB show my real server as **INSERVICE** even though I have taken it down or physically disconnected it? | The **INSERVICE** and **OUTOFSERVICE** states indicate whether the network administrator *intends* for that real server to be used when it is operational. A real server that was **INSERVICE** but was removed from the selection list dynamically by IOS SLB as a result of automatic failure detection, is marked as **FAILED**. Use the **show ip slb reals detail** command to display these real server states.<br><br>Beginning with release 12.1(1)E, **INSERVICE** is changed to **OPERATIONAL**, to better reflect what is actually occurring. |
| Why is IOS SLB not balancing correctly? I am using dispatched mode, the servers are leaving sockets open, and I am seeing RSTs in response to a number of SYNs. Curiously, sometimes things work fine. | Enter the **show mls flow** command:<br><br>```Router# sh mls flow<br>current ip flowmask for unicast: full flow<br>current ipx flowmask for unicast: destination only```<br><br>The current IP flowmask must be **full flow**. If it is not, correct the problem using the **mls flow ip full** command:<br><br>```Router# config t<br>Enter configuration commands, one per line.<br>End with CNTL/Z.<br>Router(config)# mls flow ip full<br>Router(config)#``` |

| Question | Answer |
|---|---|
| How can I verify that IOS SLB sticky connections are working properly? | Use the following procedure:<br><br>1. Configure the sticky connections.<br><br>2. Start a client connection.<br><br>3. Enter the **show ip slb reals detail** and **show ip slb conns** commands.<br><br>4. Examine the real server connection counts. The real server whose count increased is the one to which the client connection is assigned.<br><br>5. Enter the **show ip slb sticky** command to display the sticky relationships that IOS SLB stored.<br><br>6. End the connection.<br><br>7. Ensure that the real server's connection count decreased.<br><br>8. Restart the connection, after waiting no longer than the sticky timeout value.<br><br>9. Enter the **show ip slb conns** command again.<br><br>10. Examine the real server connection counts again, and verify that the sticky connection is assigned to the same real server as before. |
| How can I verify that server failures are being detected correctly? | Use the following procedure:<br><br>1. Use a large client population. If the number of clients is very small, tune the **numclients** keyword on the **faildetect (real server)** command so that the servers are not displayed as **FAILED**.<br><br>2. Enter the **show ip slb reals detail** command to show the status of the real servers.<br><br>3. Examine the status and connection counts of the real servers.<br><br>    – Servers that failed show a status of **FAILED**, **TESTING**, or **READY_T0_TEST**, based on whether IOS SLB is checking that the server came back up when the command was sent.<br><br>    – When a real server fails, connections that are assigned but not established (no SYN or ACK is received) are reassigned to another real server on the first inbound SYN after the **reassign** threshold is met. However, any connections that were already established are forwarded to the same real server because, while it might not be accepting new connections, it might be servicing existing ones.<br><br>    – For weighted least connections, a real server that has just been placed in service starts slowly so that it is not overloaded with new connections. (See the "Slow Start" section on page 15 for more information.) Therefore, the connection counts displayed for a new real server show connections going to other real servers (despite the new real server's lower count). The connection counts also show "dummy connections" to the new real server, which IOS SLB uses to artificially inflate the connection counts for the real server during the slow start period. |

| Question | Answer |
|---|---|
| Does the **no inservice** command take a resource out of service immediately? | When you use the **no** form of the **inservice** command to remove a firewall, firewall farm, real server, or virtual server from service, the resource quiesces gracefully. No new connections are assigned, and existing connections are allowed to complete.<br><br>To stop all existing connections for an entire firewall farm or virtual server immediately, use the **clear ip slb** command. |
| I configured both IOS SLB and input ACLs on the same Catalyst 6000 Family Switch, and now I see TCAM Capacity Exceeded messages. Why? | If you configure IOS SLB and either input ACLs or firewall load balancing on the same Catalyst 6000 Family Switch, you can exceed the capacity of the TCAM on the Policy Feature Card (PFC). To correct the problem, use the **mls ip slb search wildcard rp** command to reduce the amount of TCAM space used by IOS SLB, but be aware that this command can result in a slight increase in route processor utilization. |

# Glossary

**active standby**—Redundancy scheme in which two IOS SLB devices can load-balance the same virtual IP address while at the same time acting as backups for each other. See also *active standby*, *stateful backup*.

**bearer network**—Network that carries messages of a transport-layer protocol, and ultimately also of the session-layer protocols, between physical devices. A single session can use more than one bearer network.

**CASA**—Content Aware Services Architecture. CASA is a protocol designed to allow network appliances to selectively control the flow of IP packets through a router, switch, or other network device.

**client NAT**—Translation scheme in which the client IP address is replaced with an IP address associated with one of a group of load-balancing devices, resulting in outbound flows being routed to the correct device. See also *directed mode*, *server NAT*.

