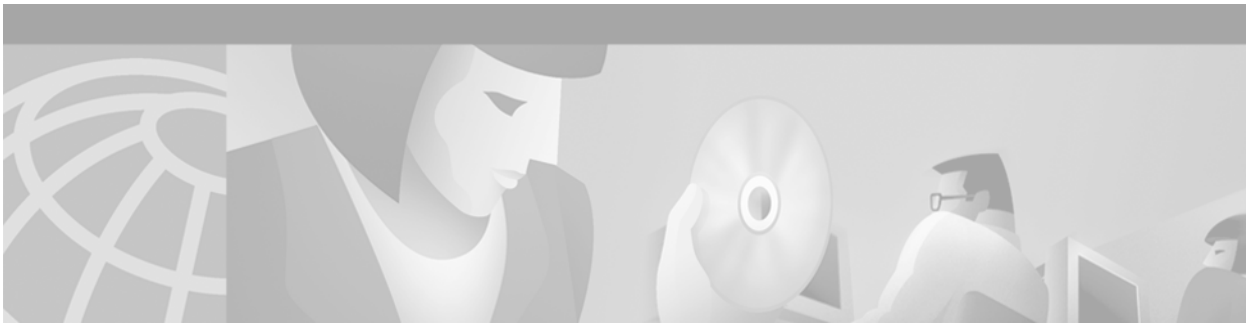

Release Notes for Cisco LocalDirector Version 4.1.1

October 2000

Note The most current Cisco documentation for released products is available on Cisco.com at http://www.cisco.com. The online documents may contain updates and modifications made after the hardcopy documents were printed.

Contents

These release notes describe the following topics:

CISCO SYSTEMS

Corporate Headquarters: Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

78-12579-01

New Features in Cisco LocalDirector Software Version 4.1.1

The following is a list of new software features supported by the Cisco LocalDirector software, version 4.1.1:

- HTTP Redirection
- Content Load Balancing
- Integrated Probe for DNS

HTTP Redirection Overview

HTTP redirection is a reliable method of implementing persistent or sticky connections. HTTP redirection also allows LocalDirector to perform effective load balancing for secure socket layer (SSL) and non-SSL connections, as well as for connections to an Internet Service Provider (ISP) that pass through a proxy server.

In a typical load-balancing environment, traffic comes from various client networks across the Internet to the virtual address on LocalDirector. LocalDirector then load balances this traffic between real servers.

A client must be connected to the same real server throughout a session for an HTTP server to maintain state on the connection. The *sticky* feature allows LocalDirector to load balance each client connection in a session to the same real server. Once sticky is applied to a real server for a particular virtual server, all client connections to the virtual server are directed to the same real server.

The current implementation of sticky uses either the client IP address, a unique cookie value, or the SSL session ID to identify the client. Each of these implementations has drawbacks.

If you apply sticky based on the client IP address, it is not effective for load-balancing techniques when connections to an ISP pass through a proxy server. In this scenario, sticky directs *all* client connections originating from behind the proxy server to the same real server, and this quickly creates a load imbalance. Also, persistence is broken if the ISP has multiple proxy servers behind a server load balancer (SLB). The SLB changes the client's address to different proxy addresses as attempts are made to load balance client requests. It is common practice for site security to configure the proxy server directly in front of LocalDirector.

The use of *cookie-sticky* provides a mechanism for inserting a unique key for each user of a virtual server. This feature is only used in nonsecure socket layer (non-SSL) connections. The cookie-sticky feature provides a means to solve the proxy-server problem and to give better load distribution at the server site.

The SSL session ID is effective only when the server is running SSL. Most configurations today only use SSL when necessary due to the heavy processing load required to maintain SSL connections. In addition, a problem has been discovered with the Microsoft Internet Explorer (IE) web browser. It has been confirmed that the browser randomly changes the current SSL session ID to 0. This means the client can no longer be identified throughout a session. This anomaly can occur more frequently as the number of Microsoft Internet Explorer users increases.

Many Web sites want to support mixed SSL and non–SSL connections. An example of mixed SSL and non–SSL connections is the Web site of an online vendor who provides a *shopping cart* for users to select items for purchase while browsing all the items for sale. The contents of the shopping cart are not considered confidential, and most online vendor administrators establish this area of the Web site as a non-SSL connection. However, as soon as a client is ready to purchase items, (often referred to as checkout time), the online vendor wants to be able to switch to a secure SSL connection to collect

confidential information (such as a credit card number and expiration date) for the purchase. At this point during the client's session, the online vendor does not want to lose the client's need to mix non-SSL and SSL connections.

The most effective solution based on customer needs is HTTP redirection. An added benefit of HTTP redirection is that this functionality can be introduced into existing customer environments without introducing new problems.

