



Configuring HTTP Header Load Balancing

This chapter describes how to configure HTTP header load balancing by creating an HTTP header field group and configuring HTTP header fields. Information in this chapter applies to all CSS models except where noted.

This chapter contains the following sections:

- HTTP Header Load Balancing Overview
- HTTP Header Load Balancing Configuration Quick Start
- Creating a Header Field Group
- Describing the Header Field Group
- Configuring a Header Field Entry
- Associating a Header Field Group to a Content Rule
- Showing a Content Rule Header Field Configuration
- Showing Header Field Groups
- Header Field Group Configuration Examples



Note

You must enable service remapping for HTTP header load balancing to work properly. For information on the service remapping feature, refer to the *Content Services Switch Basic Configuration Guide*.

HTTP Header Load Balancing Overview

Configuring HTTP header load balancing enables the CSS to inspect incoming content requests for HTTP header fields. This allows the CSS to make load balancing decisions based on the HTTP header field information and then direct content requests to the servers designed to handle the type of content being requested.

The CSS can direct content requests to specific servers based on different types of browsers or different representations of the same content that has been modified for end users. For example, a client running a hand-held personal organizer may want the same content as a client using a PC, but with fewer graphics. Or German users may want to see content in German only.

Using HTTP header load balancing eliminates the need to duplicate various forms of the same content across all of the servers, thus freeing up valuable server space. In addition to dividing the server farm for different types of clients, you can also use HTTP header load balancing to bypass non-cacheable traffic and prioritize client browser traffic from search engine services.

Using HTTP Header Load Balancing in a Content Rule

Using an HTTP header field group in a Layer 5 content rule enables a rule to be more specific than if the rule just defined an URL. The HTTP header field group makes the content match more specific. Because content rules are hierarchical, if a request for content matches more than one rule, the characteristics of the most specific rule apply to the flow. This hierarchy for Layer 5 rules is defined below. The CSS uses this order of precedence to process requests for the content, with 1 being the highest match and 4 being the lowest match.

1. Domain name, IP address, protocol, port, URL, HTTP header field group
2. IP address, protocol, port, URL, HTTP header field group
3. Domain name, protocol, port, URL, HTTP header field group
4. Protocol, port, URL, HTTP header field group

HTTP Header Load Balancing Configuration Quick Start

Table 2-1 provides a quick overview of the steps required to create and configure HTTP header load balancing. Each step includes the CLI command required to complete the task. For a complete description of each feature and all the HTTP header load balancing configuration options, refer to the sections following Table 2-1.

Ensure that you have already created and configured a service and owner for the content rules. The command examples in Table 2-1 create HTTP load balancing for owner *arrowpoint* and content rule *rule1*.

Table 2-1 HTTP Load Balancing Configuration Quick Start

Task and Command Example

1. Enter into config mode by typing **config**.

```
(config)#
```

2. Create a header field group. This example creates the group *ppilot*.

```
(config)# header-field-group ppilot  
(config-header-field-group[ppilot])#
```

3. Describe the header field group (optional).

```
(config-header-field-group[ppilot])# description "ppilot content"
```

4. Configure header field entries by defining a header, field, name, field type, and operator.

```
(config-header-field-group[ppilot])# header-field palm1  
user-agent contain "MSIE" 20
```

5. Associate the header field group to a content rule.

```
(config-owner-content[arrowpoint-rule1])# header-field-rule  
ppilot
```

6. Display the header field group to verify your configuration (optional).

```
(config)# show header-field-group
```

Creating a Header Field Group

Header field group configuration mode allows you to create a *header field group*. A header field group contains a list of user-defined header field entries used by the CSS content rule lookup process. A group can contain several header-field entries.

**Note**

When there is more than one header field entry in a group, each header field entry must be successfully matched before the CSS uses the associated content rule.

To create a header field group or to access header field group configuration mode, use the **header-field-group** command from all configuration modes except boot and RMON modes.

The prompt changes to (config-header-field-group [*group_name*]). You can also use this command in header-field-group mode to access another group.

The syntax for this mode transition command is:

header-field-group *group_name*

Enter the *group_name* of the header-field group you want to create. You must define a unique name for each header field group so different content rules can use the groups. Enter a text string with a maximum of 32 characters. To see an existing list of header-field groups, enter **header-field-group ?**.

For example:

```
(config)# header-field-group ppilot  
(config-header-field-group[ppilot])#
```

To remove a header-field group, use the **no header-field-group** command. For example:

```
(config)# no header-field-group ppilot
```

Describing the Header Field Group

Use the **description** command to provide a description for a header field group. The syntax for this command is:

```
description "text"
```

Enter the text as a quoted text string with a maximum length of 64 characters.

