CHAPTER 12

Configuring Application Acceleration and Optimization

Revised Date: 2/18/09

With application acceleration and optimization features on ACE appliances, you can configure application delivery and application acceleration options that increase productivity and efficiency. The application acceleration features optimize network performance and improve access to critical business information. This capability accelerates the performance of Web applications, including customer relationship management, portals, and online collaboration by up to 10 times.

This section includes:

Optimization Overview

The application acceleration functions of the ACE appliance apply several optimization technologies to accelerate application performance. This functionality enables enterprises to optimize network performance and improve access to critical business information.

The ACE appliance provides the following application acceleration and optimization functionality:

- Delta optimization eliminates redundant traffic on the network by computing and transmitting only the changes that occur in a Web page between successive downloads of the same page or similar pages.
- FlashForward object acceleration technology eliminates network delays associated with embedded Web objects able to be cached. such as images, style sheets, and JavaScript files by placing the responsibility for validating object freshness on the ACE appliance, rather than on the client, making the client more efficient.
- Just-in-time object acceleration enables acceleration of non-cacheable embedded objects, resulting in improved application response time by eliminating the need for clients to download these objects on each request.

- Smart image optimization automatically reduces image file sizes while optimizing image quality, resulting in faster image download times, faster page renders, and more efficient bandwidth utilization.
- Smart redirect enables the ACE appliance to automatically and transparently convert HTML META tag-based redirections into more efficient HTTP header-based redirections in order to eliminate the need for unnecessary freshness validation requests.
- Adaptive dynamic caching accelerates enterprise application performance and improves server system scalability by enabling the ACE appliance itself to fulfill requests for dynamic content, which offloads application servers and databases.

Refer to Configuring Application Acceleration and Optimization, page 12-1 or the *Cisco 4700 Series Application Control Engine Appliance Application Acceleration and Optimization Configuration Guide* for more information about application acceleration and optimization.

Related Topics

- Optimization Traffic Policies and Typical Configuration Flow, page 12-2
- Configuring Traffic Policies for HTTP Optimization, page 12-13
- Configuring Global Application Acceleration and Optimization, page 12-16

Optimization Traffic Policies and Typical Configuration Flow

To define the different optimization and application acceleration functions that you want the ACE appliance to perform, you must configure at least one each of the following:

- HTTP optimization action list—This action list specifies the actions that the ACE is to perform for application acceleration and optimization. You can configure action lists when configuring a virtual server, or as a separate procedure. See:
  - Configuring Application Acceleration and Optimization, page 4-37
  - Configuring Action Lists for Application Acceleration and Optimization, page 12-3
- Layer 7 server load-balancing class map—This class map identifies the Layer 7 server load-balancing match criteria to apply to incoming traffic, such as URL, HTTP cookie, HTTP header, or source IP address. See Configuring Virtual Context Policy Maps, page 11-30
- Layer 7 HTTP optimization policy map—This policy map applies the HTTP optimization action list and optionally an optimization parameter map to Layer 7 HTTP traffic. See Configuring Virtual Context Policy Maps, page 11-30.
- Layer 3 and Layer 4 class map—By using match criteria, this class map identifies the network traffic that can pass through the ACE appliance. The match criteria includes the VIP address for the network traffic. The ACE appliance uses these Layer 3 and Layer 4 traffic classes to perform server load balancing. See Configuring Virtual Context Policy Maps, page 11-30.
- Layer 3 and Layer 4 policy map—This policy map associates server load-balancing actions and HTTP optimization action lists with the VIP. See Setting Policy Map Rules and Actions for Layer 3/Layer 4 Network Traffic, page 11-38 and Configuring Traffic Policies for HTTP Optimization, page 12-13.
- Layer 7 server load-balancing policy map—This policy map specifies the server load-balancing actions that the ACE appliance is to perform. See Configuring Virtual Context Policy Maps, page 11-30.

You can also configure:

- Optimization parameter maps—Optimization parameter maps allow you to configure specific options for action list items. You can configure optimization parameter maps when configuring a virtual server or as a separate procedure.

  When you configure a parameter map with an action list for a class map, the ACE appliance validates the action list and parameter map configurations before deploying them.

  See:

  - Configuring Application Acceleration and Optimization, page 4-37
  - Configuring Optimization Parameter Maps, page 7-10.
- Global application acceleration and optimization options—The acceleration and optimization options allow you to apply specific acceleration and optimization features for logging and debugging on a global level on the ACE appliance. See Configuring Global Application Acceleration and Optimization, page 12-16.

