v10 WebAccelerator and iRules

Dawn Parzych, 2009-14-05

A customer recently contacted me regarding an iRule that worked in v 9.4 but did not produce the expected results in v 10. A little research revealed that the new plug-in architecture is influencing when the iRule is firing.

The iRule

The iRule is logging the value of the X-PvInfo header or indicating the header is not present.

```
when HTTP_RESPONSE {
  if {[HTTP::header exists X-PvInfo]}{
   log local0. "X-PvInfo:[HTTP::header "X-PvInfo"]"
  } else {
   log local0. "did not find X-PvInfo header"
  }
}
```

The Problem

WebAccelerator inserts the X-PvInfo header into each and every response that it processes, there should be no instances where the header is missing however in v10 the Local Traffic logs on the Big-IP show "Rule LogHeaders : did not find X-PvInfo header." Looking at the headers received by the browser I can clearly see that the header does exist.

```
(Status-Line)      HTTP/1.1 200 OK
Accept-Ranges  none
Authorization     123
Cache-Control   must-revalidate, no-cache, no-store, post-check=0, pre-check=0
Connection         Keep-Alive
Content-Encoding          gzip
Content-Length               1256
Content-Type    text/html; charset=UTF-8
Date      Fri, 01 May 2009 12:25:16 GMT
ETag       "pvfe8a15580ad513c1f56c687ef64d6bab"
Vary       Accept-Encoding
X-PvInfo             [S10202.C5877.A15567.RA0.G0.UB5831B78].[OT/html.OG/pages]
```

The Issue

With the new plug-in architecture the HTTP_RESPONSE event is triggered after the response from the server before the plug-in and WebAccelerator processes the response. The response from the server does not contain the X-PvInfo header that is inserted after WebAccelerator has processed the server response.   Prior to the plug-in architecture the HTTP_RESPONSE event fired after WebAccelerator processed the response from the server. The difference in when the iRule fires results in the log message that the header doesn't exist.

The Solution

To have an HTTP_RESPONSE event fire after WebAccelerator has processed and manipulated the response you need to define a custom virtual server configuration known as VIP targeting VIP.   VIP-to-VIP configures a front virtual server which contains the WebAccelerator post-processing iRule and the back virtual server has the WebAccelerator enabled class associated with it. Solution 9388 on AskF5 illustrates how to do this using ASM. The same applies to WebAccelerator.

The trick with VIP targeting VIP is the front server does not have a pool assigned to it the pool is selected from within the iRule. Our iRule above would need to be changed to:

```
when RULE_INIT {
# Define a global variable which references the WA-enabled VIP name
  set ::targetvip "back_vip"
}
when HTTP_REQUEST {
  virtual $::targetvip
}

when HTTP_RESPONSE {
  if {[HTTP::header exists X-PvInfo]}{
   log local0. "X-PvInfo:[HTTP::header "X-PvInfo"]"
  } else {
   log local0. "did not find X-PvInfo header"
  }
}
```

The front virtual server should have a configuration that looks something like:

```
virtual front_vip {
   snat automap
   destination 172.16.1.100:http
   ip protocol tcp
   rules post_wa_irule
   profiles
      http
      tcp
}
```

The back virtual server should have a configuration that looks something like:

```
virtual back_vip {
   snat automap
   pool testpool
   destination 172.16.1.101:http
   ip protocol tcp
   httpclass WebAccelerator
   profiles
      http-acceleration
      tcp
}
```