CHAPTER 6

Configuring Health Monitoring

This chapter describes how to configure the health monitoring on the CSM and contains these sections:

Configuring Probes for Health Monitoring

Configuring probes to the real servers allows you to determine if the real servers are operating correctly. A real server's health is categorized as follows:

- Active—The real server responds appropriately.
- Suspect—The real server is unreachable or returns an invalid response. The probes are retried.
- Failed—The real server fails to reply after a specified number of consecutive retries. You are notified and the CSM adjusts incoming connections accordingly. Probes continue to a failed server until the server becomes active again.

The CSM supports probes used to monitor real servers. Configuring a probe involves the following:

- Entering the probe submode
- Naming the probe
- Specifying the probe type

The CSM supports a variety of probe types that monitor real servers, including FTP, DNS, or HTTP.

Note By default, no probes are configured on the CSM.

To set up a probe, you must configure it by naming the probe and specifying the probe type while in probe submode.

After configuring a probe, you must associate it with a server farm for the probe to take effect. All servers in the server farm receive probes of the probe types that are associated with that server farm. You can associate one or more probe types with a server farm.

Note Do not specify a port number when you configure a probe. The probe inherits the port number from either the real server (if a port number was assigned when the real server was configured) or from the virtual server.

After you configure a probe, associate single or multiple probes with a server farm. All servers in the server farm receive probes of the probe types that are associated with that pool.

Note If you associate a probe of a particular type with a server farm containing real servers that are not running the corresponding service, the real servers send error messages when they receive a probe of that type. This action causes the CSM to place the real server in a failed state and disable the real server from the server farm.

To specify a probe type and name, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* `[`**`http`** `|` **`icmp`** `|` **`telnet`** `|` **`tcp`** `|` **`ftp`** `|` **`smtp`** `|` **`dns`**`]`	Specifies a probe type and a name[1] [2] [3]. • *probe-name* is the name of the probe being configured; it has a character string of up to 15 characters. • **http** creates an HTTP probe with a default configuration. • **icmp** creates an ICMP probe with a default configuration. • **telnet** creates a Telnet probe with a default configuration. • **tcp** creates a TCP probe with a default configuration. • **ftp** creates an FTP probe with a default configuration. • **smtp** creates an SMTP probe with a default configuration. • **dns** creates a DNS probe with a default configuration.
Step 2	`Router#` **`show module csm`** *`slot`* **`probe`**	Displays all probes and their configuration.

[1] The **no** form of this command removes the probe type from the configuration.

[2] An alert is displayed when the combination of the number of probes configured and the frequency (interval) of probes impacts the CSM's performance. If you receive this alert, change the number of probes and the probe frequency until the CSM performance is improved.

[3] Inband health monitoring provides a more scalable solution if you are receiving performance alerts.

Note When you specify a probe name and type, it is initially configured with the default values. Enter the probe configuration commands to change the default configuration.

This example shows how to configure a probe:

```
Router(config-module-csm)# probe probe1 tcp
Router(config-slb-probe-tcp)# interval 120
Router(config-slb-probe-tcp)# retries 3
Router(config-slb-probe-tcp)# failed 300
Router(config-slb-probe-tcp)# open 10
Router(config-slb-probe-tcp)# serverfarm sf4
Router(config-slb-sfarm)# real 10.1.0.105
Router(config-slb-real)# inservice
Router(config-slb-real)# probe probe1
Router(config-slb-sfarm)# vserver vs4
Router(config-slb-vserver)# virtual 10.1.0.84 tcp 80
Router(config-slb-vserver)# serverfarm sf4
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# end
```

Note There are two different timeout values: open and receive. The open timeout specifies how many seconds to wait for the connection to open (that is, how many seconds to wait for SYN ACK after sending SYN). The receive timeout specifies how many seconds to wait for data to be received (that is, how many seconds to wait for an HTTP reply after sending a GET/HHEAD request). Because TCP probes close as soon as they open without sending any data, the receive timeout is not used.

