

# Owner Configuration Mode Commands

Owner configuration mode allows you to configure an owner. An owner is an entity that owns Web content and uses the CSS to manage access to that content through content rules. Up to 255 owners can use a single CSS and each owner has a configurable profile.

To access owner configuration mode, use the **owner** command from any mode except ACL, boot, group, and RMON alarm, event, and history configuration modes. The prompt changes to (config-owner [*owner\_name*]). You can also use this command in owner mode to access another owner. For information about commands available in this mode, refer to the following commands.

Use the **no** form of this command to delete an existing owner.

```
owner owner_name
no owner existing_owner_name
```

---

## Syntax Description

*owner\_name*

The name of a new owner you want to create or the name of an existing owner. Enter an unquoted text string with no spaces and a maximum length of 31 characters. To see a list of existing owner names, enter:

```
owner ?
```

---

## (config-owner) address

To enter the address for the owner of the Web hosting service, use the **address** command. Use the **no** form of this command to delete an address for the owner.

```
address "address"
no address
```

---

## Syntax Description

*address*

The street address for the owner. Enter a quoted text string with a maximum length of 128 characters.

---

## (config-owner) billing-info

To enter billing information about the owner providing the Web hosting service, use the **billing-info** command. Use the **no** form of this command to delete the billing information.

```

billing-info "information"
no billing-info

```

Syntax Description	<i>information</i>	The billing information about the owner. Enter a quoted text string with a maximum length of 128 characters.
--------------------	--------------------	--

## (config-owner) case

To define whether the matching of content requests to the owner's rules is case-sensitive, use the **case** command.

```

case [insensitive|sensitive]

```

Syntax Description	<b>insensitive</b>	Matching of the owner's rules is not case-sensitive. Uppercase and lowercase characters in content requests are ignored.
	<b>sensitive</b>	Matching of the owner's rules is case-sensitive. Uppercase and lowercase characters in content requests are used as part of the matching criteria.

## (config-owner) content

To access content configuration mode and configure a content rule, use the **content** command. Use the **no** form of this command to an existing content rule.

```
content content_rule_name  
no content content_rule_name
```

---

### Syntax Description

<i>content_rule_name</i>	The name of a new content rule you want to create or an existing rule you want to modify. Enter an unquoted text string with no spaces and a maximum length of 31 characters. Enter an existing name exactly. To see a list of existing rules, enter:
--------------------------	---

**content ?**

---

---

### Usage Guidelines

When you use the **content** command to access this mode, the prompt changes to (config-owner-content [*owner-rule*]). For information about commands available in this mode, refer to the commands in “Content Configuration Mode Commands”.

---

### Related Commands

**show rule**  
**(config) service**

## (config-owner) dns

To set the peer DNS exchange policy for the owner, use the **dns** command. The default DNS exchange policy for the owner is none; the owner is hidden from the CSS peer. Use the **no** form of this command to reset the DNS exchange policy for the owner to its default setting of none.

```
dns [accept|push|both]
no dns
```

### Syntax Description

<b>accept</b>	Accepts all content rules proposed by the CSS peer
<b>push</b>	Advertises the owner and push all content rules onto the CSS peer
<b>both</b>	Advertises the owner and push all content rules onto the CSS peer, and accept all content rules proposed by the CSS peer

### Related Commands

```
(config) dns
(config) dns-server
(config-owner-content) add
```

## (config-owner) dnsbalance

To determine where to resolve a request for a domain name into an IP address, use the **dnsbalance** command. Use the **no** form of this command to reset the DNS load balancing method to its default setting of **roundrobin**.

**dnsbalance** [**leastloaded**|**preferlocal**|**roundrobin**]  
**no dnsbalance**

### Syntax Description

<b>leastloaded</b>	Resolves the request to the least-loaded local or remote domain site. The CSS first compares load numbers. If the load number between domain sites is within 50, then the CSS compares their response times. The site with the faster response time is considered the least loaded site.
--------------------	--



**Note** For the **leastloaded** option to work properly, all domain sites must be running a minimum of CSS software version 3.02.

<b>preferlocal</b>	Resolves the request to a local VIP address. If all local systems exceed their load threshold, the CSS chooses the least-loaded remote system VIP address as the resolved address for the domain name.
--------------------	--

<b>roundrobin</b>	Resolves the request by evenly distributing the load to resolve domain names amongst content domain sites, local and remote. The CSS does not include sites that exceed their local load threshold.
-------------------	---

### Usage Guidelines

The DNS load balancing method configured for the owner applies to all of its content rules. To set a different method to a specific content rule, use the **(config-owner-content) dnsbalance** command.

### Related Commands

**(config-owner-content) dnsbalance**

## (config-owner) email-address

To enter an email address for the owner providing the Web hosting service, use the **email-address** command. Use the **no** form of this command to delete the email address for the owner.

**email-address** *“information”*  
**no email-address**

Syntax Description	<i>information</i>	The email address for the owner. Enter a quoted text string with a maximum length of 64 characters.
--------------------	--------------------	---

## (config-owner) no

To negate a command or set it to its default, use the **no** command. For information on general **no** commands you can use in this mode, refer to the general **no** command. The following options are available in owner mode:

Syntax Description	<b>no acl</b> <i>index</i>	Deletes an ACL
	<b>no address</b>	Deletes the address for the owner
	<b>no billing-info</b>	Deletes the billing information for the owner
	<b>no content</b> <i>content_rule_name</i>	Deletes an existing content rule
	<b>no dns</b>	Resets the DNS exchange policy for the owner to its default setting of none
	<b>no dnsbalance</b>	Resets the DNS load balancing method for the owner to its default setting of roundrobin
	<b>no email-address</b>	Deletes the email address for the owner
	<b>no owner</b> <i>owner_name</i>	Deletes an existing owner
	<b>no rmon-event</b> <i>index</i>	Deletes an RMON event
	<b>no rmon-history</b> <i>index</i>	Deletes an RMON history

## (config-owner) zero

To set the content rule counters to zero, use the **zero** command.

```
zero all
```

---

**Related Commands**    show rule

# Content Configuration Mode Commands

Content configuration mode allows you to configure a content rule. In the CSS, content refers to Web content that exists as an object or set of objects. Content rules:

- Describe the content available at services
- Tell the CSS where the content physically resides, whether local or remote
- Specify how to treat requests for the content

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.

To access content mode, use the **content** command from owner mode. The prompt changes to (config-owner-content [*owner-rule*]). You can also use this command in content mode to access another content rule. For information about commands available in this mode, refer to the following commands.

Use the **no** form of this command to delete an existing content rule.

(config-owner) **content** *name*  
**no content** *content\_rule\_name*

---

## Syntax Description

*name*

The name of a new content rule you want to create or an existing rule you want to modify. Enter an unquoted text string with no spaces and a maximum length of 31 characters. Enter an existing name exactly. To see a list of existing rules, enter:

**content ?**

---

## (config-owner-content) active

To activate the content rule you are configuring, use the **active** command. Activating a content rule includes it in CSS content rule matching and load balancing decisions.

```
active
```

### Related Commands

```
show rule
show running-config
(config-owner-content) suspend
```

## (config-owner-content) add

To add an existing service to the content rule or to specify a DNS name that maps to the content rule, use the **add** command. The options for this command are:

- **add dns...**, specifies a DNS name that maps to the content rule
- **add service...**, adds an existing service to the content rule

For more information on these options, refer to the following commands.