Configuration Tasks

The following tasks must be performed to configure the HTTP redirection feature:

- Create a virtual server
- Create URLs
- Bind the URLs to a virtual server
- Create a direct IP address for each server
- Create a link for each direct IP address
- Create a back up for each direct IP address

Configuring HTTP Redirection

The following configuration example uses one LocalDirector and two Web servers. LocalDirector has a virtual server with the IP address 1.1.1.1 and the two real servers have addresses 1.1.1.10 and 1.1.1.11. The real servers and the virtual server are registered in DNS as follows:

```
1.1.1.1 www.acme.com
1.1.1.10 coyote.acme.com
1.1.1.11 roadrunner.acme.com
```

To use HTTP redirection to load balance between these real machines, follow these steps:

	Command	Purpose
Step 1	`LocalDirector(config)# virtual 1.1.1.1:80:0:tcp is`	Creates a virtual server to accept connections from the network.

Step 2	`LocalDirector(config-if)# url coyote http://coyote.acme.com/%p`
	`LocalDirector(config)# url roadrunner http://roadrunner.acme.com/%p`
	Note The %p macro is used here so that LocalDirector includes the incoming path on the redirect it sends out. For instance if the client requests http://www.acme.com/main.html, the redirect would be for http://coyote.acme.com/main.html, if %p was not used the redirect would be for http://coyote.acme.com.
		Defines a Uniform Resource Locator (URL). Allows you to create a short identifier to replace a long URL string.

Step 3	`LocalDirector(config)# direct-ip 1.1.1.10:80:0:tcp is` `LocalDirector(config)# direct-ip 1.1.1.11:80:0:tcp is`	Creates the same direct IP address for the virtual server and the real servers in LocalDirector[1].
Step 4	`LocalDirector(config)# link coyote 1.1.1.10:80:0:tcp` `LocalDirector(config)# link roadrunner 1.1.1.11:80:0:tcp`	Links each direct IP address.[2]
Step 5	`LocalDirector(config)# backup 1.1.1.10:80:0:tcp 1.1.1.1:80:0:tcp` `LocalDirector(config)# backup 1.1.1.11:80:0:tcp 1.1.1.1:80:0:tcp`	Creates a backup for both direct IP addresses.[3]

1. The **direct-ip** command creates the same direct IP address for the virtual server and the real servers in LocalDirector and binds them together. The client then actually goes to the virtual server when redirected. If the **direct-ip** command is given only one IP address and port combination, that address is used on both the real server and the virtual server. When the client sends a request to that address, LocalDirector answers on behalf of the server with its MAC address. An optional second IP address and port combination may be given to the **direct-ip** command. If so, the first address is the virtual server and the second address is the real server. This allows server administrators the option of not changing server IP addresses or advertising internal IP addresses to the world.
2. The **link** command is not necessary for HTTP redirection to work, however it provides the means to dynamically update URLs. When a direct IP address is linked to a URL, the URL inherits the connection counters and status of the direct IP address. The connection counters are necessary if you are doing leastconns; the connection counters let the predictor accurately know how many connections have currently been redirected to that particular URL. If the direct IP address is changed to Out of Service (OOS), the URL that it is linked to will automatically be set to OOS also.
3. Backups are used to prevent the user from book marking a broken link. If the user is redirected to coyote.acme.com, for instance, they may choose to book mark that site. If they come back to that book mark later and coyote happens to be failed, the backup will send them back to www.acme.com which will allow them to get redirected to a server that is in service.

Restrictions

The following restrictions apply to HTTP redirection mode:

- URL real servers cannot be bound to virtual servers that already have an SSL sticky time-out, FTP proxy service, or any other sticky/proxy service.
- URLs can only be used as backups for virtual servers, not real servers.
- You will get an error message if you try to use the **direct-ip** command with a virtual server that has been associated with a content *rule_name* with the **virtual** command.
- You can only use the **direct-ip** command with TCP servers.

External Restrictions and Configuration

If redirection is applied, the web servers must be able to dynamically create absolute URLs using the HOST field of the content (GET) request. IP addresses needed to create DIPs must be routable or NIC-registered. URLs used must be DNS registered unless routable IP addresses are used.

HTTP Redirection Feature Command Reference

This section documents new and modified commands associated with the HTTP redirect feature.

backup

The **backup** command has been modified to allow you to specify a URL as backup of virtual servers. The command syntax is:

[no] backup {*real_id* **|** *virtual_id***}** *backup_id*

Syntax Description

virtual_id	Virtual server ID to bind to, and includes the virtual server IP address or name, port number, bind-id, and protocol.
real_id	Real server IP address or name, port (if a port-bound server), bind-id, and protocol.
backup_id	You can specify either a virtual server, real server, or a url_id as the backup. Use the IP address or name, port number (if a port-bound server), bind-id, and protocol of the real or virtual server that becomes the backup. If you specify a url_id you can only backup a virtual server. You cannot backup a real server with a url_id.

Usage Guidelines

The url_id is the ID for the uniform resource locator of a real server as defined by the **url** command. If you use a url_id as the designated backup server, the url_id can be used to backup only a virtual server.

Note URL real servers cannot be bound to virtual servers that already have an SSL sticky time-out, FTP proxy service, or any other sticky/proxy service.

bind

The **bind** command has been modified to allow binding of URLs to a real server. The command syntax is:

[**no**] **bind** *virtual_id* {*real_id* | *url_id*} {*real_id* | *url_id...*}

Syntax Description

virtual_id	Virtual server ID to bind to, and includes the virtual server IP address or name, port number, bind-id, and protocol. **Note** URLs cannot be bound to FTP proxy virtual servers, or any virtual server that has a sticky service such as cookie sticky, SSL sticky, generic sticky, etc.).
real_id	Real server IP address or name, port (if a port-bound server), bind-id, and protocol.
url_id	The ID for the uniform resource locator of the real server as defined by the **url** command.