Probe Configuration Commands

These commands are common to all probe types:

Command	Purpose

`Router(config-slb-probe)#`
`interval` *`seconds`*
	Sets the interval between probes in seconds (from the end of the previous probe to the beginning of the next probe) when the server is healthy[1] [2] [3].
	Range = 2–65535 seconds
	Default = 120 seconds

`Router(config-slb-probe)#`
`retries` *`retry-count`*
	Sets the number of failed probes that are allowed before marking the server as failed[1].
	Range = 0–65535
	Default = 3

`Router(config-slb-probe)#`
`failed` *`failed-interval`*
	Sets the time between health checks when the server has been marked as filed. The time is in seconds[1].
	Range = 5–65535
	Default = 300 seconds

`Router(config-slb-probe)#` **`open`**
`open-timeout`
	Sets the maximum time to wait for a TCP connection. This command is not used for any non-TCP health checks (ICMP or DNS[1]).
	Range = 1–65535
	Default = 10 seconds

[1] The **no** form of this command restores the defaults.

[2] An alert is displayed when the combination of the number of probes configured and the frequency (interval) of probes impacts the CSM's performance. If you receive this alert, change the number of probes and the probe frequency until the CSM performance is improved.

[3] Inband health monitoring provides a more scalable solution if you are receiving performance alerts.

Configuring an HTTP Probe

An HTTP probe establishes an HTTP connection to a real server and then sends an HTTP request and verifies the response. The **probe** *probe-name* **http** command places the user in HTTP probe configuration submode.

To configure an HTTP probe, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* ***`http`***	Configures an HTTP probe and enters the HTTP probe submode[1].
Step 2	`Router(config-slb-probe-http)#` **`credentials`** *`username`* `[`*`password`*`]`	Configures basic authentication values for the HTTP SLB probe[1].
Step 3	`Router(config-slb-probe-http)#` **`expect status`** *`min-number`* `[`*`max-number`*`]`	Configures a status code to expect from the HTTP probe. You can configure multiple status ranges by entering one **expect status** command at a time[1]. *min-number*—If you do not specify a *max-number*, this number is taken as a single status code. If you specify a *max-number*, this number is taken as the minimum status code of a range. *max-number*—The maximum status code in a range. The default range is 0–999. (Any response from the server is considered valid.) **Note** If no maximum is specified, this command takes a single number (min-number). If you specify both a min-number and a max-number, it takes the range of numbers.
Step 4	`Router(config-slb-probe-http)#` **`header`** *`field-name`* `[`*`field-value`*`]`	Configures a header field for the HTTP probe. Multiple header fields may be specified[1].
Step 5	`Router(config-slb-probe-http)#` **`request`** `[`**`method`** `[`**`get`**`|`**`head`**`]] [`**`url`** *`path`*`]`	Configures the request method used by an HTTP probe[1]: • **get**—The HTTP **get** request method directs the server to get this page • **head**—The HTTP **head** request method directs the server to get only the header for this page • **url**—A character string of up to 1275 characters specifies the URL path; the default path is "/" **Note** The CSM supports only the **get** and **head** request methods; it does not support the **post** and other methods. The default method is **get**.

[1] The **no** form of this command restores the defaults.

Configuring an ICMP Probe

An ICMP probe sends an ICMP echo (for example, ping) to the real server. The **probe icmp** command enters the ICMP probe configuration mode. All the common **probe** commands are supported except the **open** command, which is ignored.

To configure an ICMP probe, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* **`icmp`**	Configures an ICMP probe and enters the ICMP probe submode[1].
Step 2	`Router(config-slb-probe-icmp)#` **`interval`**	Configures the intervals to wait between probes of a failed server and between probes.
Step 3	`Router(config-slb-probe-icmp)#` **`receive`**	Specifies the time to make a TCP connection to receive a reply from the server.
Step 4	`Router(config-slb-probe-icmp)#` **`retries`**	Limits the number of retries before considering the server as failed.

[1] The **no** form of this command restores the defaults.

Configuring a TCP Probe

A TCP probe establishes and removes connections. The **probe tcp** command enters the TCP probe configuration mode. All the common **probe** commands are supported.