### add dns

To specify a DNS name that maps to the content rule, use the **add dns** command.

```
add dns dns_name {ttl_value}
remove dns dns_name
```

### Syntax Description

<i>dns_name</i>	The DNS name mapped to the content rule. Enter the name as a case-sensitive unquoted text string with no spaces and a maximum length of 63 characters.
<i>ttl_value</i>	The optional Time to Live (TTL) value in seconds. This value sets how long the DNS client remembers the IP address response to the query. Enter a value of 0 to 255. The default is 0.

---

**Command Modes**      Owner-Content

---

**Usage Guidelines**      To add a TTL value to an existing DNS name, remove the name and then re-add it with the TTL value  
 Use the **remove dns** command to remove a DNS name mapped to the content rule.

---

**Related Commands**      **show rule**  
                                   **(config) dns**  
                                   **(config) dns-server**  
                                   **(config-owner) dns**  
                                   **(config-owner-content) remove**

## add service

To add an existing service to the content rule, use the **add service** command. Adding a service includes it in the resource pool that the CSS uses for load balancing requests for the content governed by the rule.

```
add service service_name weight number
remove service service_name
```

---

<b>Syntax Description</b>	<i>service_name</i>	The name of the existing service. Enter the name as a case-sensitive unquoted text string with no spaces.
	<b>weight</b>	This option allows you to assign a weight to the service used when you configure ACA and weighted roundrobin load balancing on the content rule. When you assign a higher weight, the CSS redirects more requests to the service. To configure load balancing, refer to the <b>(config-owner-content) balance</b> command.
	<i>number</i>	The weight for the service. Enter a number from 1 to 10. The default is the weight configured for the service through the <b>(config-service) weight</b> command. By default, all services have a weight of 1.

---

---

**Command Modes** Owner-Content

---

**Usage Guidelines** Use the **remove service** command to remove a service.

---

**Related Commands** **show rule**  
**(config) service**  
**(config-owner-content) balance**  
**(config-owner-content) remove**  
**(config-service) weight**

## **(config-owner-content) advanced-balance**

To specify an advanced load balancing method for the content rule, including stickiness, use the **advanced-balance** command. A content rule is “sticky” when additional sessions from the same user or client are sent to the same service as the first connection, overriding normal load balancing. By default, the advanced balancing method is disabled. Use the **no** form of this command to disable the advanced balancing method.

```
advanced-balance [arrowpoint-cookie|cookies|cookieurl|none  
|sticky-srcip|sticky-srcip-dstport|ssl|url|wap-msisdn]  
no advanced-balance
```

Syntax Description		
	<b>arrowpoint-cookie</b>	<p>Enables the content rule to stick the client to the server based on the unique service identifier information of the selected server in the ArrowPoint-generated cookie. Configure the service identifier by using the <b>(config-service) string</b> command. You do not need to configure string match criteria.</p> <p>For information on configuring the ArrowPoint-generated cookie, refer to the <b>(config-owner-content) arrowpoint-cookie</b> command. You can use this option with any Layer 5 content rule.</p>
	<b>cookies</b>	<p>Enables the content rule to stick the client to the server based on the configured string found in the HTTP cookie header. You must specify a port in the content rule to use this option. The CSS will then spoof the connection. A content rule with a sticky configuration set to <b>advanced-balance cookies</b> requires all clients to enable cookies on their browser.</p> <p>When a client makes an initial request, they do not have a cookie. But once they go to a server that is capable of setting cookies, they receive the cookie from the server. Each subsequent request contains the cookie until the cookie expires. A string in a cookie can be used to stick a client to a server. The service mode <b>string</b> command enables you to specify where the CSS should locate the string within the cookie.</p> <p>The CSS processes the cookie using:</p> <ul style="list-style-type: none"> <li>• An exact match that you set up when you configure the services</li> <li>• Data for a hash algorithm</li> </ul>

---

<b>cookieurl</b>	<p>Same as <b>advanced-balance cookies</b>, but if the CSS cannot find the cookie header in the HTTP packet, this type of failover looks up the URL extensions (that is, the portion after the “?” in the URL) based on the same string criteria. You can use this option with any Layer 5 HTTP content rule.</p> <p>This option is useful if a Microsoft<sup>®</sup> IIS Web server is used with Cookie Munger, that dynamically places the session state information in the cookie header or URL extension depending on whether or not the client can accept cookies.</p> <p>Some client applications do not accept cookies. When a site depends upon the information in the cookie, administrators sometimes modify the server application so that it appends the cookie data to the parameters section of the URL. The parameters typically follow a “?” at the end of the main data section of the URL.</p> <p>Advanced-balanced <b>cookieurl</b> sticks a client to a server based on locating the configured string:</p> <ul style="list-style-type: none"><li>• In the cookie, if a cookie exists</li><li>• In the parameters section of the URL if no cookie exists</li></ul> <p>The string can either be an exact match or be hashed.</p>
<b>none</b>	Disables the advanced-balancing method for the rule (default).

---

---

<b>sticky-srcip</b>	Enables the content rule to stick a client to a server based on the client IP address, also known as Layer 3 stickiness. You can use this option with Layer 3, 4, or 5 content rules.
<b>sticky-srcip-dstport</b>	Enables the content rule to stick a client to a server based on both the client IP address and the server destination port number, also known as Layer 4 stickiness. You can use this option with Layer 4 or 5 content rules.
<b>ssl</b>	<p>Enables the content rule to stick the client to the server based on the Secure Socket Layer (SSL) version 3 session ID assigned by the server. The application type must be SSL for the content rule. You must specify a port in the content rule to use this option. The CSS will then spoof the connection.</p> <p>Sites where encryption is required for security purposes often use SSL. SSL contains session IDs and the CSS has the ability to use these session IDs for sticking the client to a server. In order for the CSS to successfully provide SSL stickiness, the application must be using SSL version 3 session IDs. Sticky SSL uses the sticky table. If you are concerned about the number of concurrent sessions, and not concerned about security, you should consider using <b>cookie</b>, <b>cookieurl</b>, or <b>url</b>.</p>

---

---

**url**

Enables the content rule to stick a client to a server based on a configured string found in the URL of the HTTP request. You must specify a port in the content rule to use this option. The CSS will then spoof the connection. Similar to advanced-balanced **cookie**, advanced-balanced **url** may use either an exact match method or a hash algorithm. The string can exist anywhere in the URL.

To use stickiness based on SSL version 3 session ID, configure an SSL Layer 5 rule for the service:

- Set the port to 443 with the **(config-owner-content) port** command.
- Set the URL to /\* with the **(config-owner-content) url** command.
- Enable the content rule to be sticky based on SSL with the **(config-owner-content) advanced-balance ssl** command.
- Specify the SSL application type with the **(config-owner-content) application** command.

**Note**

---

You cannot configure the **application ssl** and **url url** commands on the same content rule.

---

---

**wap-msisdn**

Enables a Layer 5 content rule to stick a client to a server based on the MSISDN header of the HTTP request. MSISDN is the header field for wireless clients using the Wireless Application Protocol (WAP).

If the CSS finds the MSISDN header in an HTTP request, the CSS generates a key based on the value in the header field. The CSS uses the key to look up an entry in the sticky table. If the entry exists, the CSS sends the client to the sticky server indicated by the table entry. If the entry does not exist, the CSS creates a new sticky entry, hashes the MSISDN value into a key, and saves the key in the entry.