Usage Guidelines

In the example where the virtual server is defined as 191.191.191.100:0 (a registered NIC), use the bind between URLs and a real server as shown:

```
LocalDirector(config)# virtual 191.191.191.100:0
LocalDirector(config)# url w1 http://www1.acme.com
LocalDirector(config)# url w2 http://www2.acme.com
LocalDirector(config)# bind 191.191.191.100:0 w1
LocalDirector(config)# bind 191.191.191.100:0 w2
```

You should create a URL_id for each Web server at your site. Use the **show bind** command to confirm binding of URLs to each real server.

direct-ip

To create a virtual server and a real server with the same IP address, and establish a one-to-one binding between the virtual and real server in LocalDirector. Use **dip** for the abbreviation of the **direct-ip** command.

[no] direct-ip *virtual_ip***[:[***port***]:[***bind-id***]:[***protocol***]]**
[*real-ip***][:[***port***]:[***bind-id***]:[***protocol***]] [***service-state***]**

You must use the **no direct-ip** command to remove the one-to-one binding between the virtual and real server in LocalDirector.

Syntax Description

virtual-ip	Virtual server IP address to bind to, and includes the virtual server IP address or name, port number, bind-id, and protocol.
	Note You will get an error message if you try to use the **direct-ip** command with a virtual server that has been associated with a content *rule_name* with the **virtual** command.
dip-port	
bind-id	(Optional) Used to bind the same *ip*:*port*:*protocol* to multiple virtual servers. Use a colon as a delimiter between the bind-id and the port number. If you do not specify a bind-id when defining a real server, the default is 0.
protocol	The protocol must be **tcp**.
real-ip	Real server IP address to bind to, and includes the real server IP address or name, port number, bind-id, and protocol.
service state	(Optional) Enter **is** for in-service state or **oos** for out-of-service state.

Note If the *real_ip* argument is not included, then the real and virtual servers have the same IP address.

Usage Guidelines

Note You cannot bind multiple real servers to dips.

Once you create and establish the one-to-one bind between the virtual and real server using the same IP address with the **direct-ip** command, the bind condition remains in effect until you use the **no direct-ip** command. Do not use the **no bind** command to remove the one-to-one bind established with the **direct-ip** command.

in-service

The **in-service** command has been modified to allow you to specify a URL. Use **is** for the abbreviation of the **in-service** command. The command syntax is:

in-service {**virtual** *virtual_id* | **real** *real_id* | **url** *url_id* | **dip** *dip_id*} [**all**]

Syntax Description

virtual	Sets a virtual server service state to in service.
virtual_id	Virtual server IP address or name, port number, bind-id, and protocol to be put in service.
real	Sets a real server service state to in service.
real_id	The IP address or name, port (if a port-bound server), bind-id, and protocol of the real server to be put in service.
url	Sets the server behind the URL to an in-service state.
url_id	The ID for the uniform resource locator of the real server as defined by the **url** command.
dip	Sets a DIP service state to in service.
dip_id	The IP address or name, port (if a port-bound server), bind-id, and protocol of the DIP to be put in service.
all	(Optional) Set the service state for all virtual servers or all real servers with the same IP address to in service. You do not need to specify port numbers and bind-ids.

link

The **link** command creates an association between a URL and a virtual server. The command syntax is:

[**no**] **link** *url_id* {*virtual_id* | *dip_id*}

Syntax Description

url_id	The ID for the uniform resource locator of the virtual server.
virtual_id	Virtual server ID to bind to, and includes the virtual server IP address or name, port number, bind-id, and protocol. The URL inherits this virtual server's weight and/or state attributes.
dip_id	The IP address created with the **dip** command for the one-to-one binding between the virtual server and the real server.

Usage Guidelines

The URL inherits all of the connection counters from the virtual server as a result of the **link** command. If you have defined the least connections (with the **predictor** command) as the load-balancing method, the URL inherits all the connection counters in the virtual server. LocalDirector assigns new connections to the physical server that has the least number of current connections. This feature ensures continued effective load balancing.

Note It is not necessary to use the **link** command for http redirection to work properly. However, if the **link** command is not used, LocalDirector could redirect clients to a failed virtual or real server created with the **dip** command.