Related Topics

- Configuring Traffic Policies for HTTP Optimization, page 12-13
- Optimization Overview, page 12-1

Configuring Action Lists for Application Acceleration and Optimization

An action list groups a series of individual application acceleration and optimization functions that are applicable to a specific type of operation.

Action lists are available for ACE 2.0 modules and ACE appliances. For information on configuring action lists for these devices, see:

- Configuring Action Lists for ACE 2.0 Modules and ACE 4710 A3(1.0) Appliances, page 12-3
- Configuring Action Lists for ACE Appliances, page 12-6

Configuring Action Lists for ACE 2.0 Modules and ACE 4710 A3(1.0) Appliances

Use this procedure to configure an action list for ACE 2.0 module and ACE 4710 A3(1.0) application acceleration and optimization.

Procedure

Step 1 Select **Config > Devices >** ***context*** **> Expert > HTTP Header Modify Action List**. The Action List table appears.

Step 2 Click **Add** to add an action list, or select an existing action list, then click **Edit** to modify it. The HTTP Header Modify Action List configuration screen appears.

Step 3 In the Action List Name field, enter a unique name for the action list. Valid entries are unquoted text strings with a maximum of 64 alphanumeric characters.

Step 4 Click:

- **Deploy Now** to deploy the action list. If this is a new action list, the screen refreshes with additional tables. Continue with Step 5
- **Cancel** to exit this procedure without saving your entry and to return to the Action List table.
- **Next** to deploy your entries and to add another action list.

Step 5 Click the Header Action tab. The Header Action table appears.

Step 6 Click **Add** to add an entry, or select an existing entry, then click **Edit** to modify it.

Step 7 Configure the header action using the information in Table 12-1.

Table 12-1 HTTP Header Modify Action List

Field	Option	Description
Header Name		Enter the name of an HTTP header. Valid entries are unquoted text strings with a maximum of 255 alphanumeric characters.
Operator	Insert	The ACE is to insert a header value in an HTTP client request, server response, or both. In the Header Value field, enter the value that you want to insert into the identified HTTP header. Valid entries are unquoted text strings with no spaces and a maximum of 255 alphanumeric characters. You can also use the following dynamic replacement strings: • %is—Insert the source IP address in the HTTP header. • %id—Insert the destination IP address in the HTTP header. • %ps—Insert the source port in the HTTP header. • %pd—Insert the destination port in the HTTP header.
	Delete	The ACE is to delete the specified HTTP header from a client request, server response, or both.
	Rewrite	The ACE is to rewrite the specified HTTP header in client request packets, server response packets, or both. 1. In the Header Value field, enter the value of the HTTP header that you want to replace. Valid entries are text strings containing 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching data strings. Use parenthesized expressions for dynamic replacement using %1 and %2 in the replacement pattern. **Note** When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([]) to match these symbols (for example, enter `www[.]xyz[.]com` instead of **www.xyz.com**). You can also use a backslash (\) to escape a dot (.) or a question mark (?). 2. In the Replace field, enter the pattern string that you want to substitute for the header value regular expression. For dynamic replacement of the first and second parenthesized expressions from the header value, use %1 and %2, respectively.

Table 12-1 HTTP Header Modify Action List (continued)

Field	Option	Description
Type	Request	The ACE performs the insertion, deletion, or rewrite on the HTTP request packets from clients.
	Response	The ACE performs the insertion, deletion, or rewrite on the HTTP response packets from servers.
	Both	The ACE performs the insertion, deletion, or rewrite on both the HTTP request packets and the response packets.

Step 8 Click:

- **Deploy Now** to deploy this configuration.
- **Cancel** to exit this procedure without saving your entry and to return to the Header Action table.
- **Next** to deploy your entries and to add another header action.

Step 9 Click the SSL Action tab. The SSL Action table appears.

Step 10 Click **Add** to add a new action, or select an existing entry, then click **Edit** to modify it. The SSL Action configuration screen appears.

Step 11 In the URL Expression field, enter the URL regular expression string that is to be matched in the Location header. If the URL in the Location header matches the string that you specify, the ACE rewrites the URL from http:// to https:// and rewrites the port number if configured.

Valid entries are unquoted text strings with no spaces and a maximum of 255 alphanumeric characters. You can enter a text string with spaces provided that you enclose the entire string in quotation marks ("). The ACE supports the use of regular expressions for matching data strings.