To configure a TCP probe, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* **`tcp`**	Configures a TCP probe and enters the TCP probe submode[1].
Step 2	`Router(config-slb-probe-icmp)#` **`interval`**	Configures the intervals to wait between probes of a failed server and between probes.
Step 3	`Router(config-slb-probe-icmp)#` **`retries`**	Limits the number of retries before considering the server as failed.

[1] The **no** form of this command restores the defaults.

Configuring FTP, SMTP, and Telnet Probes

An FTP, SMTP, or Telnet probe establishes a connection to the real server and verifies that a greeting from the application was received. The **probe** (**ftp, smtp,** or **telnet**) command enters the corresponding probe configuration mode. All the **probe** common options are supported. Multiple status ranges are supported, one command at a time.

To configure a status code to expect from the FTP, SMTP, or Telnet probe, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* `[`**`ftp`** `|` **`smtp`** `|` **`telnet`**`]`	Configures an FTP, SMTP, or Telnet probe and enters the FTP, SMTP, or Telnet probe submode[1].
Step 2	`Router(config-slb-probe-icmp)#` **`interval`**	Configures the intervals to wait between probes of a failed server and between probes.

	Command	Purpose
Step 3	`Router(config-slb-probe-icmp)#` **`receive`**	Specifies the time to make a TCP connection to receive a reply from the server
Step 4	`Router(config-slb-probe-icmp)#` **`retries`**	Limits the number of retries before considering the server as failed.

1. The **no** form of this command restores the defaults.

Specifying the DNS Resolve Request

A DNS probe sends a domain name resolve request to the real server and verifies the returned IP address. The **probe dns** command places the user in DNS probe configuration submode. All the probe common options are supported except **open,** which is ignored.

To specify the domain name resolve request, perform this task:

	Command	Purpose
Step 1	`Router(config-module-csm)#` **`probe`** *`probe-name`* **`dns`**	Configures a DNS probe and enters the tcp probe submode[1].
Step 2	`Router(config-slb-probe-dns)# [`**`failed`**` | `**`interval`**` | `**`retries`**` | `**`receive`**`]`	Configures the times to wait between probes to make a DNS connection, to receive a reply from the server, and to limit the number of retries before considering the real server as failed.

1. The **no** form of this command restores the defaults.

Configuring Route Health Injection

These sections describe the route health injection (RHI):

- Understanding RHI, page 6-6
- Configuring RHI for Virtual Servers, page 6-8

Understanding RHI

These sections describe the RHI:

- RHI Overview, page 6-7
- Routing to VIP Addresses Without RHI, page 6-7
- Routing to VIP Addresses with RHI, page 6-8
- Understanding How the CSM Determines VIP Availability, page 6-8
- Understanding Propagation of VIP Availability Information, page 6-8

RHI Overview

RHI allows the CSM to advertise the availability of a VIP address throughout the network. Multiple CSM devices with identical VIP addresses and services can exist throughout the network. One CSM can override the server load-balancing services over the other devices if the services are no longer available on the other devices. One CSM also can provide the services because it is logically closer to the client systems than other server load-balancing devices.

Note RHI is restricted to intranets because the CSM advertises the VIP address as a host route and most routers do not propagate the host-route information to the Internet.

To enable RHI, configure the CSM to do the following:

- Probe real servers and identify available virtual servers and VIP addresses
- Advertise accurate VIP address availability information to the MSFC whenever a change occurs

  Note On power-up with RHI enabled, the CSM sends a message to the MSFC as each VIP address becomes available.

The MSFC periodically propagates the VIP address availability information that RHI provides.

Note RHI is normally restricted to intranets; for security reasons, most routers do not propagate host-route information to the Internet.

Routing to VIP Addresses Without RHI

Without RHI, traffic reaches the VIP address by following a route to the client VLAN to which the VIP address belongs. When the CSM powers on, the MSFC creates routes to client VLANs in its routing table and shares this route information with other routers. To reach the VIP, the client systems rely on the router to send the requests to the network subnet address where the individual VIP address lives.