Note If you do not use the **link** command, use the **predictor** command with either **roundrobin** or **loaded** load balancing type for the virtual server.

out-of-service

The **out-of-service** command has been modified to allow you to specify a URL. Use **oos** for the abbreviation of the **out-of-service** command. The command syntax is:

out-of-service {**virtual** *virtual_id*} | {**real** *real_id*} | {**url** *url_id* | **dip** *dip_id*} [**oos | maintenance | sticky | failed**] [**all**]

Syntax Description

virtual	Sets a virtual server service state to out-of-service.
virtual_id	Virtual server IP address or name, port number, bind-id, and protocol to be put out-of-service.
real	Sets a real server service state to out-of-service.
real_id	The IP address or name, port (if a port-bound server), bind-id, and protocol of the real server to be put out-of-service.
url	Sets the server behind the URL to an out-of-service state.
url_id	The ID for the uniform resource locator of the real server as defined by the **url** command.
dip	Sets a DIP service state to out-of-service.
dip_id	The IP address or name, port (if a port-bound server), bind-id, and protocol of the DIP to be put out-of-service.
oos (default)	(Optional) The default state; no new connections are sent to the server. Connections are cleared when server is back in service.
maintenance	(Optional) Similar to **oos**, but connections to the server are not cleared when put back in service.
sticky	(Optional) Same as **maintenance**, but only clients with sticky associations continue to receive those connections.
failed	(Optional) The server is failed by an external source (for example, another device notifies LocalDirector that an application is down). For real machines, the retry function is disabled. For virtual servers, no new connections are accepted. Once the real or virtual server is put back in service, all connections are cleared.
all	(Optional) Set the service state for all virtual servers or all real servers with the same IP address to out-of-service. You do not need to specify port numbers and bind-ids.

url

A Uniform Resource Locator (URL) is often a very long string of characters. The **url** command allows you to create a short identifier to replace a URL string. You must use the **url** command to define a Uniform Resource Locator. The command syntax is:

[no] url *url_id url_short_identifier* **[***http-code***]**

Syntax Description

url_id	The ID for the uniform resource locator.
url_string	The identifier you specify for this URL. Use valid URL forms (for example, http://www.acme.com). You can include valid macros that LocalDirector uses to retain parts of the original requests in the redirect URL. Valid macros include: • **%h** for hostname • **%p** for directory • **%s** for port number For example, a short identifier could be entered as: **http://%h/new/%p** for: http://doc.system.cisco/new/instructions/ The length of the URL string is limited to 128 characters.
http-code	(Optional) The HTTP error code used in the redirect. If no code is specified, the default code used is: 302 Moved Temporarily The other Redirection 3xx code supported is: 301 Moved Permanently

Usage Guidelines

Once you use the **url** command to define the short identifier, you must continue to use that short identifier for all references to that URL.

weight

The **weight** command has been modified to work with a URL. The command syntax is:

[**no**] **weight** {*real_id* | *url_id*} *number* [*time_value*]

Syntax Description

real_id	The IP address or name, port (if a port-bound server), bind-id, and protocol of a real server.
url_id	The ID for the uniform resource locator of the real server as defined by the **url** command.
number	The number that is averaged to determine the distribution of current connections among real servers. The default is one, and the value can be a whole number from 0 to 100 ~~65535~~. A value of 0 is equivalent to placing the server out of service.
time_value	(Optional) The time in seconds that a dynamic weight value remains in effect. The parameter allows an external agents like an SNMP controller or DFP to override the configured (static) weight number for a period of time set by the time_value. Valid inputs are whole numbers greater than or equal to zero or 'never' (default). Zero defaults to "never." An "m" or "s" appended to the input sets the input denomination to minutes or seconds, respectively. The static time_value should always be set to "never."

Content Load Balancing Overview

LocalDirector server load balancing techniques can now be maximized with the implementation of client content load balancing services. Content load balancing in LocalDirector is based on Layer 4 through Layer 7 protocol implementations such as hypertext transfer protocol (HTTP) message exchanges between Web servers and client browsers. LocalDirector also handles persistence integrity with HTTP 1.1 clients in a content load-balancing environment.

The content load balancing feature in LocalDirector looks for the specified rule matches within the http header that are generated as a result of the Uniform Resource Locator (URL) request from the http client or browser. LocalDirector parses the string containing the URL and HTTP header data and uses content-rule matches to determine which server receives the GET request from the client.

The following is an example of the complete contents of the http header generated by the client and sent to the http server when the URL entered is http://www.acme.com/home/index.html

```
GET /home/index.html HTTP/1.0\r\n
Connection: Keep-Alive\r\n
User-Agent:  Mozilla/4.74 [en] (WinNT; U)\r\n
Host: www.acme.com\r\n
Accept: Image/gif, Image/x-xbitmap, Image/jpeg..
Accept-Encoding: gzip\r\n
Accept-Language: en\r\n
Accept-Charset: iso-8859-1, *, utf-8\r\n
\r\n
```

In LocalDirector software version 4.1.1, you can configure LocalDirector to search and match content strings in the incoming HTTP header. You define the load balancing rules that can be used by LocalDirector based upon a successful match of a content string. Also, you can use the **service** command to define LocalDirector actions for services supported through the virtual server. For example, when you use the **clb-close** argument with the **service** command, you ensure that LocalDirector uses load balancing services for the initial content rule matched. LocalDirector ensures that the destination server receives a Connection:close, but that the client continues to use that server for all other requests during the same connection (persistence).