Step 12 In the Port field, enter the SSL port number from which the ACE translates a clear port number before sending the server redirect response to the client. Valid entries are integers from 1 to 65535. The default is 443.

Step 13 In the Clear Port field, enter the clear port number to which the ACE translates the SSL port number before sending a server redirect response to the client. Valid entries are integers from 1 to 65535. The default is 443.

Step 14 Click:

- **Deploy Now** to deploy this configuration.
- **Cancel** to exit this procedure without saving your entry and to return to the SSL Action table.
- **Next** to deploy your entries and to add another SSL action.

Related Topics

- Configuring Action Lists for ACE Appliances, page 12-6
- Configuring Pattern Replacements, page 12-10

Configuring Action Lists for ACE Appliances

Use this procedure to configure an action list for ACE appliance application acceleration and optimization.

Tip You can also configure action lists when configuring a virtual server. For more information, see Configuring Application Acceleration and Optimization, page 4-37.

Procedure

Step 1 Select the item to configure:

- To configure a virtual context, select **Config > Devices >** ***context*** **> Expert > Action List**. The Action List table appears.
- To configure a configuration building block, select **Config > Global > All Building Blocks >** *building_block* **> Expert > Action List**.

Note The options in this procedure appear for ACE 4710-type configuration building blocks only.

Step 2 Click **Add** to add a new action list, or select an existing action list, then click **Edit** to modify it.

Step 3 Configure the action list using the information in Table 12-2.

Table 12-2 Action List Configuration Options

Field	Description
Action List Name	Enter a unique name for the action list. Valid entries are unquoted text strings with a maximum of 64 alphanumeric characters.
Enable Delta	Delta optimization dynamically updates client browser caches directly with content differences, or deltas, resulting in faster page downloads. Select the check box to enable delta optimization for the specified URLs. Clear the check box to disable this feature.
Enable AppScope	AppScope runs on the Management Console of the optional Cisco AVS 3180A Management Station and measures end-to-end application performance. Select the check box to enable AppScope performance monitoring for use with the ACE appliance. Clear the check box to disable this feature.
Fast Redirect	The Fast Redirect feature specifies that the ACE appliance is to intercept 302 responses from the origin server and make a second request on behalf of the client for the redirect URL, fetch it, and send it to the client. This feature applies only to redirects within the same domain. Select the check box to enable the fast redirect feature. Clear the check box to disable this feature.

Table 12-2 Action List Configuration Options (continued)

Field	Description
FlashForward	The FlashForward feature reduces bandwidth usage and accelerates embedded object downloading by combining local object storage with dynamic renaming of embedded objects, thereby enforcing object freshness within the parent HTML page. Specify FlashForward implementation: • N/A—This option is not configured. • FlashForward—FlashForward is to be enabled for the specified URLs and that embedded objects are to be transformed. • FlashForward Object—FlashForward static caching is to be enabled for the objects that the corresponding URLs refer to, such as Cascading Style Sheets (CSS), JPEG, and GIF files.
FlashConnect	The FlashConnect feature reduces bandwidth usage and accelerates the downloading of objects that are embedded within HTML pages. FlashConnect dynamically renames embedded objects by adding a prefix and changing the hostname so that the objects appear to reside on different hosts. FlashConnect then has the browser open a separate connection to the origin server for each object and retrieve the objects in parallel instead of sequentially. **Note** If you enable this feature, you must configure DNS so that all requests for the rewritten object URLs are resolved back to the ACE appliance that rewrote them initially. Use the CLI to configure DNS on the ACE appliance. Specify FlashConnect implementation: • N/A—This option is not configured. • Flash Connect—FlashConnect is to be enabled for the specified URLs. • Flash Connect Object—FlashConnect is to be enabled for corresponding embedded object URLs.
Cache Dynamic	Select this check box to enable Adaptive Dynamic Caching for the specified URLs even if the expiration settings in the response indicate that the content is dynamic. The expiration of cache objects is controlled by the cache expiration settings based on time or server load. Clear this check box to disable this feature.
Cache Forward	Specify cache forwarding implementation: • N/A—This option is not configured. • With Wait—Cache forwarding is enabled with the wait option for the specified URL. If the object has expired but the maximum cache TTL time period has not yet expired, the ACE appliance sends a request to the origin server for the object. A user requesting this page continues to receive content from the cache during this time but must wait for the object to be updated before the request is satisfied. When the fresh object is returned, it is sent to the requesting user and the cache is updated. • Without Wait—Cache forwarding is enabled without the wait option.
Dynamic Etag	This feature enables the acceleration of noncacheable embedded objects, which results in improved application response time. When enabled, this feature eliminates the need for users to download noncacheable objects on each request. Select the check box to indicate that the ACE appliance is to implement just-in-time object acceleration for embedded objects not able to be cached. Clear the check box to disable this feature.