If the CSS does not find the MSISDN header, the CSS load balances the client request based on the balance method configured through the **(config-owner-content) balance** command.

You can use the **advanced-balance wap-msisdn** command alone or with the MSISDN header field type. Refer to the **(config-header-field-group) header-field** command.

---



---

**Command Modes**

Owner-Content

---

**Related Commands**

**show rule**  
**(config-owner-content) sticky-mask**  
**(config-owner-content) string range**

## (config-owner-content) application

To specify the application type associated with the content rule, use the **application** command. The application type enables the CSS to correctly interpret the data stream matching the content rule and parse them. Otherwise, the data stream packets are rejected. Use the **no** form of this command to reset the application type to its default setting of HTTP.

**application** *type*  
**no application**

---

### Syntax Description

<i>type</i>	The application type. Enter one of the following: <ul style="list-style-type: none"><li>• <b>bypass</b>, to bypass the matching of the content rule and send the request directly to the origin server.</li><li>• <b>http</b> (default), to process HTTP data streams.</li><li>• <b>ftp-control</b>, to process FTP data streams.</li><li>• <b>realaudio-control</b>, to process RealAudio Control data streams.</li><li>• <b>ssl</b>, to process Secure Socket Layer (SSL) protocol data streams.</li></ul>
-------------	--

---

---

### Usage Guidelines

You cannot configure the **application ssl** and **url urql** commands on the same content rule.

Define an application type for non-standard ports.

## (config-owner-content) arrowpoint-cookie

To configure the ArrowPoint cookie path and expiration, use the **arrowpoint-cookie** command.

- **arrowpoint-cookie browser-expire...**, allows the browser to expire the cookie
- **arrowpoint-cookie expiration...**, sets an expiration time which the CSS compares with the time associated with the cookie
- **arrowpoint-cookie expire-services...**, expires the service information when the cookie expires
- **arrowpoint-cookie path...**, sets the cookie path to a configured path

For more information about these options, refer to the following commands.

---

### Usage Guidelines

The CSS transparently generates the ArrowPoint cookie for the client, the client stores it and returns it in subsequent requests, and the CSS later uses it to maintain the client-server stickiness. The cookie contains the sticky information itself and does not refer to a sticky table.

If you configure the arrowpoint-cookie method in the content rule, the CSS always check for the existence of the ArrowPoint cookie when it receives a client request. If this cookie does not exist, the CSS performs a server load balance and generates an ArrowPoint cookie. When the CSS finds the cookie in the client request, it unscrambles the cookie data and then validates it. Then, the CSS checks the cookie expiration time. If the cookie has expired, the CSS sends a new cookie the information about the server where the client was stuck. This will allow for the appearance of an uninterrupted connection.

If the cookie format is valid, the CSS ensures the consistency between the cookie and the CSS configuration. When all the validations are passed, the CSS forwards the client request to the server indicated by the server identifier. Otherwise, the CSS treats the request as an initial request.

If the cookie is valid but the sticky server is not available, the CSS uses the sticky-serverdown-failover configuration to handle the request. If the configured sticky-serverdown-failover type is **balance**, then the CSS treats the client request as an initial client request without the ArrowPoint cookie, runs through the load balance algorithm to choose a server, and then redirects with a generated ArrowPoint cookie. If the sticky-serverdown-failover type is **redirect**, the CSS redirects the request to the specified URL.

---

**Related Commands**

**show rule**  
**(config-owner-content) advanced-balance**

## arrowpoint-cookie browser-expire

To allow the browser to expire the ArrowPoint cookie based on the expiration time, use the **arrowpoint-cookie browser-expire** command. To configure the expiration time, refer to the **arrowpoint-cookie expiration** command. Use the **no** form of this command to allow the CSS to expire the cookie.

**arrowpoint-cookie browser-expire**  
**no arrowpoint-cookie browser-expire**

---

**Command Modes**

Owner-Content

---

**Usage Guidelines**

When the cookie expires, all sticky information is lost.

---

**Related Commands**

**arrowpoint-cookie expiration**

## arrowpoint-cookie expiration

To set the cookie duration time which the CSS compares with the time associated with the ArrowPoint cookie, use the **arrowpoint-cookie expiration** command. When the cookie time exceeds the duration time, the CSS expires the cookie. To allow the browser to expire the cookie, use the **arrowpoint-cookie browser-expire** command. Use the **no** form of this command to expire the cookie when you close the browser.

**arrowpoint-cookie expiration *dd:hh:mm:ss***  
**no arrowpoint-cookie expiration**

Syntax Description		
	<i>dd</i>	The number of days. Valid numbers are from 00 to 99.
	<i>hh</i>	The number of hours. Valid numbers are from 00 to 99.
	<i>mm</i>	The number of minutes. Valid numbers are from 00 to 99.
	<i>ss</i>	The number of seconds. Valid numbers are from 00 to 99.

**Command Modes** Owner-Content

**Usage Guidelines** If the cookie has expired, the CSS sends a new cookie that includes the server where the client was stuck. This will allow for the appearance of an uninterrupted connection.

**Examples** For example, to set an expiration time of 2 days, 3 hours, 21 minutes and 0 seconds, enter:

**arrowpoint-cookie expiration 02:03:21:00**

**Related Commands** **arrowpoint-cookie browser-expire**

## arrowpoint-cookie expire-services

To expire service information when the cookie expires before sending a new cookie, use the **arrowpoint-cookie expire-services** command. By default, when the cookie expires, the CSS sends a new cookie with the server information from the expired cookie. Use the **no** form of this command to reset the default behavior.

**arrowpoint-cookie expire-services**  
**no arrowpoint-cookie expire-services**

---

**Command Modes**      Owner-Content

## arrowpoint-cookie path

To set the ArrowPoint cookie path to a configured path, use the **arrowpoint-cookie path** command. Otherwise, the CSS sets the default path attribute of the cookie to `"/`". Use the **no** form of this command to reset the cookie path to its default of `"/`".

**arrowpoint-cookie path "path\_name"**  
**no arrowpoint-cookie path**

---

<b>Syntax Description</b>	<i>path_name</i>	The pathname where you want to send the cookie. Enter a quoted text string with a maximum of 99 characters. The default path of the cookie is <code>"/</code> ".
---------------------------	------------------	--

---



---

**Command Modes**      Owner-Content

---

**Related Commands**      (config-owner-content) **advanced-balance arrowpoint-cookie**

## (config-owner-content) balance

To specify the load-balancing algorithm for the content rule, use the **balance** command. Use the **no** form of this command to reset the load balancing algorithm to its default setting of roundrobin.

```
balance [acaldestip|domain|domainhash|leastconn|roundrobin|srcip|url
|urlhash|weightedrr]
no balance
```

### Syntax Description

<b>aca</b>	ArrowPoint Content Awareness algorithm. The CSS correlates content request frequency with the server's cache sizes to improve cache hit rates for that server.
<b>destip</b>	Destination IP address division. The CSS directs all client requests with the same destination IP address to the same service.
<b>domain</b>	Domain name division. The CSS uses the domain name in the request URI to direct the client request to the appropriate service.
<b>domainhash</b>	Internal CSS hash algorithm based on the domain string. The CSS uses the algorithm to hash the entire domain string. Then, the CSS uses the hash result to choose the server.
<b>leastconn</b>	Least connections. The CSS chooses a running service that has the least number of connections.  We do not recommend that you use UDP content rules with the leastconn load-balancing algorithm. The service connection counters do not increment and remain at 0 because UDP is a connectionless protocol. Because the counters remain at 0, the CSS will give inconsistent results.
<b>roundrobin</b>	Roundrobin algorithm (default).
<b>srcip</b>	Source IP address division. The CSS directs all client requests with the same source IP address to the same service.
<b>url</b>	URL division. The CSS uses the URL (omitting the leading slash) in the redirect URL to direct the client requests to the appropriate service.

