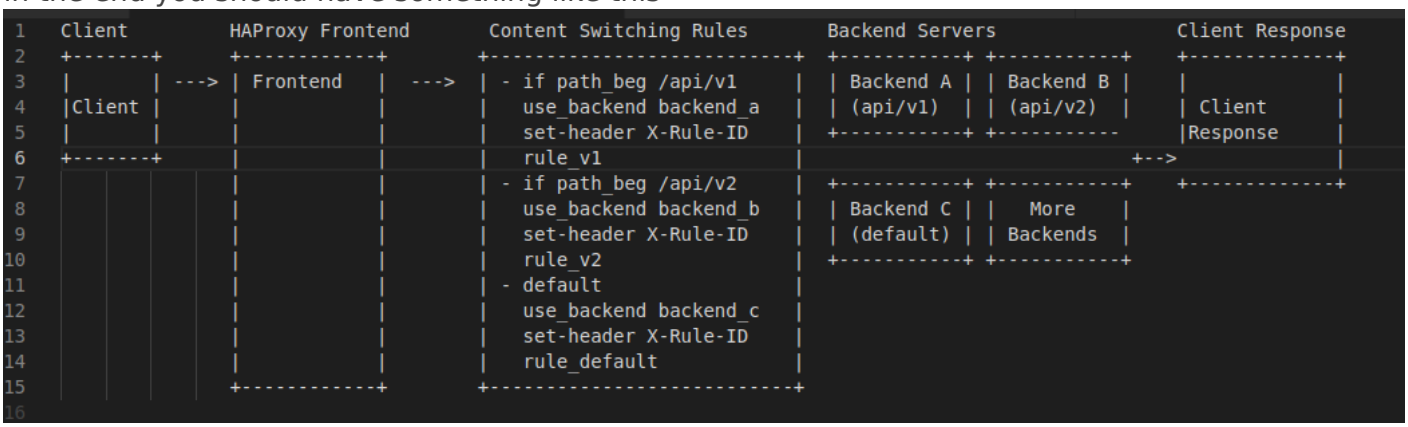
Debug custom rules using http headers with haproxy

Step 1: Define a Custom Header

You can use HAProxy's `http-request set-header` directive to add a custom header to requests passing through HAProxy. This header can contain information like a rule ID or a Lua snippet identifier.

in the end you should have something like this



Step 2: Log the Custom Header

Configure the logging format to include the custom header, so it gets logged.

Example Configuration

Here's an example configuration that demonstrates how to set a custom header and log it.

1. Define the custom header in the frontend or backend section

```
frontend http_in
    bind *:80
    mode http

    # Insert custom header with rule ID
    http-request set-header X-Rule-ID %[unique-id]
```

```
    default_backend servers

backend servers
    mode http

    server server1 192.168.1.100:80 check
```

In this example, the header `X-Rule-ID` is added to each request with a unique ID (you can customize the value as needed).

2. Customize the log format to include the custom header

```
global
    log 127.0.0.1 local0

defaults
    log global
    option httplog
    log-format "%ci:%cp [%t] %ft %b/%s %TR/%Tw/%Tc/%Tr/%Tt %ST %B %CC %CS %tsc
%ac/%fc/%bc/%sc/%rc %sq/%bq %hr %hs %{+Q}r %ST %B %H %{+Q}r %hr %hs %{+Q}r %ht(X-Rule-ID)"

frontend http_in
    bind *:80
    mode http

    # Insert custom header with rule ID
    http-request set-header X-Rule-ID %[unique-id]

    default_backend servers

backend servers
    mode http

    server server1 192.168.1.100:80 check
```

In the log format, `%ht(X-Rule-ID)` is used to log the value of the `X-Rule-ID` header.

Using Lua to Set Headers

If you need more complex logic for setting the header, you can use a Lua script:

1. Enable Lua support in HAProxy

Ensure your HAProxy is built with Lua support. You can check this by running `haproxy -vv` and looking for `Built with Lua`.

2. Create a Lua script (`/etc/haproxy/lua/add_header.lua`)

```
core.register_action("set_rule_id", { "http-req" }, function(txn)
    txn.http:req_set_header("X-Rule-ID", "your_rule_id_or_identifier")
end)
```

3. Update the HAProxy configuration to use the Lua script

```
global
    lua-load /etc/haproxy/lua/add_header.lua
    log 127.0.0.1 local0

defaults
    log global
    option httplog
    log-format "%ci:%cp [%t] %ft %b/%s %TR/%Tw/%Tc/%Tr/%Tt %ST %B %CC %CS %tsc
%ac/%fc/%bc/%sc/%rc %sq/%bq %hr %hs %{+Q}r %ST %B %H %{+Q}r %hr %hs %{+Q}r %ht(X-Rule-ID)"

frontend http_in
    bind *:80
    mode http

    # Call Lua script to set the header
    http-request lua.set_rule_id

    default_backend servers

backend servers
    mode http

    server server1 192.168.1.100:80 check
```

In this configuration:

- The Lua script sets the `X-Rule-ID` header.
- The log format includes the custom header.

Sample access.log output

There are rule identifiers at the end of every line like rule_v2 or rule_v1

```
192.168.0.1:54321 [11/Jun/2024:15:23:45.123] http_in backend_a/server1 0/0/2/3/5 200 1234 - - ---- 2/1/0/0/0 0/0 "GET /api/v1/resource HTTP/1.1" 200 1234 - - ---- %{+Q}r %hr %hs %{+Q}r rule_v1
192.168.0.2:54322 [11/Jun/2024:15:23:46.456] http_in backend_b/server2 0/0/3/4/7 200 2345 - - ---- 2/1/0/0/0 0/0 "GET /api/v2/resource HTTP/1.1" 200 2345 - - ---- %{+Q}r %hr %hs %{+Q}r rule_v2
192.168.0.3:54323 [11/Jun/2024:15:23:47.789] http_in backend_c/server3 0/0/1/2/3 404 345 - - ---- 2/1/0/0/0 0/0 "GET /other/resource HTTP/1.1" 404 345 - - ---- %{+Q}r %hr %hs %{+Q}r rule_default
```

Summary

By configuring HAProxy to set custom headers and customizing the log format to include these headers, you can effectively log custom identifiers, rule IDs, or other information that may be critical for your debugging and monitoring purposes. This setup provides a powerful way to gain insights into how requests are being processed and routed through your HAProxy instance.

Revision #3
Created 2024-06-12 04:50:54 UTC by Mesut Bayrak
Updated 2024-06-12 05:15:27 UTC by Mesut Bayrak