Table 12-2 Action List Configuration Options (continued)

Field	Description
Meta Refresh	The Meta Refresh feature enables the ACE appliance to automatically and transparently convert HTML META tag redirections into more efficient HTTP header-based redirections. When enabled, this feature eliminates the need for unnecessary requests to validate freshness and results in significantly faster page response time. Select the check box to enable the smart URL redirection. Clear the check box to disable this feature.
XSLT Merge	Select the check box to indicate that the ACE appliance is to apply XSL style sheet transformations to an XML source document and return the resulting HTML document to the requestor. The ACE appliance applies other optimizations after the XML is transformed, but before the result is returned to the requestor. Clear the check box to disable this feature.
Image Type	Image optimization controls how the ACE appliance compresses JPEG and PNG images. Image optimization is not applied to small images, such as thumbnails, or when optimization reduces the file size by less than 10 percent. Image optimization is not intended for images with many high-frequency components that do not compress well. Specify how the ACE appliance is to handle image optimization: • N/A—This option is not configured. • Standard—The ACE appliance is to perform standard image optimization and smooth the image, if needed, to reduce noise. • Advanced—The ACE appliance is to override standard settings and control individual optimization options.

Step 4 Click:

- **Deploy Now** to immediately deploy this configuration. The ACE appliance validates the action list configuration, and the URL Map table appears. To add URL mappings, continue with Step 5.
- **OK** to save your entries. This option appears for configuration building blocks. To add URL mappings, continue with Step 5.
- **Cancel** to exit this procedure without saving your entries and to return to the Action List table.
- **Next** to deploy your entries and to configure another action list.

Step 5 Click the URL Map tab. The URL Map table appears.

Step 6 Click **Add** to add a new URL mapping. The URL Map configuration screen appears.

Note You cannot modify an existing URL mapping. Instead, delete the existing mapping, then add a new one.

Step 7 Configure a URL mapping using the information in Table 12-3.

Table 12-3 URL Mapping Configuration Options

Field	Description
URL Scope	Select the portion of the URL that is to be remapped: • All—URLs are to be altered, regardless of their locations. • Content—The content is to be altered, not just that which appears in URLs. • Cookie—The domain section of cookies are to be altered. • Header—URLs only within the Location response-header field are to be altered. • Html—URLs only within the URL attribute of META HTTP-EQUIV tags and within the SRC attribute of the HTML tags BASE, HREF, IMG, LINK, SCRIPT, and STYLE are to be altered.
Replacement Directive	Indicate how the URL is to be altered: • Host—The host portion of the URL that is specified in the Source field is to be replaced with the string specified in the Destination field. • Pattern—The portion of the input stream specified in the Source field is to be replaced with the string specified in the Destination field. • Port—The port of the URL that is specified in the Source field is to be replaced with the port specified in the Destination field. • Protocol—The URL protocol HTTP is to be replaced with HTTPS or that HTTPS is to be replaced with HTTP.
Source	Enter the string or value that is to be replaced: • For Host replacements, enter the host portion that is to be replaced. • For Pattern replacements, enter a regular expression that defines subexpressions within the input stream that are to be replaced. • For Port replacements, enter the port number that is to be replaced. • For Protocol replacement, enter HTTP or HTTPS as the protocol to be replaced.
Destination	Enter the string or value that is to replace the entry in the Source field: • For Host replacements, enter the new host string. Valid entries contain a maximum of 64 alphanumeric characters • For Pattern replacements, enter the pattern that is to replace the existing pattern. Valid entries contain a maximum of 64 alphanumeric characters. For more information on Pattern replacements, see Configuring Pattern Replacements, page 12-10. • For Port replacements, enter the port number that is to replace the existing port number. Valid entries are integers from 0 to 65535. • For Protocol replacements, enter HTTP or HTTPS as the protocol to replace the existing protocol.