---

<b>urlhash</b>	Internal CSS hash algorithm based on the URL string. The CSS uses the algorithm to hash the entire URL string. Then, the CSS uses the hash result to choose the server.
<b>weightedrr</b>	Weighted roundrobin algorithm. The CSS uses the roundrobin algorithm but weighs some services more heavily than others. You can configure the weight of a service when you add it to the rule.

---

---

**Usage Guidelines**

Before you can change the balance method, you must suspend the rule.

---

**Related Commands**

**show rule**  
**(config-owner-content) add service**  
**(config-owner-content) advanced-balance**  
**(config-owner-content) string operation**

## **(config-owner-content) dns-disable-local**

To disable DNS on the content rule, use the **dns-disable-local** command. The CSS informs other CSSs through APP that the services related to the content rule are not available for DNS activities. However, the services remain active for other activities.

Use the **no** form of this command to enable DNS on the content rule.


**dns-disable-local**  
**no dns-disable-local**

## (config-owner-content) dnsbalance

To determine where to resolve a request for a domain name into an IP address, use the **dnsbalance** command. By default, the content rule will use the DNS load balancing method assigned to the owner.

Use the **no** form of this command to reset the DNS load balancing method to its default setting of using the method assigned to the owner.

```
dnsbalance [leastloaded|preferlocal|roundrobin|useownerdnsbalance]  
no dnsbalance
```

Syntax Description		
	<b>leastloaded</b>	Resolves the request to the least-loaded local or remote domain site. The CSS first compares load numbers. If the load number between domain sites is within 50, then the CSS compares their response times. The site with the faster response time is considered the least loaded site.
		 <p><b>Note</b> For the <b>leastloaded</b> option to work properly, all domain sites must be running a minimum of CSS software version 3.02.</p>
	<b>preferlocal</b>	Resolves the request to a local VIP address. If all local systems exceed their load threshold, the CSS chooses the least-loaded remote system VIP address as the resolved address for the domain name.
	<b>roundrobin</b>	Resolves the request by evenly distributing the load to resolve domain names amongst content domain sites, local and remote. The CSS does not include sites that exceed their local load threshold.
	<b>useownerdnsbalance</b>	Resolves the request by using the DNS load balancing method assigned to the owner. This is the default method for the content rule. If you do not implicitly set an owner method, the CSS uses the default owner DNS load balancing method of roundrobin.

**Usage Guidelines**

Before you can change the DNS balance method, you must suspend the rule.

**Related Commands**

(config-owner) dnsbalance

**(config-owner-content) failover**

To define what will happen to content requests when a service fails or is suspended, use the **failover** command. For the CSS to use this setting, you must configure keepalive for each service. Use the **no** form of this command to reset the failover for the content rule to its default setting of linear. The linear failover method distributes the content requests to the failed service evenly between the remaining services.

```
failover [bypass|next]
no failover
```

**Syntax Description**

<b>bypass</b>	Bypasses all services and send the content request directly to the origin service.
<b>next</b>	Sends the content requests to the service next to the failed service. The CSS selects which service is next to the failed one by referring to the order in which the services were configured.

**Usage Guidelines**

If you remove a service, the CSS rebalances the remaining services. The CSS does not use the **failover** command setting.

Before you can change the failover method, you must suspend the rule.

**Related Commands**

(config-owner-content) balance  
(config-service) keepalive

## (config-owner-content) header-field-rule

To associate a header-field group to a content rule, and optionally assign a weight value to the header-field group, use the **header-field-rule** command. Use the **no** form of this command to remove the header-field group from the rule.

```
header-field-rule name {weight number}
no header-field-rule
```

### Syntax Description

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

### Usage Guidelines

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.

Before you can change the header-field group, you must suspend the rule.

### Related Commands

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

## (config-owner-content) hotlist

To enable the hotlist for the content rule and configure hotlist parameters, use the **hotlist** command. A hotlist lists the most requested content during a user-defined period of time. The options for this content mode command are:

- **hotlist**, enables the hotlist for the content rule
- **hotlist interval...**, sets the hotlist refresh interval
- **hotlist size...**, sets the size of the hotlist
- **hotlist threshold...**, sets the hotlist threshold
- **hotlist type...**, sets the hotlist type to hit count

For more information on these options, refer to the following commands.

### hotlist

To enable the hotlist for the content rule, use the **hotlist** command. Use the **no** form of this command to disable the hotlist for the content rule.

```
hotlist
no hotlist
```

---

**Command Modes**

Owner-Content

## hotlist interval

To set the hotlist refresh interval, use the **hotlist interval** command. Use the **no** form of this command to reset the hotlist interval for the content rule to its default setting of 1 minute.

**hotlist interval** *time*  
**no hotlist interval**

<b>Syntax Description</b>	<i>time</i>	The interval, in minutes, for refreshing the hotlist. Enter an integer from 1 to 60. The default is 1.
---------------------------	-------------	--

<b>Command Modes</b>	Owner-Content
----------------------	---------------

## hotlist size

To set the size of the hotlist, use the **hotlist size** command. Use the **no** form of this command to reset the hotlist size for the content rule to its default setting of 10 entries.

**hotlist size** *entries*  
**no hotlist size**

<b>Syntax Description</b>	<i>entries</i>	The total number of hotlist entries that is maintained for the rule. Enter an integer from 1 to 100. The default is 10.
---------------------------	----------------	---

<b>Command Modes</b>	Owner-Content
----------------------	---------------

## hotlist threshold

To set the hotlist threshold, use the **hotlist threshold** command. Use the **no** form of this command to reset the hotlist threshold for the content rule to its default setting of 0.

```
hotlist threshold threshold  
no hotlist threshold
```

<b>Syntax Description</b>	<i>threshold</i>	The threshold below which an item is not considered hot. Enter an integer from 0 to 65535. The default is 0.
---------------------------	------------------	--

<b>Command Modes</b>	Owner-Content
----------------------	---------------

## hotlist type

To set the hotlist type, use the **hotlist type** command. Currently, the CSS supports only the hit count hotlist type, which is the default setting. Hit count is the number of times that the content is accessed. Use the **no** form of this command to reset the hotlist type for the content rule to its default setting of hitCount.

```
hotlist type hitCount  
no hotlist type
```

<b>Command Modes</b>	Owner-Content
----------------------	---------------

## (config-owner-content) load-threshold

To set the normalized load threshold for the availability of each local service on the content rule, use the **load-threshold** command. When the service load metric exceeds this threshold, the local service becomes unavailable and is redirected to the remote services.

Use the **no** form of this command to reset the load threshold for the content rule to its default setting of 254.