The content load balancing feature requires the assignment of a specific content rule to a virtual server at the time that virtual server is created. Each configuration of a virtual server can contain a pointer to a content rule. Wildcard matching functions are implemented with the following restrictions:

Wildcard Character	Description of Function	Literal Input
?	In the content string, one character is this position can have any value.	\?
*	Multiple characters of any value can appear in the content string.	*

LocalDirector continues to support proxy services for TCP exchanges, as well as secondary proxy services such as:

- HTTP redirect client-to-server persistence for all Secure Sockets Layer-based (SSL-based) applications (HTTP redirect)
- Client-to-server persistence based on LocalDirector insertion of cookies or server-generated cookies (cookie sticky)
- Client-to-server persistence based on SSL session ID (sticky SSL)

Configuring a Virtual Server with Content Load Balancing

The following steps progress through a sample configuration.

	Command	Purpose
Step 1	`LocalDirector(config)# content-rule home1 depth 1024 "/home/index.htm*www.acme.com/home/"`	Creates the name and definition of the content rule for content load balancing. The name on the content rule is home1, the data packet is searched to a "depth" of 1024 bytes, and the search should match the URL substring of www.acme.com/home/index.htm.
Step 2	`LocalDirector(config)# virtual 10.10.10.1:442:1:tcp:home1 is`	Defines the virtual server and specifies the content rule that is associated with this virtual server.

	Command	Purpose

Step 3	`LocalDirector(config)# show rule`
	`Rule Name Depth Content to Match`
	`gold001 1024 “/files/customer/gold/?????.lst”`
	`spec204 2048 “/public/info/sales/corporate/filesys/gateway/lev...”`
	`images:152 256 “/files/images/*.gif”`
	`images:153 256 “/files/images/*.jpg”`
		Displays all information for all content rules defined for this LocalDirector. Note that the display for all rule names containing more than 48 characters is abbreviated with three dots (...).

Step 4	`LocalDirector(config)# show rule spec204`
	`Rule Name Depth Content to Match`
	`spec204 2048 “/public/info/sales/corporate/filesys/gateway/leveraged/index”`
		Shows the entire rule name.

Configuring a Default Virtual Server for Content Load Balancing

Use the **virtual** command to define a default virtual server but do not specify a content-rule. If you create a default virtual server, the first virtual server in the list displayed with the **show virtual** command is the default virtual server. All packets for this virtual server that do not contain a pattern match to one of the content-rules are directed to the default virtual server.

If you do not create a default virtual server and no content match is found, Local Director sends a TCP RESET to the client and the connection is closed because no rule was matched and a default rule did not exist.

	Command	Purpose
Step 1	`LocalDirector(config)# virtual 10.10.10.12:0:0:tcp is`	Defines a default virtual server (the server that receives all traffic that does not match any of the content rules defined for a particular *virtual-id*)

Step 2	`LocalDirector(config)# virtual 10.10.10.10:0:0:tcp:rule01 is`
	
	`LocalDirector(config)# virtual 10.10.10.11:0:0:tcp:rule02 is`
		Defines rules for other virtual servers.

Caution You should not create a rule name that contains only the asterisk (*) wildcard.

Content Load Balancing Feature Command Reference

This section documents new and modified commands associated with the content load balancing feature.

content-rule

To define the rules to use for content load balancing for a virtual server, use the **content-rule** command. To remove a content rule, no virtual servers can be bound when the **no content-rule** command is issued. Use **rule** for the abbreviation of the **content-rule** command.

[**no**] **content-rule** *rule_name* [**depth** *number of bytes*] **"***content_string***"**

Syntax Description

rule_name	Enter a maximum of 8 characters that identifies the name of the rule used in the **virtual** command. You can use any alphanumeric characters and the colon (:) character.
depth *number of bytes*	(Optional) Defines the number of bytes in the data packet to be examined for a match with the *content_ string*. The default depth is 2048 bytes.
"*content_string***"**	Defines the actual character string to be matched by examining the data packet. You must surround characters with quotation marks (" "). A maximum of 256 characters (including the surrounding question marks) is allowed in the *content_string*.

Note Wildcard characters including the question mark (?) and asterisk (*) can be used in the *content_string* parameter with the following meanings and functions:

? indicates one character in this position can be any value. Use \? to specify a literal match for the question mark character.

* indicates that multiple characters of any value can be in this position. Use * to specify a literal match for the asterisk character.

A *content_string* of "*" matches any data and is an automatic match for using the *rule_name* defined with the "*" *content_string*.

Usage Guidelines

Caution You should not create a rule name that contains only the asterisk (*) wildcard as the content string. All content would match and go to the virtual server configured for this rule.

Note You can define a maximum of 512 content rules in each LocalDirector.

Note LocalDirector cannot support multi-node load balancing (MNLB) and content load balancing for a virtual server at the same time.

Note LocalDirector cannot support accelerated server load balancing (ASLB) and content load balancing for a virtual server at the same time.

Note LocalDirector cannot support sticky connections and content load balancing for a virtual server at the same time.

Note LocalDirector cannot support HTTP redirect and content load balancing for a virtual server at the same time.

Note LocalDirector cannot support FTP proxy and content load balancing for a virtual server at the same time.

service

To set the type of service enhancements provided by the virtual server, use the **service** command. Use the **no service** command to reset the service.

