Alteon
34.x

Alteon Level 1
Lab Manual
SLB Persistence

Table of Contents

Objectives

After viewing the “Persistence with Insert Cookie” training module and completing this lab, you should be able to:

- Configure IP persistence by using Hash or Persistent Hash.
- Configure L7 cookie persistence using passive, rewrite or insert mode.

Overview

When configuring a load-balanced service, it is important to maintain information consistency across multiple requests in a user’s session. Sending all requests in a session to the same backend server is known as *persistence* or *stickiness*.

Persistent server load balancing uses Layer 3 and Layer 7 persistency. Layer 3 handles only IP addresses, Hash or Persistent Hash are used as the metric.

Layer 7 persistency depends on the application. We use HTTP with passive cookies, cookie rewrite, and cookie insert to provide persistence.

Lab Preparation: Restore SLB Setup

Before you begin this lab:

a. You should have successfully completed SLB configuration.
b. Access Alteon management port and login. Import SLB configuration or make sure it is already in the Alteon configuration.
c. Verify your SLB configuration is properly working.
d. At Cyber Controller is config Alteon-A_SLB-Configuration.tgz available. You sync this to Alteon-B in case of HA.

Lab Activities

Here is a summary of what you will be performing in this Lab:

1. Configure Persistent SLB
   - Set group metric to Persistent Hash
   - Change metric to Round Robin
   - Select rport 8080
   - Change server cookie
   - Enable passive cookie
   - Revert rport back
   - Configure to insert session cookie
2. Validate your Configuration

Persistent SLB by SLB Metric

1. Set the group metric to PHASH.
   To verify this configuration change generate test-traffic and verify all persistent connections to a single server even after session table expires (60 seconds).

Configuration → Application Delivery → Server Resources → Server Groups

a) Double-click Group1 to edit
b) **Group Settings** tab
   SLB Metric: Persistent Hash
c) **Apply**, **Save**, **Sync**
d) **Monitoring → Application Delivery → Server Resources → Real Servers**
e) Access VIP on HTTP by browser, preferred Chrome, open and close a couple of times.
f) Press refresh button ⟳ watch Total Session count changes. Only one WebServer get requests.

2. Change group metric to *Round Robin*.

a) **Configuration → Application Delivery → Server Resources → Server Groups**
b) Edit Group1
c) **Group Settings** tab:
   SLB Metric = Round Robin

Cookie Based Persistency

Passive Cookie Based Persistence

Currently, the web application, our web shop HACKAZON, which sets cookies is on real port 8080 available.

3. Change the virtual server rport value to 8080.
NOTE: You will notice a different service than previously. Do not be alarmed.

a) **Configuration → Application Delivery → Virtual Services**
b) Edit Virt1
c) Edit virtual service HTTP
d) **Properties** tab set **Real Server Port**: 8080

4. Clear the session table and clear the SLB statistics.

CLI:

```
/oper/slb/clear
/stats/slb/clear
Confirm clearing all SLB statistics [y/n]: y
```

5. Enable passive cookie-based persistence on the virtual server service.
   Enable the VIRT to track passive cookies.
   The server set a cookie called PHPSESSID which we are going to track for persistency
   Example of a server cookie: “Set-Cookie: PHPSESSID=a3tfarqt7nf1dksccuq8t2p961; path=/”

a) **Configuration → Application Delivery → Virtual Services**
b) Edit Virt1
c) Edit virtual service HTTP/Port 80 → **Persistency**

Persistency tab Settings	
Persistency Mode	Cookie
Cookie Mode	Passive
Cookie Type	Header
Cookie Name	PHPSESSID
Value Length	16
Value Offset	1

d) **Submit, Apply** and **Save**

Test the configuration.

a. First disable all cookies in the browser. In Chrome settings search for cookies and set ‘Block all cookies’. Since the shop have same content on both web server you can’t detect load balancing by content. Watch the statistics **/st/slb/vi Virt1 80.** Refresh browser by Ctrl+F5 and watch stats again. Total sessions, Throughput and Cookie Misses increase.

b. Then enable cookies, set Chrome to Allow all Cookies, and notice sessions go to the same server.

Reset statistics between the tests.

CLI:

```
/stats/slb/virt Virt1 80
```

Specially view the cookie related information at the end of the output:

```
HTTP header loadbalance stats:
'Cookie' Hits:                      59
'Cookie' Misses:                     0
Cookie + URL Matches:                     0
Cookie - URL        :                     0
Total 'Cookie' Sessions:            59
------------------------------------------------------------------
Persistent Cookie SLB maintenance stats:
                          Current  Total      Highest
                          Sessions Sessions   Sessions
------------------------- -------- ---------- --------
Unique Sessions                  8          0        0
Cookie Sessions                  6         59        6
Persistent Port Sessions         0          0        0
```

Close all browsers and time out of the session table (or clear the session table). Notice the connection to a new server (unlike phash).