**load-threshold** *threshold*  
**no load-threshold**

<b>Syntax Description</b>	<i>threshold</i>	The maximum load. Enter an integer from 2 through 254. The default is 254, which is the maximum load. To view the load on services, enter:  <b>show service</b>
---------------------------	------------------	---

## (config-owner-content) no

To negate a command or set it to its default for the content rule, use the **no** command. For information on general **no** commands you can use in this mode, refer to the general **no** command. The following options are available in content mode.

<b>Syntax Description</b>	<b>no acl</b> <i>index</i>	Deletes an ACL
	<b>no advanced-balance</b>	Disables the advanced balancing method for the content rule
	<b>no application</b>	Resets the application type to its default setting of HTTP
	<b>no arrowpoint-cookie browser-expire</b>	Resets the cookie expiration method to the time configured through the <b>arrowpoint-cookie expiration</b> command
	<b>no arrowpoint-cookie expiration</b>	Resets the expiration time to one year after the timestamp on the cookie

<b>no arrowpoint-cookie expire-services</b>	Resets the default behavior of sending a new ArrowPoint cookie with the server information from the expired cookie
<b>no arrowpoint-cookie path</b>	Resets the ArrowPoint cookie path to its default of “/”
<b>no balance</b>	Resets the load balancing algorithm to its default setting of roundrobin
<b>no dns-disable-local</b>	Enables DNS on the content rule
<b>no dnsbalance</b>	Resets the DNS load balancing method to its default setting of using the method assigned to the owner
<b>no failover</b>	Resets the failover to its default setting of linear
<b>no header-field-rule</b>	Removes the header-field group from the content rule
<b>no hotlist</b>	Disables the hotlist for the content rule
<b>no hotlist interval</b>	Resets the hotlist interval to its default setting of 1minute
<b>no hotlist size</b>	Resets the hotlist size to its default setting of 10 entries
<b>no hotlist threshold</b>	Resets the hotlist threshold to its default setting of 0 entries
<b>no hotlist type</b>	Resets the hotlist type to its default setting of hit count
<b>no load-threshold</b>	Resets the load threshold to its default setting of 254
<b>no owner</b> <i>existing_owner_name</i>	Deletes an existing owner
<b>no persistent</b>	Disables persistence
<b>no port</b>	Resets the port number to its default value of 0
<b>no primarySorryServer</b>	Removes the primary sorry service from the rule
<b>no protocol</b>	Resets the protocol for the content rule to its default of <b>any</b>
<b>no redirect</b>	Deletes the redirect URL
<b>no redundancy-l4-stateless</b>	Disables stateless redundancy failover

<b>no rmon-event</b> <i>index</i>	Deletes an RMON event
<b>no rmon-history</b> <i>index</i>	Deletes an RMON history
<b>no secondarySorryServer</b>	Removes the secondary sorry service from the rule
<b>no sticky-inact-timeout</b>	Disables the sticky connection inactivity timeout feature
<b>no sticky-mask</b>	Resets the sticky mask to 255.255.255.255
<b>no sticky-no-cookie-found-action</b>	Resets sticky-no-cookie-found-action to the default of loadbalance
<b>no sticky-serverdown-failover</b>	Sets the sticky server failover method to its default setting of using the configured load balancing method
<b>no string ascii-conversion</b>	Enables the ASCII conversion of escaped special characters within the specified sticky range before applying any processing to the string
<b>no string eos-char</b>	Clears the end of string characters as the delimiters for the sticky string
<b>no string operation</b>	Resets the string operation to choose a server by matching a service cookie in the sticky string
<b>no string prefix</b>	Clears the string prefix
<b>no string process-length</b>	Resets the number of bytes that the string operation will use to its default of 0
<b>no string range</b>	Resets the string range within a cookie, URL, or URL extension from a client to its default setting of 1 to 100
<b>no string skip-length</b>	Resets the number of bytes to skip after the end of a prefix to find the string result to its default of 0
<b>no url</b>	Removes the URL for the content rule
<b>no vip address</b>	Clears the VIP address for the content rule

## (config-owner-content) param-bypass

To enable content requests to bypass transparent caches when the CSS detects special terminators in the requests, use the **param-bypass** command. These terminators include "#" and "?" which indicate that the content is dependent on the arguments that follow the terminators. Because the content returned by the server is dependent on the content request itself, the returned content is not cacheable.

**param-bypass** [**disable**|**enable**]

Syntax Description		
	<b>disable</b>	Content requests with special terminators do not bypass transparent caches. This is the default setting.
	<b>enable</b>	Content requests with special terminators bypass transparent caches and are forwarded to the origin server.
Related Commands	<b>show rule</b>	

## (config-owner-content) persistent

To maintain a persistent connection with a server, use the **persistent** command. By default, persistence is enabled. Use the **no** form of this command to disable persistence.

**persistent**  
**no persistent**

### Usage Guidelines

In content rule persistence, the CSS keeps the client on the same service connection specified by the content rule for an entire flow session as long as a new content request:

- Matches on the same content rule that specified the current service
- Matches on a new content rule that contains the current service, even if a different best service is specified by the content rule
- Does not match on a content rule, but a previous content rule match connected the client to the current service

If you are using transparent caches (which prefetch content) or mirrored-content servers, this scheme works well because the same content is available on each service.



### Note

If a request for content on a persistent connection matches on a new content rule that does not contain the current service, or persistence is disabled and there is a better service configured in the content rule, the CSS redirects or remaps the current connection to a new best service based on the setting of the **(config) persistent reset** command, if configured. If you do not configure persistence reset, the CSS performs an HTTP redirect by default.

Disabling persistence allows the CSS to move a connection to a better service on the same rule or to utilize cache bypass functionality (EQLs or failover bypass). Disable persistence on a content rule with:

- A balance method of domain or domain hash when using proxy caches
- A balance method of url or urlhash when using transparent caches

- A failover method of bypass when using transparent caches
- An EQL bypass with a transparent cache
- Adding a sorry server to a content rule

**Related Commands** (config) bypass persistence  
(config) persistence reset

## (config-owner-content) port

To specify the content rule's TCP/UDP port number, use the **port** command. The port number for a content rule is the port number used by incoming requests for the content governed by the rule. Use the **no** form of this command to reset the port number to its default value of 0.

**port** *number*  
**no port**

<b>Syntax Description</b>	<i>number</i>	<p>The TCP or UDP incoming port number associated with the content rule. Enter an integer from 0 to 65535. The default value is 0.</p> <p>To use stickiness based on the Secure Socket Layer (SSL) session ID, set the port to 443. Then, set the URL to /* with the <b>(config-owner-content) url</b> command, and enable stickiness with the <b>(config-owner-content) advanced-balance ssl</b> command. Then specify an SSL application type.</p>
---------------------------	---------------	--

**Usage Guidelines** Before you can change the port number, you must suspend the rule.

**Related Commands** show rule  
(config-owner-content) protocol

## (config-owner-content) primarySorryServer

To configure the primary sorry service for the content rule, use the **primarySorryServer** command. A sorry service is a server that is used for content requests when all other services are unavailable. You can configure the service to contain content, or to provide a drop or redirect message. The service is not used in load balancing. Use the **no** form of this command to remove a primary sorry service.

```
primarySorryServer service_name
no primarySorryServer
```

---

### Syntax Description

<i>service_name</i>	The name of the existing service. Enter the name as a case-sensitive unquoted text string with no spaces.
---------------------	---

---



---

### Usage Guidelines

Do not configure a sorry server on a content rule used for matching non-cacheable content.