[**no**] **service** *virtual_id* [**ftp-proxy** | **clb-close**]

Syntax Description

virtual_id	Virtual server IP address or name, port number, bind-id, and protocol of the virtual server where connections are replicated.
ftp-proxy	Enables the FTP service.
clb-close	(Optional) Close all connections to this virtual server after completing the first GET request. The default behavior is to use content load balancing services for all connections based on the initial rule matched, and use this virtual server for all other requests in the same connection (persistence).

Usage Guidelines

Note The default behavior is to use content load balancing services for all connections based on the initial rule matched, and use this virtual server for all other requests in the same connection (persistence). The default behavior is to support HTTP 1.1 pipelined connection requests.

Note The maximum number of rule-based virtual servers that share the same machine identification (ip_address:port:bind-id:protocol) is 64 including a default virtual server.

show connections

Use the **show connections** command to see the number of currently used, and maximum used proxy connection objects for content-load balancing.

show connections

Syntax Description

This command has no arguments or keywords.

Usage Guidelines

Proxy connection objects include:

- All open TCP handshakes
- All currently open/active connections
- All connections being closed
- All open/inactive connections
- All connections not properly closed and waiting for LocalDirector timers to expire and close connection.

show rule

Use the **show rule** command to see all rule names, number of bytes (column heading is Depth) to be examined for a match, and the first 48 characters of the content string for the matching function.

show rule *rule_name*

Syntax Description

rule_name	Specify the name of the rule for a display of the rule name, depth, and entire content string for the matching function.

Examples

```
LocalDirector(config)# show rule
Rule Name    Depth      Content to Match
gold001      1024       “/files/customer/gold/?????.lst”
spec204      2048       “/public/info/sales/corporate/filesys/gateway/lev...”
images:152    256        “/files/images/*.gif”
images:153    256        “/files/images/*.jpg”

LocalDirector(config)# show rule spec204
Rule Name    Depth      Content to Match
spec204      2048       “/public/info/sales/corporate/filesys/gateway/leveraged/index”
```

virtual

To create a virtual server to accept connections from the network, or to specify a content rule to be associated with this virtual server, use the **virtual** command. Use the **no virtual** command to remove the virtual server from LocalDirector.

[**no**] **virtual** *virtual_name* | *virtual_ip*
[**:**[*virtual_port*]**:**[*bind-id*]**:**[*protocol*]**:**[*rule_name*]] [*service-state*]

Syntax Description

virtual_name	Name of the virtual server being defined. **Note** You will get an error message if you try to use the **direct-ip** command with a virtual server that has been associated with a content *rule_name* with the **virtual** command.
virtual_ip	IP address of the virtual server being defined.
virtual_port	(Optional) Port traffic that runs on the server. Use a colon as a delimiter between the IP address and the port number. If you do not identify a specific port, all traffic is allowed to pass to the server and the port is labeled 0. Servers with a port specified are called to as “port-bound” servers.
bind-id	(Optional) Used with the **assign** command to direct traffic to a specific location. Use a colon as a delimiter between the bind-id and port number. If you do not specify a bind-id when defining a virtual server, the default is :0. Any client IP address *not* identified by an **assign** command statement will be directed to the default bind-id of 0.
protocol	(Optional) Protocol to use. The default value is **tcp**, but **udp** is available. Use a colon as a delimiter between the bind-id and protocol. **Note** You can only configure a content *rule_name* for TCP virtual servers.
rule_name	(Optional) Name of the content rule to be used for connections associated with this virtual server. A maximum of 8 characters can be used in the name of the rule.
service-state	(Optional) In service (**is**) or out of service (**oos**). The default is **oos**.

Integrated Probe for DNS Overview

This implementation of a DNS probe allows LocalDirector to automatically fail or un-fail reals, running DNS servers, based on probe results. Probes are constantly sent to the DNS servers to determine their status. If a DNS server fails to respond to a certain number of probes it will be marked as EFAILED. As soon as the DNS server starts responding to DNS probes again, it is returned to the in-service state.

Configuring Integrated Probe for DNS

In the following configuration example, LocalDirector is configured with two virtual servers 10.10.10.10 and 10.10.10.20; and three real servers 10.10.10.50, 10.10.10.60, and 10.10.10.70. Real servers 10.10.10.50 and 10.10.10.60 are bound to virtual server 10.10.10.10; and real servers 10.10.10.60 and 10.10.10.70 are bound to virtual server 10.10.10.20.

To activate the DNS probe, follow these configuration steps:

	Command	Purpose
Step 1	`LocalDirector(config)# probe virtual 10.10.10.10 dns 1`	Enables the LocalDirector DNS probe requests to be sent every 10 seconds. **Note** DNS probe can only be configured on virtual or real servers that use the User Datagram Protocol (UDP). **Note** You would only specify a real server with the **probe real** command to override the interval time set with the **probe virtual** command.

	Command	Purpose
Step 2	`LocalDirector(config)# probedns nodename server1.com`	Configures the name of the node to be used in the DNS query requests sent to the DNS Servers **Note** Configure the real DNS server to respond to DNS queries for server1.com. **Note** Verify that the real DNS server is functioning.
Step 3	`LocalDirector(config)# probeconfig dns 5`	Configures a threshold value that limits the number of DNS requests sent without a response. When this threshold is exceeded, LocalDirector marks the DNS server as failed. In this example the threshold is 5 unanswered queries.

