How to Pass Client Certificate Details to a Back-end Server

https://campus.barracuda.com/doc/19334069/

You can configure the Barracuda Load Balancer ADC to pass information from a client to the back-end server through the Barracuda Load Balancer ADC. Using this feature web servers can access client authentication information like Client Certificate parameters or authenticated username and password.

On the Barracuda Load Balancer ADC, you can add client information to a request by configuring a Request Rewrite. Headers can be inserted into the request, or existing headers can be rewritten or deleted before passing the request to the web server, which can then extract the added information. The Barracuda Load Balancer ADC provides macros you can use to communicate request parameters like client certificate details or authenticated user information through headers.

Configuring Request Rewrite to Pass Client Information to a Web Application

To configure a request rewrite rule, perform the following steps:

1. Go to the **TRAFFIC > Web Translations** page, and in the **HTTP Request Rewrite** section specify values for the following fields:
    1. **Rule Name** – Enter a name for the request rewrite rule.
    2. **Sequence Number** – Set the sequence number for the request rewrite policy. The sequence number determines the order of execution for multiple configured policies from highest (1) to lowest (1500).
    3. **Action** - Select the action. To modify client information sent to the web application, the request rewrite action should be set to **Insert Header** or **Rewrite Header**.
    4. **Header Name** – Enter the relevant Header Name, for example X-Forwarded-For.
    5. **Old Value** – Enter the initial request header to be rewritten if the Action is **Rewrite Header**. An asterisk (*) rewrites all named headers, or specify the value or expression to be rewritten.
    6. **Rewrite Value** – Enter the new value of the header to be rewritten when the **Action** is set to **Insert Header** or **Rewrite Header**. Use the macros listed below to specify parameters from the client. When rewriting a header you can specify one or more fields using the separators such as colon (:), semicolon (;), space ( ) and comma (,). In Rewrite Value, the fields can be defined for example: "Name=abc_cookie; Domain=example.com:Path=/". The rewrite-value supports substring addressing of matches, i.e. the matching substrings can be referenced using $1,$2,...$n. The following macros are supported for rewrite values**:**
        - **$X509_ORGANIZATION**, **$X509_LOCALITY**, **$X509_CN**, **$X509_COUNTRY**, **$X509_OU**,**$X509_STATE**, **$X509_EMAIL**, **$X509_SUBJECT**, **$X509_WHOLE**: Fields in the X509 client certificate when client authentication is **On**.
        - **$SRC_ADDR**: The client IP from which the request originated.

- **$DST_ADDR**: The destination address.
- **$URI**: URI.
- **$AUTH_USER**: Username of the authenticating user.
- **$AUTH_PASSWD**: Password of the authenticating user.
- **$AUTH_GROUPS**: Group associated to the authenticating user.

7. **Rewrite Condition** – Set the condition under which a rewrite should occur. An asterisk (*) indicates there are no conditions (applies to all). Details on the format of the Rewrite Condition are explained below in Rewrite Condition Format.

2. Click **Add** to add the above settings.

Note: When multiple policies are configured, the request continues to be processed by other (higher sequence number) policies. If you wish to stop processing after a particular rule is matched, click **Edit** next to the rule and set **Continue Processing** to *No*.

Rewrite Condition Format

The request **Rewrite Condition** specifies when a rewrite should occur. The **Rewrite Condition** is made up of expressions combining Request Rewrite Tokens and Operations on those tokens. These expressions can then be joined with each other using logical or (or, OR, ||) or logical and (and, AND, &&). Examples of **Rewrite Conditions**: (Header User-Agent co mozilla), (URI rco /abc*html), (Client-IP eq 10.0.0.1)&&(Method eq POST). An asterisk indicates there are no conditions for rewrite, so the rewrite is done in every case.

Request Rewrite Tokens

These tokens can be used in a request **Rewrite Condition**:

- **Header**: The HTTP header in the request. The word **Header** precedes the name of the relevant header or * to indicate all headers. Examples: Header Accept co soap, Header Soap-Action ex.
- **Client-IP**:The IP address of the client sending the request. The IP address can be either a host IP address or a subnet specified by a subnet mask. Only operations EQ and NEQ can be combined with this token. Examples: Client-IP eq 192.168.1.0/24 (subnet qualified by a netmask) Client-IP eq 192.168.1.10 (host IP address)
- **Uri**: The Uniform Resource Identifier of the resource on which to apply the rule. Example: URI rco /abc*html
- **Method**: The HTTP method in the request. Example: Method eq GET
- **Http-Version**: The HTTP protocol version of the request. Example: HTTP-Version eq HTTP/1.1
- **Parameter**: The query part of the URL which is passed to the servers as a name-value pair. In addition, the word "$NONAME_PARAM" can be used when the parameter name is absent. Examples: Parameter sid eq 1234, Parameter $NONAME_PARAM co abcd
- **Pathinfo**: The portion of URL which contains extra information about the path of the resource on the server. Example: pathinfo rco abc*

Operations for Request Rewrite

These operations can be combined with Request Rewrite Tokens in a request **Rewrite Condition**:

- **contains, CONTAINS, co, CO** – Token contains the given value.
- **ncontains, nCONTAINS, nco, nCO** – Token does not contain the given value.
- **rcontains, rCONTAINS, rco, rCO** – Token contains the given value which is interpreted as a regular expression.
- **equals, EQUALS, eq, EQ** – Token equals the given value.
- **nequals, nEQUALS, neq, nEQ** – Token does not equal the given value.
- **requals, rEQUALS, req, rEQ** – Token equals the given value interpreted as a regular expression.
- **exists, EXISTS, ex, EX** – Token exists.
- **nexists, nEXISTS, nex, nEX** – Token does not exist.