---

### Related Commands

```
show rule
(config) service
(config-owner-content) secondarySorryServer
```

## (config-owner-content) protocol

To specify the content rule's IP protocol, use the **protocol** command. The protocol for a content rule is the protocol used by incoming requests for the content governed by the rule. Use the **no** form of this command to reset the protocol to its default value of **any**.

```
protocol [any|tcp|udp]  
no protocol
```

---

### Syntax Description

<b>any</b>	The content rule uses any protocol. This is the default protocol.
<b>tcp</b>	The content rule uses the TCP protocol suite.
<b>udp</b>	The content rule uses the UDP protocol suite.

---

### Usage Guidelines

Before you can change the protocol, you must suspend the rule.

---

### Related Commands

**show rule**  
**(config-owner-content) port**

## (config-owner-content) redirect

To set HTTP status code 302 for the content rule, use the **redirect** command. Use the **no** form of this command to delete the redirect URL.

```
redirect "url"  
no redirect
```

---

### Syntax Description

<i>url</i>	The URL to send with HTTP status code 302. Enter a quoted text string and a maximum length of 64 characters.
------------	--

---

---

### Usage Guidelines

The **redirect** command makes the content at its current address unavailable to subsequent requests, and provides a message to send with the status code back to the requestor. Specify a message that returns the alternate location of the content governed by the rule.

---

### Related Commands

**show rule**

## (config-owner-content) redundancy-l4-stateless

To enable the Stateless Redundancy Failover feature for a content rule on a redundant CSS, use the **redundancy-l4-stateless** command. The CSS can set up a connection for a mid-stream TCP flow, allowing TCP traffic to continue when a failure occurs at the load-balancing CSS. By default, the CSS rejects TCP sessions that do not begin with a TCP/SYN frame. Use the **no** form of this command to reset the default behavior of the CSS.

**redundancy-l4-stateless**  
**no redundancy-l4-stateless**

### Command Modes

Owner-content configuration mode

### Usage Guidelines

The Stateless Redundancy Failover feature has specific environment and configuration requirements. The environment requirements are as follows:

- Layer 3 and Layer 4 content rules with a VIP address. This feature is not supported in Layer 5 configurations.
- Source IP address load balance method only.
- CSS-to-CSS identical server and content rule configuration including:
  - Content VIP address.
  - Content balance method.
  - Failover method.
  - Service IP address, number, and order. The CSS orders services alphabetically. Apply identical service names on the master and backup CSSs.
- Visibility of identical servers to keepalive traffic from CSS to CSS. This ensures that the redistribution of the balance method does not occur in a failover event.

Redundant routes in a high availability topology surrounding the CSS are supported. However, the topology must not balance packets in a TCP/IP socket connection across more than one Ethernet port on the CSS.

IP and VIP redundant configurations are supported. The configuration requirement for each server farm is synchronization across all CSSs of:

- Membership and IP addresses of the server farms.
- Content rule VIP address. Each CSS must share the content VIP address that is used as a balance point for the server farm.
- Source group VIP address. Define each CSS with a source group VIP address as the content VIP address to NAT source addresses for packets returning from the server. In case of a failover, the source group handles connection setups for TCP/IP retransmissions that arrive at the CSS from a server. All servers on the farm must be a member of the source group.

Do not configure source groups for outbound traffic from the servers because the backup CSS does not know which ports were mapped by the source group on the master CSS.

For more detailed information on Stateless Redundancy Failover, refer to the *Content Services Switch Advanced Configuration Guide*.

---

**Related Commands**

**show redundancy**  
**(config) ip redundancy**  
**(config) group**  
**(config) interface**  
**(config) service**  
**(config-group) redundancy-l4-stateless**  
**(config-owner) content**

## (config-owner-content) remove

To remove either a DNS name or an existing service from the content rule, use the **remove** command.

```
remove [dns dns_name|service service_name]
```

### Syntax Description

<b>dns</b>	Removes the DNS name from the content rule.
<i>dns_name</i>	The DNS name. Enter the name as a case-sensitive unquoted text string with no spaces and a maximum length of 32 characters. To see a list of DNS names, enter:  <b>remove dns ?</b>
<b>service</b>	Removes an existing service from the content rule. Removing a service removes it from the resource pool that the CSS uses for balancing the load of requests for the content governed by the rule. The CSS rebalances the remaining services. It does not use the <b>failover</b> command setting.
<i>service_name</i>	The name of the existing service. Enter the name as a case-sensitive unquoted text string with no spaces and a maximum length of 32 characters. To see a list of service names, enter:  <b>remove service ?</b>

### Related Commands

**show rule**  
**(config-owner-content) add**

## (config-owner-content) secondarySorryServer

To configure the secondary sorry service for the content rule, use the **secondarySorryServer** command. Use the **no** form of this command to remove a secondary sorry service.

```
secondarySorryServer service_name
no secondarySorryServer
```

---

### Syntax Description

*service\_name*

The name of the existing service. Enter the name as a case-sensitive unquoted text string with no spaces.

---



---

### Usage Guidelines

A secondary sorry service is a backup service that is used when the primary sorry service is unavailable. You can configure the service to contain content, or to provide a drop or redirect message. The service is not used in load balancing.

Do not configure a sorry server on a content rule used for matching non-cacheable content.

---

### Related Commands

**show rule**  
**(config) service**  
**(config-owner-content) primarySorryServer**

## (config-owner-content) show rule-header-field

To display information about the header-field rule and group associated with a content rule, use the **show rule-header-field** command.

### show rule-header-field

---

#### Examples

To display information about the header-field rule and group associated with a content rule, enter:

```
(config-owner-content[test-rule1])# show rule-header-field
header-field-rule: palmpilot.
  lookup pass: 0, lookup fail: 0.

header field group : palmpilot           Description: Palm Pilot
users
  header-field 1 user-agent contain "PalmPilot"
```

---

#### Related Commands

(config-owner-content) header-field-rule

## (config-owner-content) sticky-inact-timeout

To specify the inactivity timeout period on a sticky connection for a content rule before the CSS removes the sticky entry from the sticky table, use the **sticky-inact-timeout** command. Use the **no** form of this command to disable the sticky connection inactivity timeout feature.

**sticky-inact-timeout** *minutes*  
**no sticky-inact-timeout**

---

### Syntax Description

<i>minutes</i>	The number of minutes of inactivity. Enter a number from 0 to 65535. The default value is 0, which means this feature is disabled. When disabled, the CSS does not remove the entry from the table until the sticky table is full. When table is full, the CSS recycles the least used sticky entry first.
----------------	--

---



---

### Usage Guidelines

When you configure the inactivity timeout period, the CSS keeps the sticky entry in the sticky table for the specified amount of time. The CSS does not reuse the entry until the time expires. If the sticky table is full and none of the entries has expired, the CSS rejects the new sticky request. When the sticky connection expires, the CSS uses the configured load balance method to choose an available server for the request.

When this feature is disabled, the new sticky connection uses the oldest used sticky entry. A sticky association could exist for a time depending on the sticky traffic load on the CSS.