Integrated Probe for DNS Feature Command Reference

This section documents new and modified commands associated with the integrated probe for DNS feature.

probeconfig

To enable probe requests to be sent to the DNS servers, use the **probeconfig** command. To stop probe requests from being sent, use the **no probeconfig** command.

[**no**] **probeconfig** *probe_type threshold*

Syntax Description

probe_type	Enter **dns** to enable or disable the DNS probe.
threshold	When specified with the DNS probe enabled, this threshold is the maximum number of probes that can be sent to the DNS server without a response. If the threshold is exceeded, the real server is marked as EFAILED. Valid values range from 1 to 99.

Usage Guidelines

Use the **show probeconfig** command to display or verify settings.

Note Use of the **clear configuration** command does not remove any settings configured with the **probeconfig** command.

probedns nodename

To configure the name of the node to be used in DNS query request packets generated by the LocalDirector, use the **probedns nodename** command.

[**no**] **probedns nodename** *node_name*

Syntax Description

node_name	Enter the node name to be used in DNS query requests generated by the LocalDirector. The *node_name* parameter can be anything that has been configured on the DNS server, and if the DNS server receives a query for this *node_name*, the DNS server responds with no error codes and one or more DNS answer records. This *node_name* must be configured on all DNS servers that are to be probed by this LocalDirector.

Usage Guidelines

Note Use of the **clear configuration** command does not remove any settings configured with the **probedns nodename** command.

probe

To configure DNS probe for a User Datagram Protocol (UDP) virtual or real server , use the **probe** command.

[**no**] **probe real | virtual** {*real_id | virtual_id} probe_type interval*

Syntax Description

real	Sets DNS probe interval for a real server.
real_id	The IP address or name, port number (if a port-bound server), bind-id, and protocol of the real server.
virtual	Sets DNS probe interval for a virtual server.
virtual_id	Virtual server IP address or name, port number, bind-id, and protocol of the virtual server.
probe_type	Enter **dns** for the DNS probe.
interval	Specifies the interval of time between probe requests. Valid values range from 0 (disable probe requests) to 99. Each interval is 10 seconds. For example, if you specify the *interval* as 2, probe requests are sent every 20 seconds.

Usage Guidelines

Note If you specify DNS probes for a real server, any interval time assigned by a bound virtual server is overridden (by use of the **bind** command between the real server and virtual server and the **probe** command on the virtual server).

Note You would only specify a probe for a real server with the **probe real** command to override the interval time set with the **probe virtual** command.

Note Use of the **clear configuration** command removes all settings configured with the **probe** command.

Note It is recommended that real servers that are bound to virtuals that are going to be probed should be defined as port-bound. Otherwise, if the DNS probe fails the real server because the DNS service is unavailable will cause other possible services being used by other virtuals to also become affected.

Caveats

Caveats describe unexpected behavior in Cisco LocalDirector software releases. This section lists the open and resolved caveats for LocalDirector software version 4.1.1.

Open Caveats—Cisco LocalDirector Software Version 4.1.1

- CSCdr99437

  Physical disconnection of real server does not change status from In Service to Out Of Service in Content Flow Manager (CFM) Product.

- CSCds39929

  Under extreme load, Content Load Balancing (CLB) performance may vary.

- CSCds05044

  If real servers are added to a virtual server already discovered by Content Flow Manager (CFM), the additional real servers will not appear in Content Flow Manager.

- CSCds18931

  High volume of traffic to a virtual server configured for Content Load Balancing (CLB) causes other virtual servers configured without Content Load Balancing to go into FAILED state.

- CSCds20895

  LocalDirector takes 6 seconds to retransmit synchronous packets to a server when that server responds with a bogus ACK packet.

- CSCds16524

  Under extreme load, an interface that is not shutdown and not plugged into the switch will allocate blocks and never free blocks. Shutting down the interface stops the leak. If it is brought back up, the leak continues.

- CSCds42088

  Under extreme load in Cookie Insert mode, the cookie may not expire in the exact time specified.

- CSCds18917

  Content Load Balancing rule content string cannot have a white space for the beginning character.

- CSCds29043

  If direct IP addresses (dips) havethe same real server IP and different port/bind numbers they cannot be removed from the LocalDirector.

- CSCds39923

  Under extreme load in Cookie Insert mode, connections proxied by LocalDirector will not have cookies inserted by LocalDirector.

Resolved Caveats—Cisco LocalDirector Software Version 4.1.1

- CSCdr22912

  For FTP data connections, LocalDirector is not forwarding syn/ack from client.

- CSCds29199

  LocalDirector does not source TCP Reset packets to client if server is in failed state.

- CSCds16931

  The LocalDirector will source ICMP ECHO requests from server side vlan.

- CSCdm44441

  LocalDirector does not rotor UDP traffic with source port of 0.

- CSCdr71284

  Boot config does not work past version 3.1.4.

- CSCds17070

  STATIC command does not allow connections from the server to the client.

- CSCdp49763

  Sticky timer is only updated on new connections.