6. Revert virtual server rport back to port 80 (server has no cookie).

a) **Configuration → Application Delivery → Virtual Services**
b) Edit Virt1
c) Edit virtual service HTTP
d) **Properties** tab
Real server Port: 80

Insert Cookie Based Persistence

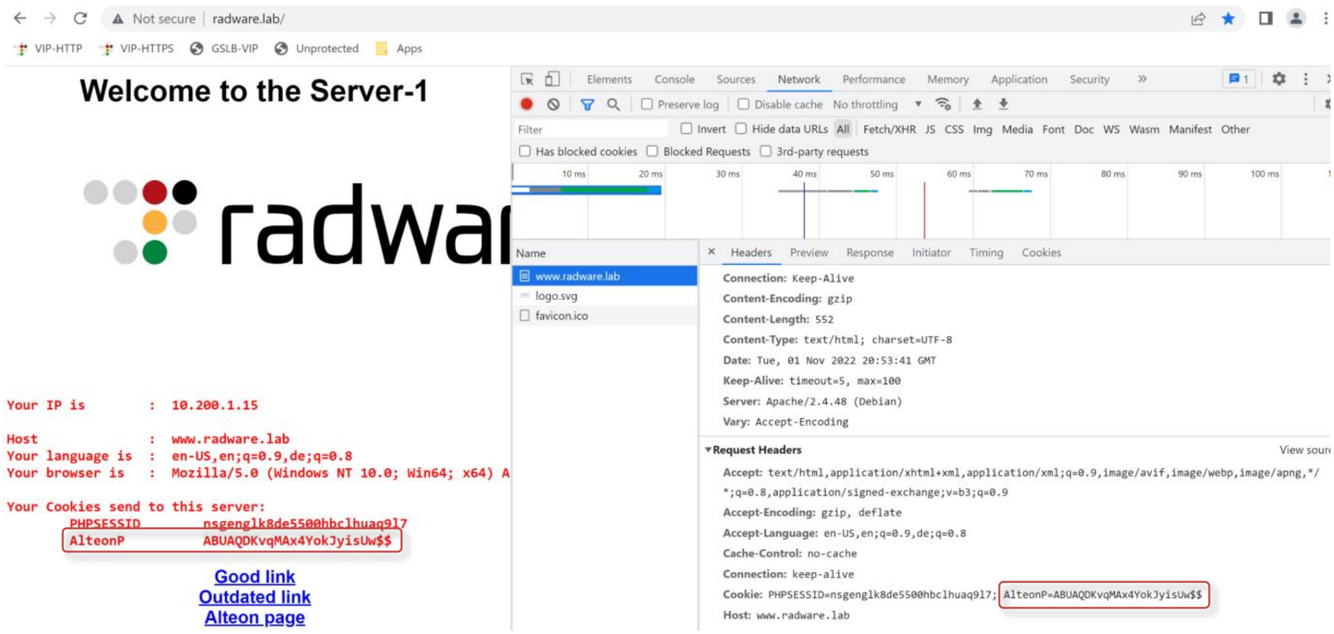
1. Configure virtual server to insert a session cookie.
Optionally you can insert a time duration cookie (expires by date or expires by time).

a) **Configuration → Application Delivery → Virtual Services**
b) Edit Virt1
c) Edit virtual service HTTP
d) **Persistency** tab:

Persistency tab Settings	
Persistency Mode	Cookie
Cookie Mode	Insert
Cookie Name	AlteonP (Note: In production, use a different cookie name for security purposes)
Domain Name	Exclude
Secure	Disable

Validate your Configuration

1. Inspect the cookie inserted.
   a. Connect to VIP using Chrome with Web Developer – Network (Ctrl+Shift+i).
   b. Click reload.
   c. Notice “AlteonP” session cookie and PHPSESSID from passive setup.



Welcome to the Server-1

```
Your IP is         :  10.200.1.15

Host               :  www.radware.lab
Your language is   :  en-US,en;q=0.9,de;q=0.8
Your browser is    :  Mozilla/5.0 (Windows NT 10.0; Win64; x64) A

Your Cookies send to this server:
        PHPSESSID          nsgenglk8de5500hbclhuaq9l7
        AlteonP            ABUAQDKvqMAx4YokJyisUw$$
```

Good link
Outdated link
Alteon page

```
Connection: Keep-Alive
Content-Encoding: gzip
Content-Length: 552
Content-Type: text/html; charset=UTF-8
Date: Tue, 01 Nov 2022 20:53:41 GMT
Keep-Alive: timeout=5, max=100
Server: Apache/2.4.48 (Debian)
Vary: Accept-Encoding
▾Request Headers
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/
*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,de;q=0.8
Cache-Control: no-cache
Connection: keep-alive
Cookie: PHPSESSID=nsgenglk8de5500hbclhuaq9l7; AlteonP=ABUAQDKvqMAx4YokJyisUw$$
Host: www.radware.lab
```

Notice the cookie inserted by the Alteon under the cookie session. Cookie rewrite/insert generates a persistency entry (p-entry) in the session table.

2. Compare the inserted cookie (as previous) with Alteon cookie table.

CLI:

```
/i/slb/sess/dip 192.168.175.50

Printing Sessions for SP 1
1,00: 0x358de7a82cc125c1c0a8af3200006800, 192.168.175.50 10.200.1.100 g:Group1  age 10 PS c:1
1,01: 192.168.175.20 59398, 192.168.175.50 http ->
      10.200.1.15 2136 10.200.1.100 http tcp age 8 v:11   E



/oper/slb/b64encod 0x358de7a82cc125c1c0a8af3200006800

==>  AGgAADKvqMDBJcEsqOeNNQ$$
```

3. Export configuration as a backup. Name it PERSISTENT SLB LAB.

4. Revert virtual server rport back to port 80 and remove any cookie persistency. Next labs do require standard SLB setup.

For questions, contact **training@Radware.com**