---

### Related Commands

**show rule**

## (config-owner-content) sticky-mask

To mask a group of client IP addresses in order to preserve the client connection state when the client's source IP address changes, use the **sticky-mask** command. Use the **no** form of this command to reset the default sticky mask of 255.255.255.255.

```
sticky-mask subnet_mask  
no sticky-mask
```

---

### Syntax Description

---

<i>subnet_mask</i>	The subnet mask used for stickiness. Enter the IP mask in dotted-decimal format (for example, 255.255.255.0). The default is 255.255.255.255.
--------------------	---

---

---

### Usage Guidelines

The client's source IP address change occurs when a client-server connection is lost and the client sends a different IP address. The CSS needs to reconnect the client to the same server that is preserving the client information.

---

### Related Commands

**show rule**  
**(config-owner-content) advanced-balance**  
**(config-owner-content) string range**

## (config-owner-content) sticky-no-cookie-found-action

To specify the action the CSS should take for a sticky cookie content rule when it cannot locate the cookie header or the specified cookie string in the client request, use the **sticky-no-cookie-found-action** command. Use the **no** form of this command to reset sticky-no-cookie-found-action to the default of loadbalance.

```
sticky-no-cookie-found-action [loadbalance|redirect
  "URL"|reject|service name]
no sticky-no-cookie-found-action
```

Syntax Description		
<b>loadbalance</b>		The CSS uses the configured balanced method when no cookie is found in the client request. This option is the default setting.
<b>redirect</b> "URL"		The CSS redirect the client request to a specified URL string when no cookie found in the client request. When using this option, you must also specify a redirect URL. Enter the redirect <i>URL</i> as a quoted text string from 0 to 64 characters.
<b>reject</b>		The CSS rejects the client request when no cookie is found in the request.
<b>service name</b>		The CSS sends the no cookie client request to the specified service when no cookie is found in the request.

## (config-owner-content) sticky-serverdown-failover

To define what will happen when a sticky string is found but the associated service has failed or is suspended, use the **sticky-serverdown-failover** command. Use the **no** form of this command to set the sticky failover method to its default setting of using the configured load balancing method.

### sticky-serverdown-failover

[balance|redirect|reject|sticky-srcip|sticky-srcip-dstport]

no sticky-serverdown-failover

### Syntax Description

<b>balance</b>	Sets the failover method to use a service based on the configured load balancing method.
<b>redirect</b>	Sets the failover method to use a service based on the currently configured redirect string. If a redirect string is not configured, the load balancing method is used.
<b>reject</b>	Rejects the content request.
<b>sticky-srcip</b>	Sets the failover method to use a service based on the client IP address. This is dependent on the sticky configuration.
<b>sticky-srcip-dstport</b>	Sets the failover method to use a service based on the client IP address and the server destination port. This is dependent on the sticky configuration.

### Related Commands

(config-owner-content) **balance**  
 (config-owner-content) **redirect**

## (config-owner-content) string

To set string criteria to derive string results and the method to choose a destination server for the result, use the **string** command and its options. The string result is a sticky string in the cookie header, URL, or URL extension based on a sticky type being configured.

The options for this content mode command are:

- **string ascii-conversion**..., enables or disables the ASCII conversion of escaped special characters within the specified sticky range before applying any processing to the string
- **string eos-char**..., specifies the delimiters for the sticky string
- **string operation**..., specifies the method to choose a destination server for a string result
- **string prefix**..., specifies the string prefix located in the sticky range
- **string process-length**..., specifies how many bytes, after the end of the prefix designated by the **string prefix** command and skipping the bytes designated by the **string skip-length** command, that the string operation will use
- **string range**..., specifies the starting and ending byte positions within a cookie, URL, or URL extension from a client
- **string skip-length**..., specifies how many bytes to skip after the end of prefix to find the string result

For more information on these options, refer to the following commands.

## string ascii-conversion

To enable or disable the ASCII conversion of escaped special characters within the specified sticky range before applying any processing to the string, use the **string ascii-conversion** command. By default, ACSII conversion is enabled. Use the **no** form of this command to re-enable the ASCII conversion of special escaped characters.

```
string ascii-conversion [enable|disable]
no string ascii-conversion
```

---

### Command Modes

Owner-Content

---

### Related Commands

(config-owner-content) string range

## string eos-char

To specify up to three ASCII characters as the delimiters for the sticky string, use the **string eos-char** command. For example, in a cookie header, a “;” character is usually used as a delimiter; in a URL extension, a “&” character is often used as a delimiter. Use the **no** form of this command to clear the end of string characters.

```
string eos-char “characters”
no string eos-char
```

---

### Syntax Description

<i>character</i>	The end of string characters. Enter a quoted text string with a maximum of three characters.
------------------	--

---



---

### Command Modes

Owner-Content

**Usage Guidelines**

The CSS uses this command if the **string process-length** command is not configured; the **string process-length** command has higher precedence. If neither commands are configured, the CSS uses the maximum 64 bytes for the final string operation.

**Related Commands**

(config-owner-content) **string process-length**

**string operation**

To determine the method to choose a destination server for a string result, derived from the settings of the **string** criteria commands, use the **string operation** command. You can choose a server by using the configured balance method or by using the hash key generated by specified sticky hash type.

Use the **no** form of this command to reset the string operation to its default setting, choosing a server by matching a service cookie in the sticky string.

**string operation** [**match-service-cookie**|**hash-a**|**hash-crc32**|**hash-xor**]  
**no string operation**

**Syntax Description**

<b>match-service-cookie</b>	Choose a server by matching a service cookie in the sticky string. This is the default setting. When a match is not found, the server is chosen by using the configured balance method (for example, roundrobin).
<b>hash-a</b>	Choose a server by applying a basic hash algorithm on the hash string to generate the hash key.  If the selected server is out of service, the CSS performs a rehash to choose another server.

---

<b>hash-crc32</b>	Choose a server by applying the CRC32 algorithm on the hash string to generate a hash key.  If the selected server is out of service, the CSS performs a rehash to choose another server.
<b>hash-xor</b>	Choose a server by performing an Exclusive OR (XOR) each byte of the hash string to derive the final hash key.  If the selected server is out of service, the CSS performs a rehash to choose another server.

---

---

**Command Modes**

Owner-Content

---

**Related Commands**

(config-owner-content) string **ascii-conversion**  
(config-owner-content) string **eos-char**  
(config-owner-content) string **prefix**  
(config-owner-content) string **process-length**  
(config-owner-content) string **skip-length**

## string prefix

To specify the string prefix located in the sticky range, use the **string prefix** command. Use the **no** form of this command to clear the string prefix.

**string prefix** *“text”*  
**no string prefix**

<b>Syntax Description</b>	<i>text</i>	<p>The string prefix. Enter a quoted text string with a maximum of 30 characters. The default has no prefix (“”).</p> <p>If you do not configure the string prefix, the string functions start from the beginning of the cookie, URL, or URL extension, depending on the sticky type. If the string prefix is configured but is not found in the specified sticky range, load balancing defaults to the roundrobin method.</p>
<b>Command Modes</b>	Owner-Content	
<b>Related Commands</b>	<p>(config-owner-content) <b>advanced-balance</b>            (config-owner-content) <b>string range</b></p>	

## string process-length

To specify how many bytes, after the end of the prefix designated by the **string prefix** command and skipping the bytes designated by the **string skip-length** command, that the string operation will use, use the **string process-length** command. Use the **no** form of this command to set the number of bytes to its default setting of 0.