If the subnet or segment is reachable but the virtual servers on the CSM at this location are not operating, the requests fail. Other CSM devices can be at different locations. However, the routers only send the requests based on the logical distance to the subnet.

Without RHI, traffic is sent to the VIP address without any verification that the VIP address is available. The real servers attached to the VIP might not be active.

Note By default, the CSM will not advertise the configured VIP addresses.

Routing to VIP Addresses with RHI

With RHI, the CSM sends advertisements to the MSFC when VIP addresses become available and withdraws advertisements for VIP addresses that are no longer available. The router looks in the routing table to find the path information it needs to send the request from the client to the VIP address. When the RHI feature is turned on, the advertised VIP address information is the most specific match. The request for the client is sent through the path where it reaches the CSM with active VIP services.

When multiple instances of a VIP address exist, a client router receives the information it needs (availability and hop count) for each instance of a VIP address, allowing it to determine the best available route to that VIP address. The router picks the path where the CSM is logically closer to the client system.

Note With RHI, you must also configure probes because the CSM determines if it can reach a given VIP address by probing all the real servers that serve its content. After determining if it can reach a VIP address, the CSM shares this availability information with the MSFC. The MSFC, in turn, propagates this VIP availability information to the rest of the intranet.

Understanding How the CSM Determines VIP Availability

For the CSM to determine if a VIP is available, you must configure a probe (HTTP, ICMP, Telnet, TCP, FTP, SMTP, or DNS) and associate it with a server farm. With probes configured, the CSM performs these checks:

- Probes all real servers on all server farms configured for probing
- Identifies server farms that are reachable (have at least one reachable real server)
- Identifies virtual servers that are reachable (have at least one reachable server farm)
- Identifies VIPs that are reachable (have at least one reachable virtual server)

Understanding Propagation of VIP Availability Information

With RHI, the CSM sends advertise messages to the MSFC containing the available VIP addresses. The MSFC adds an entry in its routing table for each VIP address it receives from the CSM. The routing protocol running on the MSFC sends routing table updates to other routers. When a VIP address becomes unavailable, its route is no longer advertised, the entry times out, and the routing protocol propagates the change.

Note For RHI to work on the CSM, the MSFC in the chassis in which the CSM resides must run Cisco IOS Release 12.1.7(E) or later and must be configured as the client-side router.

Configuring RHI for Virtual Servers

To configure RHI for the virtual servers, follow these steps:

Step 1 Verify that you have configured VLANs (see the "Configuring VLANs" section on page 3-8).

Step 2 Associate the probe with a server farm (see the "Configuring Server Farms" section on page 3-10).

Step 3 Configure the CSM to probe real servers (see the "Configuring Probes for Health Monitoring" section on page 6-1).

Step 4 Enter the **advertise active** SLB virtual server command to enable RHI for each virtual server:

```
Router(config-module-csm)# vserver virtual_server_name
Router(config-slb-vserver)# advertise active
```

This example shows how to enable RHI for the virtual server named vserver1:

```
Router(config-module-csm)# vserver vserver1
Router(config-slb-vserver)# advertise active
```

Configuring Inband Health Monitoring

These sections describe inband health monitoring:

- Understanding Inband Health Monitoring, page 6-9
- Configuring Inband Health Monitoring, page 6-9

Understanding Inband Health Monitoring

To efficiently balance connections, the CSM must continuously monitor the health of all real servers in its configuration. The inband health monitoring feature is configured for each server farm to monitor the health of the servers. The parameters configured per server farm are then applied to each real server in that server farm. You can configure the number of abnormal end sessions that occur before the system considers the real server unreachable and you also can specify a time to wait before a real server is reintroduced into the server farm and a connection attempt is made.

This feature works with health probes. If health probes and inband health monitoring are both configured on a particular server, both sets of health checks are required to keep a real server in service within a server farm. If either health checking feature finds a server out of service, the server will not be selected by the CSM for load balancing.