Step 8 Click:

- **Deploy Now** to immediately deploy this configuration. This option appears for virtual contexts.
- **OK** to save your entries. This option appears for configuration building blocks.
- **Cancel** to exit this procedure without saving your entries and to return to the URL Map table.
- **Next** to deploy your entries and to add another URL mapping.

Related Topics

- Optimization Traffic Policies and Typical Configuration Flow, page 12-2
- Configuring Optimization Parameter Maps, page 12-12
- Configuring Traffic Policies for HTTP Optimization, page 12-13
- Configuring Global Application Acceleration and Optimization, page 12-16

Configuring Pattern Replacements

When configuring Pattern replacements, use regular expression syntax to define subexpressions within the input stream that are to be altered. Subexpressions are delimited using parentheses (). The numbering of the subexpressions begins with 1 and is the number of the left-parenthesis "(" counting from the left.

For example, the following pattern defines three subexpressions in the input stream:

```
(.*)(fast)(.*)
```

The first subexpression is everything before the word "fast." The second subexpression is the word "fast." The third subexpression is everything after the word "fast."

When configuring replacement directives in action lists (see Configuring Action Lists for Application Acceleration and Optimization, page 12-3, Step 7), the string identified in the Source field is replaced by the string specified in the Destination field. In the Destination field, you can use any of the parameter expander functions listed in Table 12-4, or one or more urlmap_pattern(*number*) variables, which refer to specific subexpressions in the original input stream. For example, $urlmap_pattern(0) matches the entire input stream, $urlmap_pattern(1) matches the first subexpression, $urlmap_pattern(2) matches the second subexpression, and so on.

If the specified subexpression does not exist in the input stream, then the variable evaluates to the empty string.

Note The parameter expander functions listed in Table 12-4 apply only to the context of the HTTP request from the client, not to the data stream being sent as a response to the client. For example, the function $http_query_string() evaluates to the query string of the request URL.

Table 12-4 lists the parameter expander functions that you can use.

Table 12-4 Parameter Expander Functions

Variable	Description
$(*number*)	Expands to the corresponding matching subexpression (by *number*) in the URL pattern. Subexpressions are marked in a URL pattern using parentheses (). The numbering of the subexpressions begins with 1 and is the number of the left-parenthesis "(" counting from the left. You can specify any positive integer for the number. $(0) matches the entire URL. For example, if the URL pattern is ((http://server/.*)/(.*)/)a.jsp, and the URL that matches it is http://server/main/sub/a.jsp?category=shoes&session=99999, then the following are correct: $(0) = http://server/main/sub/a.jsp $(1) = http://server/main/sub/ $(2) = http://server/main $(3) = sub If the specified subexpression does not exist in the URL pattern, then the variable expands to the empty string.
$http_query_string()	Expands to the value of the whole query string in the URL. For example, if the URL is http://myhost/dothis?param1=value1&param2=value2, then the following is correct: $http_query_string() = param1=value1&param2=value2 This function applies to both GET and POST requests.
$http_query_param(*query-param-name*) The obsolete syntax is also supported: $param(*query-param-name*)	Expands to the value of the named query parameter (case-sensitive). For example, if the URL is http://server/main/sub/a.jsp?category=shoes&session=99999, then the following are correct: $http_query_param(category) = shoes $http_query_param(session) = 99999 If the specified parameter does not exist in the query, then the variable expands to the empty string. This function applies to both GET and POST requests.
$http_cookie(*cookie-name*)	Evaluates to the value of the named cookie. For example, $http_cookie(cookiexyz). The cookie name is case-sensitive.
$http_header(*request-header-name*)	Evaluates to the value of the specified HTTP request header. In the case of multivalued headers, it is the single representation as specified in the HTTP specification. For example, $http_header(user-agent). The HTTP header name is not case-sensitive.
$http_method()	Evaluates to the HTTP method used for the request, such as GET or POST.

Table 12-4 Parameter Expander Functions (continued)

Variable	Description
Boolean Functions: $http_query_param_present(*query-param-name*) $http_query_param_notpresent(*query-param-name*) $http_cookie_present(*cookie-name*) $http_cookie_notpresent(*cookie-name*) $http_header_present(*request-header-name*) $http_header_notpresent(*request-header-name*) $http_method_present(*method-name*) $http_method_notpresent(*method-name*)	Evaluates to a Boolean value: True or False, depending on the presence or absence of the element in the request. The elements are a specific query parameter (*query-param-name*), a specific cookie (*cookie-name*), a specific request header (*request-header-name*), or a specific HTTP method (*method-name*). All identifiers are case-sensitive except for the HTTP request header name.
$regex_match(*param1*, *param2*)	Evaluates to a Boolean value: True if the two parameters match and False if they do not match. The two parameters can be any two expressions, including regular expressions, that evaluate to two strings. For example, this function: `$regex_match($http_query_param(URL), .*Store\.asp.*)` compares the query URL with the regular expression string .*Store\.asp.* If the URL matches this regular expression, this function evaluates to True.