For example,

```
(config-header-field-group[ppilot])# description "ppilot content"
```

To remove a description for a header-field group, enter:

```
(config-header-field-group[ppilot])# no description
```

Configuring a Header Field Entry

Use the **header-field** command to define a header field entry in a header field group. A header field entry contains a header field name, field type to be used, an operation to be performed, the header-string to be searched for, and an optional search length.

If a header field group contains multiple header field entries, a content request must match each entry for the rule to be used.

The syntax for this command is:

```
header-field name field_type operator {header_string {search_length}}
```

The variables and options are:

- *name* - The name uniquely identifies the header field entry. Enter the name as a string from 1 to 31 characters. You must define a header field entry name because the CSS can use the same field type multiple times in a header field group.
- *field_type* - The field type includes one of the following:
 - **user-agent**
 - **language**

- **host**
- **cache-control**
- **pragma**
- **encoding**
- **charset**
- **connection**
- **referer**
- **accept**
- **request-line**
- **cookies**
- **msisdn** - The header field type for Wireless Application Protocol (WAP). HTTP requests from some wireless gateways contain the MSISDN field in the HTTP header. By configuring the msisdn header field type in a header field group, you can load balance wireless requests. Refer to “Example 3. Wireless configuration that load balances HTTP requests based on the MSISDN header field” later in this chapter.



Note You can use this feature alone or with the **advanced-balance wap-msisdn** sticky command. Refer to Chapter 1, Configuring Sticky Parameters for Content Rules, in the section “Specifying an Advanced Load Balance Method for Sticky Content”.

- *operator* - Enter one of the following operators:
 - **exist|not-exist** - Use the **exist** and **not-exist** operators to check whether or not a specified header field exists in a content request header.
 - **equal|not-equal** {“*header_string*”} - Use the **equal** and **not-equal** operators to match a defined *header_string* to the contents of the specified header field, and determine whether or not it is equal to the header string. Enter the *header_string* as a quoted text string with a maximum of 31 characters including spaces.

- **contain** | **not-contain** {"*header_string*" {*search_length*}} - Use the **contain** and **not-contain** operators to match the configured *header_string* to a substring in the contents of the specified field type, and determine whether or not its contents contain the *header_string*. Enter the *header_string* as a quoted text string with a maximum of 31 characters including spaces.

You may include an optional *search_length* to define the header field portion to be used for the operation. If you do not define a search length, the CSS uses the entire header field (delimited by a CR and LF) for the operation. To define the search length, enter a number from 0 to 1024.

For example:

```
(config-header-field-group[ppilot])# header-field palm1 user-agent  
contain "MSIE" 20
```

```
(config-header-field-group[ppilot])# header-field palm2 user-agent  
contain "palm"
```

To remove a header field entry, use the **no header-field** command. For example:

```
(config-header-field-group[ppilot])# no header-field palm1
```

Associating a Header Field Group to a Content Rule

Use the **header-field-rule** command to associate a header field group with a content rule, and optionally assign a weight value to the header field group. Use weights to allow the CSS to prefer one content rule over a similar content rule. For example, you want to load balance French clients to a specific server, and you also want to differentiate French Internet Explorer clients from French Netscape clients. If it is more important to direct the French clients to a specific server than to direct them to a server based on whether they are using Internet Explorer or Netscape, then you need to weight the "French" content rule higher than the "Internet Explorer/Netscape" content rule. The syntax for this content mode command is:

```
header-field-rule name {weight number}
```

The variables are:

- *name* - The name of the header field group used with the content rule. To see a list of groups, enter **header-field-rule ?**.
- **weight number** - The weight you want to assign to the header field group. Enter a number from 0 to 1024. The default weight is 0.

For example:

```
(config-owner-content[arrowpoint-rule1])# header-field-rule french  
weight 3
```

```
(config-owner-content[arrowpoint-rule1])# header-field-rule ppilot
```

To remove the header field group from the content rule, enter:

```
(config-owner-content[arrowpoint-rule1])# no header-field-rule
```

Showing a Content Rule Header Field Configuration

Use the **show rule-header-field** command to display information about the header field rule and group associated with a content rule. The syntax is:

```
show rule-header-field
```

For example, to display information about the header-field rule and group associated with a specific content rule, enter:

```
(config-owner-content[arrowpoint-rule1])# show rule-header-field
```

Showing Header Field Groups

Use the **show header-field-group** command to display the configuration for all header field groups or a specific group. This command is available in all modes.

The syntax and options for this command are:

- **show header-field-group** - Display a summary of all configured header field groups
- **show header-field-group all** - Display detailed information about all configured header field groups

- **show header-field-group** *name* - Display detailed information about a specific header field group

For example, to show a summary of all configured header field groups, enter:

```
(config)# show header-field-group
```

Table 2-2 describes the fields in the **show header-field-group** output.