Configuring Inband Health Monitoring

To configure inband health monitoring, follow these steps:

Step 1 Verify that you have configured server farms (see the "Configuring Server Farms" section on page 3-10).

Step 2 Enter the **serverfarm** submode command to enable inband health monitoring for each server farm:

```
Router(config-module-csm)# serverfarm serverfarm-name
Router(config-slb-sfarm)# health retries count failed seconds
```

Note Retries are the number of abnormal end sessions that the CSM will tolerate before removing a real server from service. The failed time is the number of seconds which the CSM waits before reattempting a connection to a real server that was removed from service by inband health checking.

This example shows how to enable inband health monitoring for a server farm named geo:

```
Router(config-module-csm)# serverfarm geo
Router(config-slb-sfarm)# health retries 43 failed 160
```

Configuring HTTP Return Code Checking

These sections describe HTTP retrn code checking:

- Understanding HTTP Return Code Checking, page 6-10
- Configuring HTTP Return Code Checking, page 6-11

Understanding HTTP Return Code Checking

The return error code checking (return code parsing) feature is used to indicate when a server is not returning web pages correctly. This feature extends the capability of CSM to inspect packets, parse the HTML return codes, and act upon the return codes returned by the server.

After receiving an HTTP request from the CSM, the server responds with an HTTP return code. The CSM can use the HTTP return error codes to determine the availability of the server. The CSM can be configured to take a server out of use in response to receiving specific return codes.

A list of predefined codes (100 through 599) are in RFC 2616. For return code checking, some codes are more usable than others. For example, a return code of 404 is defined as a URL not found, which may be the result of the user entering the URL incorrectly. Error code 404 also might mean that the web server has a hardware problem, such as a defective disk drive preventing the server from finding the data requested. In this case, the web server is still alive, but the server cannot send the requested data because of the defective disk drive. Because of the inability of the server to return the data, you do not want future requests for data sent to this server. To determine the error codes you want to use for return code checking, refer to RFC 2616.

When HTTP return code checking is configured, the CSM monitors HTTP responses from all balanced HTTP connections and logs the occurrence of the return code for each real server. The CSM stores return code counts. When a threshold for a return code is reached, the CSM may send syslog messages or remove the server from service.

A default action, counting return codes, syslog messaging, or removing the real server from service or a set of these actions can be applied to a server farm. You also can bind a single virtual group to multiple server farms allowing you to reuse a single return code server farm policy on multiple server farms.

Note Configuring HTTP return code checking on a virtual server impacts the performance of that virtual server. Once return code parsing is enabled, all HTTP server responses must be parsed for return codes.

Configuring HTTP Return Code Checking

Configuring return error code checking involves configuring the attributes of a server farm and associating it with a return code map.

To configure the return code checking follow these steps:

Step 1 Verify that you have configured HTTP virtual servers (see the "Configuring Virtual Servers" section on page 3-17).

Step 2 Enter the map return code command to enable return code mapping and enter the return code map submode:

```
Router(config-module-csm)# map name retcode
```

Step 3 Configure the return code parsing:

```
Router(config-slb-map-retcode)# match protocol http retcode min max action [count | log |
remove] threshold [reset seconds]
```

You can set up as many matches as you want in the map.

Step 4 Assign a return code map to a server farm:

```
Router(config-slb-sfarm)# retcode-map name
```

This example shows how to enable return error code checking:

```
Router(config-module-csm)#map httpcodes retcode
Route(config-slb-map-retcode)#match protocol http retcode 401 401 action log 5 reset 120
Route(config-slb-map-retcode)#match protocol http retcode 402 415 action count
Route(config-slb-map-retcode)#match protocol http retcode 500 500 action remove 3 reset 0
Route(config-slb-map-retcode)#match protocol http retcode 503 503 action remove 3 reset 0
Route(config-slb-map-retcode)#exit
Router(config-module-csm)#serverfarm farm1
Router(config-slb-sfarm)#retcode-map httpcodes
Router(config-slb-sfarm)#exit
Router(config-module-csm)#end
```