Configuring Optimization Parameter Maps

Use this procedure to configure an Optimization parameter map for use with a Layer 3/Layer 4 policy map.

Optimization parameter maps can be configured for ACE appliances and ACE 4710-type configuration building blocks only.

Tip You can also configure optimization parameter maps when configuring a virtual server. For more information, see Configuring Application Acceleration and Optimization, page 4-37.

Procedure

Step 1 Select the item to configure:

- To configure a virtual context, select **Config > Devices >** ***context*** **> Load Balancing > Parameter Maps > Optimization Parameter Map**.
- To configure a configuration building block, select **Config > Global > All Building Blocks >** *building_block* **> Load Balancing > Parameter Maps > Optimization Parameter Map**.

The Optimization Parameter Map table appears.

Step 2 Click **Add** to add a new parameter map, or select an existing parameter map, then click **Edit** to modify it. The Optimization Parameter Map configuration screen appears.

Step 3 In the Parameter Name field, enter a unique name for this parameter map. Valid entries are unquoted text strings with no spaces and a maximum of 64 alphanumeric characters.

Step 4 Configure optimization using the information in Table 7-6.

Step 5 Click:

- **Deploy Now** to immediately deploy this configuration. The ACE validates the parameter map configuration and deploys it. This option appears for virtual contexts.
- **OK** to save your entries. This option appears for configuration building blocks.
- **Cancel** to exit this procedure without saving your entries and to return to the Parameter Map table.
- **Next** to accept your entries and to add another parameter map.

Related Topics

- Optimization Traffic Policies and Typical Configuration Flow, page 12-2
- Configuring Action Lists for Application Acceleration and Optimization, page 12-3
- Configuring Traffic Policies for HTTP Optimization, page 12-13
- Configuring Global Application Acceleration and Optimization, page 12-16

Configuring Traffic Policies for HTTP Optimization

Table 12-5 provides a high-level overview of the steps required to configure HTTP optimization on an ACE appliance.

Note Table 12-5 includes only the significant steps in each task. For detailed information on configuring these items, select the links provided, click **Help** in the ANM GUI, or refer to Configuring Traffic Policies, page 11-1.

Assumption

A virtual IP address has been configured for the context in which you configure HTTP optimization.

Table 12-5 Configuring Traffic Policies for HTTP Optimization

	Task	Procedure
Step 1	Create a Layer 7 class map for server load balancing.	1. Select **Config > Devices >** ***context*** **> Expert > Class Map**. 2. Click **Add** to add a new class map. 3. In the Class Map Type field, select **Layer 7 Server Load Balancing**. 4. In the Match Type field, select the method the ACE appliance is to use to evaluate multiple match statements when multiple match conditions exist in the class map. 5. Click **Deploy Now**. 6. Configure match conditions for this class map. For more information, see: • Configuring Virtual Context Class Maps, page 11-6 • Setting Match Conditions for Layer 7 Server Load Balancing Class Maps, page 11-13
Step 2	Create an HTTP optimization action list to specify the optimization actions that are to be performed.	1. Select **Config > Devices >** ***context*** **> Expert > Action List**. 2. Click **Add** to add a new action list. 3. Configure the action list using the information in Table 12-2. 4. Click **Deploy Now**. 5. In the URL Map table, click **Add** to add a new URL mapping. 6. Configure a URL mapping using the information in Table 12-3. 7. Click **Deploy Now**. For more information, see Configuring Action Lists for Application Acceleration and Optimization, page 12-3.
Step 3	Create a Layer 7 HTTP optimization policy map and associate it with the server load-balancing class map in Step 1 and the action list configured in Step 2.	1. Select **Config > Devices >** ***context*** **> Expert > Policy Map**. 2. Click **Add** to add a new policy map. 3. In the Type field, select **Layer 7 HTTP Optimization**. 4. Click **Deploy Now**. 5. In the Rules table, add the server load-balancing class map created in Step 1. 6. In the Action table, add the action list created in Step 2. For more information, see: • Configuring Virtual Context Policy Maps, page 11-30 • Setting Policy Map Rules and Actions for Layer 7 HTTP Optimization, page 11-53