- CSCdr33076

  If real server does not answer for a connection initated by LocalDirector in a proxied connection, once connection is reassigned to new real server, a reboot can occur.

- CSCdp99442

  Backup virtual does not bring its real IS after FAILING.

- CSCdr32478

  LocalDirector does not report the correct information to DD when there is a change in it Bindid Database.

- CSCdr33978

  When the DFP agent is active on the LocalDirector and all virtual servers have been removed the LocalDirector does not send out preference info messages as required by the DFP specification. This causes the DD to continually retry to establish the connection.

- CSCdr55437

  LocalDirector clock occasionally stops running after the **clock set** command.

- CSCdr37219 [Duplicate of CSCdr74300]

  LocalDirector crashes when it completely runs out of connection objects.

Related Documentation

Use these release notes in conjunction with the following documents:

- *Release Notes for Cisco LocalDirector Software Version 4.1.2*
- *Cisco LocalDirector Configuration and Command Reference Guide*
- *Cisco LocalDirector 417 Hardware Installation Guide*
- *Cisco LocalDirector Installation and Configuration Guide, Version 3.3*

Obtaining Documentation

The following sections provide sources for obtaining documentation from Cisco Systems.

World Wide Web

You can access the most current Cisco documentation on the World Wide Web at the following sites:

- http://www.cisco.com
- http://www-china.cisco.com
- http://www-europe.cisco.com

Documentation CD-ROM

Cisco documentation and additional literature are available in a CD-ROM package, which ships with your product. The Documentation CD-ROM is updated monthly and may be more current than printed documentation. The CD-ROM package is available as a single unit or as an annual subscription.

Ordering Documentation

Cisco documentation is available in the following ways:

- Registered Cisco Direct Customers can order Cisco Product documentation from the Networking Products MarketPlace:

  http://www.cisco.com/cgi-bin/order/order_root.pl

- Registered Cisco.com users can order the Documentation CD-ROM through the online Subscription Store:

  http://www.cisco.com/go/subscription

- Nonregistered CCO users can order documentation through a local account representative by calling Cisco corporate headquarters (California, USA) at 408 526-7208 or, in North America, by calling 800 553-NETS(6387).

Documentation Feedback

If you are reading Cisco product documentation on the World Wide Web, you can submit technical comments electronically. Click **Feedback** in the toolbar and select **Documentation**. After you complete the form, click **Submit** to send it to Cisco.

You can e-mail your comments to bug-doc@cisco.com.

To submit your comments by mail, for your convenience many documents contain a response card behind the front cover. Otherwise, you can mail your comments to the following address:

Cisco Systems, Inc.
Document Resource Connection
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

Obtaining Technical Assistance

Cisco provides Cisco.com as a starting point for all technical assistance. Customers and partners can obtain documentation, troubleshooting tips, and sample configurations from online tools. For Cisco.com registered users, additional troubleshooting tools are available from the TAC website.

Cisco.com

Cisco.com is the foundation of a suite of interactive, networked services that provides immediate, open access to Cisco information and resources at anytime, from anywhere in the world. This highly integrated Internet application is a powerful, easy-to-use tool for doing business with Cisco.

Cisco.com provides a broad range of features and services to help customers and partners streamline business processes and improve productivity. Through Cisco.com, you can find information about Cisco and our networking solutions, services, and programs. In addition, you can resolve technical issues with online technical support, download and test software packages, and order Cisco learning materials and merchandise. Valuable online skill assessment, training, and certification programs are also available.

Customers and partners can self-register on Cisco.com to obtain additional personalized information and services. Registered users can order products, check on the status of an order, access technical support, and view benefits specific to their relationships with Cisco.

To access Cisco.com, go to the following website:

http://www.cisco.com

Technical Assistance Center

The Cisco TAC website is available to all customers who need technical assistance with a Cisco product or technology that is under warranty or covered by a maintenance contract.

Contacting TAC by Using the Cisco TAC Website

If you have a priority level 3 (P3) or priority level 4 (P4) problem, contact TAC by going to the TAC website:

http://www.cisco.com/tac

P3 and P4 level problems are defined as follows:

- P3—Your network performance is degraded. Network functionality is noticeably impaired, but most business operations continue.
- P4—You need information or assistance on Cisco product capabilities, product installation, or basic product configuration.

In each of the above cases, use the Cisco TAC website to quickly find answers to your questions.

To register for Cisco.com, go to the following website:

http://www.cisco.com/register/

If you cannot resolve your technical issue by using the TAC online resources, Cisco.com registered users can open a case online by using the TAC Case Open tool at the following website:

http://www.cisco.com/tac/caseopen

Contacting TAC by Telephone

If you have a priority level 1(P1) or priority level 2 (P2) problem, contact TAC by telephone and immediately open a case. To obtain a directory of toll-free numbers for your country, go to the following website:

http://www.cisco.com/warp/public/687/Directory/DirTAC.shtml

P1 and P2 level problems are defined as follows:

- P1—Your production network is down, causing a critical impact to business operations if service is not restored quickly. No workaround is available.
- P2—Your production network is severely degraded, affecting significant aspects of your business operations. No workaround is available.

This document is to be used in conjunction with the documents listed in the "Related Documentation" section.