**client subsystem**—Users, such as IOS SLB or GPRS, of the DFP agent function.

**cluster**—Set of computer systems that are connected through multisystem hardware or software to supply services traditionally provided by a single system. This arrangement provides higher availability and better scalability of the system.

**content-aware networking**—Networking strategy that enables content to be dynamically distributed. Because content can be dynamically cached, it can be located at any given place at any given time and distributed between the servers and the location of the webcache. Cisco has developed the ContentFlow architecture and the DFP to enable networks to provide content-aware networking services.

**ContentFlow architecture**—Cisco's content-aware networking architecture that describes message flows and actions in a distributed environment.

**DFP**—Dynamic Feedback Protocol. Allows host agents to dynamically report the change in status of the host systems providing a virtual service. The status reported is a relative weight that specifies a host server's capacity to perform work.

**DFP agent**—Object in a load-balanced environment that dynamically reports changes in status of the host systems that provide a virtual service. The status reported is a relative weight that specifies a host server's capacity to perform work. See also *DFP manager*.

**DFP manager**—Object in a load-balanced environment that collects status reports from DFP agents. See also *DFP agent*.

**directed mode**—Session redirection mode in which the virtual server can be assigned an IP address that is not known to any of the real servers. IOS SLB translates packets exchanged between a client and real server, translating the virtual server IP address to a real server IP address through NAT. See also *dispatched mode*, *NAT*.

**dispatched mode**—Session redirection mode in which the virtual server address is known to the real servers. The virtual server IP address must be configured as a loopback address, or secondary IP address, on each of the real servers. IOS SLB redirects packets to the real servers at the media access control (MAC) layer. Since the virtual server IP address is not modified in dispatched mode, the real servers must be Layer 2-adjacent to IOS SLB, or intervening routers might not be able to route to the chosen real server. See also *directed mode*.

**Dynamic Feedback Protocol**—See *DFP*.

**firewall**—Router or access server, or several routers or access servers, designated as a buffer between any connected public networks and a private network. A firewall router uses access lists and other methods to ensure the security of the private network.

**firewall farm**—Group of firewalls.

**firewall load balancing**—Load-balancing scheme in which the network administrator configures a group of firewalls into a firewall farm. When a client initiates a connection, IOS SLB chooses a firewall for the connection based on a hash algorithm.

**General Packet Radio Service**—See *GPRS*.

**GPRS Tunneling Protocol**—See *GTP*.

**gateway GPRS support node**—See *GGSN*.

**Global System for Mobile Communications**—See *GSM*.

**GGSN**—Gateway GPRS Support Node. A GPRS network entity that serves as the mobile wireless gateway between an SGSN and PDNs. The GGSN allows mobile wireless users to access PDNs. See also *GPRS*, *GSM*, *GTP*, *SGSN*.

**GPRS**—General Packet Radio Service. An ETSI standard that defines the implementation of packet data services on a GSM network. See also *GGSN*, *GSM*, *GTP*, *SGSN*.

**GSM**—Global System for Mobile Communications. A second-generation (2G) mobile wireless networking standard defined by ETSI and deployed widely throughout the world. See also *GGSN*, *GPRS*, *GTP*, *SGSN*.

**GTP**—GPRS Tunneling Protocol. Protocol that handles the flow of user packet data and signaling information between the SGSN and GGSN in a GPRS network. GTP is defined on both the Gn and Gp interfaces of a GPRS network. See also *GGSN*, *GPRS*, *GSM*, *SGSN*.

**HSRP**—Hot Standby Router Protocol. Provides high network availability and transparent network topology changes. HSRP creates a Hot Standby router group with a lead router that services all packets sent to the Hot Standby address. The lead router is monitored by other routers in the group, and if it fails, one of these standby routers inherits the lead position and the Hot Standby group address. See also *redundancy*, *stateful backup*, *stateless backup*.

**HTTP redirect load balancing**—Load-balancing scheme in which all HTTP requests that belong to the same transaction are directed to the same real server.

**IOS SLB**—IOS Server Load Balancing. Load-balancing function in which the network administrator defines a virtual server that represents a group of real servers in a cluster of network servers known as a server farm. When a client initiates a connection to the virtual server, IOS SLB chooses a real server for the connection based on a configured load-balancing algorithm.

**load balancing**—Spreading user requests among available servers within a cluster of servers, based on a variety of algorithms.

**MD5**—Message Digest Algorithm Version 5. Neighbor router authentication scheme used to ensure reliability and security when routing updates are exchanged between neighbor routers.

**Message Digest Algorithm Version 5**—See *MD5*.

**NAT**—Network Address Translation. Modification of one or more of the following fields in an IP packet: source IP address, destination IP address, source TCP/UDP port, destination TCP/UDP port. See also *client NAT*, *server NAT*.