Table 12-5 Configuring Traffic Policies for HTTP Optimization (continued)

	Task	Procedure
Step 4	Create a Layer 3/Layer 4 class map for server load balancing.	1. Select **Config > Devices >** ***context*** **> Expert > Class Map**. 2. Click **Add** to add a new class map. 3. In the Class Map Type field, select **Layer 3/4 Network Traffic**. 4. In the Match Type field, select the method the ACE appliance is to use to evaluate multiple match statements when multiple match conditions exist in the class map. 5. Click **Deploy Now**. 6. Configure Virtual Address match conditions for this class map. For more information, see: • Configuring Virtual Context Class Maps, page 11-6 • Setting Match Conditions for Layer 3/Layer 4 Network Traffic Class Maps, page 11-9
Step 5	Create a Layer 7 policy map for server load balancing and associate it with the Layer 7 server load-balancing class map from Step 1.	1. Select **Config > Devices >** ***context*** **> Expert > Policy Map**. 2. Click **Add** to add a new policy map. 3. In the Type field, select **Layer 7 Server Load Balancing**. 4. Click **Deploy Now**. 5. Associate the Layer 7 server load-balancing class map configured in Step 1 with this policy map by adding it to the Rule table. For more information, see: • Configuring Virtual Context Policy Maps, page 11-30 • Setting Policy Map Rules and Actions for Layer 7 Server Load-Balancing Traffic, page 11-56
Step 6	Create a Layer 3/Layer 4 network traffic policy map and associate it with the: • Layer 3/Layer 4 server load-balancing class map configured in Step 4 • Layer 7 server load-balancing policy map configured in Step 5 • HTTP optimization policy map configured in Step 3	1. Select **Config > Devices >** ***context*** **> Expert > Policy Map**. 2. Click **Add** to add a new policy map. 3. In the Type field, select **Layer 3/4 Network Traffic**. 4. Click **Deploy Now**. 5. In the Rule table, add the Layer 3/Layer 4 server load-balancing class map configured in Step 4. 6. In the Action table, add the: – Layer 7 server load-balancing policy map created in Step 5 – HTTP optimization policy map created in Step 3 For more information, see: • Configuring Virtual Context Policy Maps, page 11-30 • Setting Policy Map Rules and Actions for Layer 3/Layer 4 Network Traffic, page 11-38

Related Topics

- Optimization Traffic Policies and Typical Configuration Flow, page 12-2
- Configuring Action Lists for Application Acceleration and Optimization, page 12-3
- Optimization Overview, page 12-1

Enabling HTTP Optimization Using Virtual Servers

Use this procedure to configure HTTP optimization using virtual servers.

Procedure

Step 1 Create a virtual server by following the instructions in Configuring Virtual Servers, page 4-2.

Step 2 Configure HTTP optimization by following the instructions in Configuring Application Acceleration and Optimization, page 4-37.

Related Topics

- Configuring Traffic Policies for HTTP Optimization, page 12-13
- Optimization Traffic Policies and Typical Configuration Flow, page 12-2

Configuring Global Application Acceleration and Optimization

Note This functionality is available for Admin contexts only and only on ACE appliances.

ANM allows you to configure global application acceleration and optimization options for logging and debugging as performed by the ACE appliance.

Procedure

Step 1 Select **Config > Virtual Contexts >** ***admin_context*** **> System > Acceleration and Optimization**. The Application Acceleration and Optimization configuration screen appears.

Step 2 In the Debug Level field, enter the maximum level of system log messages to be sent to the syslog server, using the values in Table 3-4. The severity level that you specify indicates that you want syslog messages at that level and the more severe levels. For example, if you enter 3 for Error, syslog displays Error, Critical, Alert, and Emergency messages.

Step 3 Check the AppScope Log check box to indicate that the ACE appliance is to upload optimization statistical log information to the optional AVS 3180A Management station. Clear the check box to indicate that the ACE appliance is not to upload this information.

Step 4 Click **Deploy Now** to immediately deploy this configuration on the ACE appliance.

Related Topics

- Optimization Overview, page 12-1
- Optimization Traffic Policies and Typical Configuration Flow, page 12-2