Table 2-2 *Field Descriptions for the show header-field-group Command*

Field	Description
Header field group	The name of the header-field group
Description	The configured description for the header-field group

Header Field Group Configuration Examples

When configuring header field groups, it is good practice to configure rules to be specific in rule matching (as shown in configuration example 2). If the rules are not specific enough, the CSS may match a client request to the first rule it finds and the first matched rule could change on subsequent requests.

Configuration examples 1 and 2 show the header field group and owner portions of a running-config. Configuration example 3 shows a wireless configuration.

Example 1. Header field group configuration that is ambiguous in rule-matching capabilities.

Example 1 shows a configuration that is ambiguous. If a client request specifies the language as French and the user-agent as Netscape, this request could match equally to ruleA2 or ruleA3. In this example, the rule matching may not be consistent. One method to solve the ambiguity between ruleA2 and ruleA3 is to use different weight values. If you assign a weight value of 10 to header field rule B when you associate it with ruleA2, the CSS will always use ruleA2 as a match to the example client request. Another method is to configure more specific rules as shown in configuration example 2.

Header Field Group Configuration Examples

```

! ***** HEADER FIELD GROUP *****

header-field-group A
header-field ua1 language equal "en"

header-field-group B
header-field ua2 language equal "fr"

header-field-group C
header-field-group ua3 user-agent contain "Netscape"

! ***** OWNER *****

owner arrowpoint
content ruleA
protocol tcp
vip address 192.168.128.151
port 80
url "/"
add service server1
add service server2

content ruleA1
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule A
add service server11
add service server12

content ruleA2
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule B
add service server21
add service server22

content ruleA3
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule C
add service server31
add service server32

```

Example 2. Header field group configuration that broadens the rule-matching capabilities.

Example 2 shows the same configuration as Example 1 only modified to broaden the rule-matching capabilities. Each content rule is specific. The client request specifying the language as French and the user-agent as Netscape will only match on Rule A2.

```
! ***** HEADER FIELD GROUP *****

header-field-group A
header-field ua1 language equal "en"
header-field ua2 user-agent contain "Netscape"

header-field-group B
header-field ua3 language equal "fr"
header-field ua4 user-agent contain "Netscape"

header-field-group C
header-field ua5 language equal "en"
header-field ua6 user-agent not-contain "Netscape"

header-field-group D
header-field ua7 language equal "fr"
header-field ua8 user-agent not-contain "Netscape"

! ***** OWNER *****

owner arrowpoint
content ruleA
protocol tcp
vip address 192.168.128.151
port 80
url "/"
add service server1
add service server2

content ruleA1
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule A
add service server11
add service server12

content ruleA2
protocol tcp
vip address 192.168.128.151
```

```
port 80
url "/"
header-field-rule B
add service server21
add service server22

content ruleA3
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule C
add service server31
add service server32

content ruleA4
protocol tcp
vip address 192.168.128.151
port 80
url "/"
header-field-rule D
add service server41
add service server42
```

Example 3. Wireless configuration that load balances HTTP requests based on the MSISDN header field

Example 3 shows a configuration that makes load-balancing decisions based on whether or not a client is a wireless client. Wireless devices use the Wireless Application Protocol (WAP). When a wireless client sends a request for content, the WAP protocol gateway (a device that translates requests from the WAP protocol stack to the WWW protocol stack) generates the MSISDN field and adds it to the HTTP header. You can test for the presence of the MSISDN header field using the **exist** and **not-exist** operators in the header field entry of a header field group. Then, you can make load-balancing decisions based on the presence or absence of the MSISDN header field. For details on configuring the MSISDN header field type, refer to “Configuring a Header Field Entry” earlier in this chapter.

In the following example, any TCP port 80 traffic destined for VIP 192.168.128.151 that has the MSISDN field in the HTTP header will hit the content rule ruleWap. Any TCP port 80 traffic destined for 192.168.128.151 that does not have the MSISDN field in the HTTP header will hit the content rule ruleNoWap.

```
header-field-group wap
  header-field 1 msisdn exist

owner arrowpoint
  content ruleWap
    vip address 192.168.128.151
    protocol tcp
    port 80
    url "/*"
    add service server1
    add service server2
    header-field-rule wap
    active

  content ruleNoWap
    vip address 192.168.128.151
    protocol tcp
    port 80
    url "/*"
    add service server21
    add service server22
    active
```

**Note**

You can use the MSISDN header field with the **advanced-balance wap-msisdn** command to configure wireless users for e-commerce applications. For details, refer to Chapter 1, Configuring Sticky Parameters for Content Rules in the section “Configuring Wireless Users for E-Commerce Applications”.

■ Header Field Group Configuration Examples