**NetFlow switching**—High-performance network-layer switching path that captures as part of its switching function a rich set of traffic statistics including user, protocol, port, and type of service information.

**Network Address Translation**—See *NAT*.

**packet data protocol**—See *PDP*.

**PDP**—Packet data protocol. Network protocol used by external packet data networks that communicate with a GPRS network. IP is an example of a PDP supported by GPRS.

**port-bound**—Server configuration scheme in which a virtual server IP address represents one set of real servers for one service, such as Hypertext Transfer Protocol (HTTP), and a different set of real servers for another service, such as Telnet.

**real server**—The specification of a physical server associated with a virtual server. The specification includes the real server's IP address and an optional weight to be used by the virtual server predictor.

**redundancy**—The duplication of devices, services, or connections so that, in the event of a failure, the redundant devices, services, or connections can perform the work of those that failed. See also *stateful backup*, *stateless backup*.

**round robin**—See *weighted round robin*.

**Secure Socket Layer**—See *SSL*.

**server cluster**—See *server farm*.

**server farm**—Also called a server cluster. Group of real servers that provide various applications and services.

**Server Load Balancing**—See *IOS SLB*.

**server NAT**—Translation scheme in which the virtual server IP address is replaced with the real server IP address (and vice versa), allowing servers to be many hops away from the load-balancing device, and enabling intervening routers to route to them without requiring tunnelling. See also *client NAT*, *directed mode*, *server port translation*.

**server port translation**—Translation scheme in which the virtual server port number is replaced with the real server port number (and vice versa), allowing servers to be many hops away from the load-balancing device, and enabling intervening routers to route to them without requiring tunnelling. See also *client NAT*, *directed mode*, *server NAT*.

**services manager**—Functionality built into IOS SLB that makes load-balancing decisions based on application availability, server capacity, and load distribution algorithms such as weighted round robin or weighted least connections, or the DFP. The services manager determines a real server for the packet flow using load balancing and server/application feedback.

**serving GPRS support node**—See *SGSN*.

**SGSN**—serving GPRS support node. A GPRS network entity that sends data to and receives data from mobile stations, and maintains information about the location of a mobile station. The SGSN communicates between the mobile station and the GGSN; the GGSN provides access to the data network. See also *GGSN*, *GPRS*, *GSM*, *GTP*.

**SLB**—See *IOS SLB*.

**SSL**—Secure Socket Layer. Encryption technology for the web used to provide secure transactions such as the transmission of credit card numbers for e-commerce.

**stateful backup**—Redundancy scheme that enables IOS SLB to incrementally backup its load-balancing decisions, or "keep state," between primary and backup switches. See also *active standby*, *stateless backup*.

**stateless backup**—Redundancy scheme that provides high network availability by routing IP flows from hosts on Ethernet networks without relying on the availability of a single Layer 3 switch. See also *active standby*, *stateful backup*.

**sticky connections**—Load-balancing scheme in which new connections from a client IP address or subnet are assigned to the same real server (for server load balancing) or firewall (for firewall load balancing) as were previous connections from that address or subnet.

**virtual server**—Presents a single address that represents an application server farm to clients.

**WAP**—Wireless Application Protocol. Suite of protocols used to deliver services to wireless devices.

**weighted least connection**—Load-balancing algorithm in which the next real server chosen for a new connection to the virtual server is the server with the fewest active connections. Each real server is assigned a weight, $n$, that represents its capacity to handle connections, as compared to the other real servers associated with the virtual server. The server with the fewest connections is based on the number of active connections on each server, and on the relative capacity of each server. The capacity of a given real server is calculated as the assigned weight of that server divided by the sum of the assigned weights of all of the real servers associated with that virtual server, or $n_1/(n_1+n_2+n_3...)$.

**weighted round robin**—Load-balancing algorithm in which the real server used for a new connection to the virtual server is chosen in a circular fashion. Each real server is assigned a weight, $n$, that represents its capacity to handle connections, as compared to the other real servers associated with the virtual server. New connections are assigned to a given real server $n$ times before the next real server in the list is chosen.

**Wireless Application Protocol**—See *WAP*.

**Wireless Session Protocol**—See *WSP*.

**Wireless Transaction Protocol**—See *WTP*.

**Wireless Transport Security Layer**—See *WTLS*.

**workload agents**—Value-added software components developed for specific platforms by third-party developers. Workload agents run on server platforms or on platforms that manage server farms. Workload agents deliver server and application information to the services manager. This information enables the services manager to make optimum server selection.

**WSP**—Wireless Session Protocol. Session-layer protocol of the WAP suite.

**WTLS**—Wireless Transport Security Layer. Layer that provides security between WAP clients and WAP gateways.

**WTP**—Wireless Transaction Protocol. Transaction-layer protocol of the WAP suite.