**string process-length** *bytes*  
**no string process-length**

<b>Syntax Description</b>	<i>bytes</i>	The number of bytes. Enter a number from 0 to 64. The default is 0.
---------------------------	--------------	---

<b>Command Modes</b>	Owner-Content
----------------------	---------------

<b>Usage Guidelines</b>	The <b>string process-length</b> command has higher precedence than the <b>string eos-char</b> command. If neither commands are configured, the CSS uses the maximum 64 bytes for the final string operation.
-------------------------	---

<b>Related Commands</b>	(config-owner-content) <b>string eos-char</b> (config-owner-content) <b>string operation</b> (config-owner-content) <b>string prefix</b> (config-owner-content) <b>string skip-length</b>
-------------------------	--

## string range

To specify the starting and ending byte positions within a cookie, URL, or URL extension from a client, use the **string range** command. Use the **no** form of this command to reset the range to its default setting of 1 to 100.

```
string range start_byte to end_byte
no string range
```

Syntax Description		
	<i>start_byte</i>	The starting byte position of the cookie, URL, or URL extension after the header. Enter an integer from 1 to 1999. The default is 1. Make sure that the starting byte position is less than the end byte.
	<i>end_byte</i>	The ending byte position of the cookie, URL, or URL extension. Enter an integer from 2 to 2000. The default is 100. Make sure that the ending byte position is more than the start byte.

**Command Modes** Owner-Content

**Usage Guidelines** By specifying the range of bytes, the CSS processes the information located only within the range. This limits the amount of information that the CSS has to process when examining each cookie, URL, or URL extension, enhancing its performance.



**Note**

If the starting position is beyond the cookie, URL or URL extension, the CSS does not perform the string function. When the ending position is beyond the cookie, URL or URL extension, the string processing stops at the end of the corresponding header.

**Related Commands** (config-owner-content) **advanced-balance**  
(config-owner-content) **sticky-mask**

## string skip-length

To specify how many bytes to skip after the end of the prefix to find the string result, use the **string skip-length** command. Use the **no** form of this command to set the number of bytes to its default setting of 0.

**string skip-length** *bytes*  
**no string skip-length**

---

<b>Syntax Description</b>	<i>bytes</i>	The number of bytes. Enter a number from 0 to 64. The default is 0.
---------------------------	--------------	---

---

---

<b>Command Modes</b>	Owner-Content
----------------------	---------------

---

<b>Related Commands</b>	(config-owner-content) string prefix
-------------------------	--------------------------------------

## (config-owner-content) suspend

To deactivate the content rule, denying access to the content governed by the rule, use the **suspend** command. Suspending a content rule does not affect existing flows to the content; it only applies to future requests for the content.

```
suspend
```

### Related Commands

```
show rule
(config-owner-content) active
```

## (config-owner-content) url

To specify the Uniform Resource Locator (URL) for the content, use the **url** command. Use the **no** form of this command to remove the URL or the URQL from the content rule.

```
url [{"url_name"}|["{url_path}/*"] [eql eql_name|dql dql_name
    {eql_name}]|urql urql_name]
no url {urql}
```

### Syntax Description

---

*url\_name*

The URL for the content. Enter a quoted text string with a maximum length of 252 characters. You must place a slash character (/) at the beginning of the URL, for example, “/files/test.gif”.



---

**Note** Do not include the ? or # parameter character in the URL string. The CSS terminates the URL at these parameter characters.

---

To specify a domain name, place two slashes at the beginning of the URL. For example, “//www.access.com/\*” allows the rule to match on the HTTP traffic that contains the www.access.com domain name in the HTTP host tag.

To specify certain wildcard operations for wildcard matching, use a “\*” character to specify a wildcard match. Examples of supported wildcards are:

- **/\*.html** which matches all requests with the .html extension
- **/newfiles/\*.jpg** which matches all requests for files beginning with /newfiles and have the .jpg extension.
- **/newfiles/\*** which matches all requests for files beginning with /newfiles.
- **/newfiles/1.jpg** which matches requests for the /newfile/1.jpg file only.

To use stickiness based on Secure Socket Layer (SSL) session ID, set the URL to /\*. Also, set the port to 443 with the **(config-owner-content) port** command and enable stickiness with the **(config-owner-content) advanced-balance ssl** command. Then specify an SSL application type.

---

<i>url_path</i>	<p>An optional path to any content file that has its file extension defined in the EQL. Enter a quoted text string. You must place:</p> <ul style="list-style-type: none"> <li>• A slash character (/) at the beginning of the path</li> <li>• /* characters at the end of the path</li> </ul> <p>An example is “/announcements/new/*”.</p> <p>To specify a domain name, place two slashes at the beginning of the URL.</p>
<b>eql</b>	Specifies the URL for any content file that has its file extension defined in an EQL or that has its domain defined in a DQL.
<i>eql_name</i>	<p>The name of the EQL. To see a list of EQLs, enter:</p> <p style="text-align: center;"><b>eql ?</b></p>
<b>dql</b>	Specifies the URL for any content file that has its file extension defined in an EQL or that has its domain defined in a DQL.
<i>dql_name</i>	<p>The name of the DQL. To see a list of DQLs, enter:</p> <p style="text-align: center;"><b>dql ?</b></p>
<b>urql</b>	Specifies a URQL consisting of a group of URLs to the content rule.
<i>urql_name</i>	<p>The name of the URQL. You can only assign one URQL per rule. To see a list of URQLs, enter:</p> <p style="text-align: center;"><b>urql ?</b></p>

---

**Usage Guidelines**

Before you can change the URL for the content rule, you must suspend the rule and you must remove the current URL.

When you configure content replication and staging, you must configure an URL or URQL in a content rule to define which files you want replicated:

- Use an URL to specify files.
- Use an URQL to define a static list of files.

Then add the subscriber services to the content rule.

If you want all files in all directories replicated, you do not need to create a content rule. Create a content rule to specify only those files you want replicated.

You cannot configure the **application ssl** and **url urql** commands on the same content rule.

**Note**

---

For caching environments, you can configure a domain content rule by placing two slash characters (*//*) at the front of the *url\_name* or *url\_path*. The rule matches HTTP traffic that contains the domain name in the HTTP host tag.

---

---

**Related Commands**

**show content**  
**show rule**  
**(config) eql**  
**(config) urql**

## (config-owner-content) vip address

To specify the content rule virtual IP (VIP) address or a range of addresses, use the **vip address** command. The VIP address for a content rule is the IP address used by incoming requests for the content governed by the rule. Use the **no** form of this command to clear the VIP address.

```

vip address ip_or_host { range number }
no vip address

```

<b>Syntax Description</b>	<i>ip_or_host</i>	The IP address or name for the content rule. Enter the address in either dotted-decimal IP notation (for example, 192.168.11.1) or mnemonic host-name format (for example, myhost.mydomain.com).
	<b>range</b> <i>number</i>	The <b>range</b> option allows you to specify a range of IP addresses starting with the VIP address ( <i>ip_or_host</i> ). Enter a number from 1 to 65535. The default range is 1.  For example, if you enter an IP address of 203.1.1.1 with a range of 10, the VIP addresses range from 203.1.1.1 through 203.1.1.10.
<b>Related Commands</b>	show rule	

## (config-owner-content) zero

To set the counters for the current content rule or all content rules to zero, use the **zero** command.

```

zero { all }

```

<b>Syntax Description</b>	<b>all</b>	Zeros the counters for all content rules
<b>Related Commands</b>	show rule	