It's Midnight. Do You Know Where Your Tech Support Is?

The Bhattman, 2010-24-09

By Bhattman @ gmail.com

Here is a familiar story…you are sleeping soundly. It's actually been weeks since you have slept this good. In fact you are sleeping so good that you are having one of your favorite dreams where you are flying at your own control. Then suddenly you are stirred awake, by a phone call. As you struggle to keep yourself from opening your eyes. You stumble around the night stand to pick up the phone. You think that you can salvage this night by not opening your eyes. This better be an emergency. Then the familiar voice or sound comes in. *"Hi my name is Jai and I work in the <blank> helpdesk…there is an application that appears to be down and the application owner wants to know if the POOL is UP or DOWN".* As you wrestle the feelings of sleep and anger, you respond before the latter takes over *"All the application owners can log in the website using a read-only account and look at the pool status."* You are suddenly relieved at your answer because that solution works. As you prepare your mind to drift back into sleep a response stirs you awake again "*Sir, yes but apparently the App owner is having problems entering the username and password he was given.".* With one last breath of sleep sanity you ask the Helpdesk to log in and help determine the status. Alas that is when it hits you - the helpdesk have zero connectivity and training to the F5 - just the monitoring system. You open your eyes.

Sound familiar? It should. This type of sleep deprivation is a common infliction to F5 administrators. They are faced with these kinds of situations throughout the day and night. It's no longer relevant to provide usernames and passwords to access the F5 system because it's complicated for App owners who do not use this system on a daily basis. Also the helpdesk probably doesn't access the system because contractually they are only allowed to access the central monitor – which doesn't provide POOL status information.

So what do you do?

You could get the central monitor to provide that status via SNMP and specific MIB OIDs. However, that too would be a daunting task because Pools are created and destroyed every day.

So now what?

You can write up an iRule to do this for you. How? We know that **HTTP::response** can provide a HTML response. We also know that **active_members** can tell us whether pool members are up or down

```
1: when HTTP_REQUEST
2:   if {[HTTP::uri] eq "/status" }
3:     scan [LB::select] %s%s%s%s%d command current_pool command2 current_member current_port
4:     eval [LB::select]
5:     set response {<html><head><title>$current_pool Pool Status - [clock format [clock seconds]]</title>
                      <meta http-equiv='refresh' content='10; url=http://[HTTP::host]/status'></head>}
6:     if { [active_members $current_pool] < 1 } {
7:       append response {POOL NAME:$current_pool<br/> CURRENT MEMBER:$current_member:$current_port<br/> STATUS: DOWN <br/></body>
8:     } else {
9:       append response {POOL NAME:$current_pool<br/> CURRENT MEMBER:$current_member:$current_port<br/> STATUS: UP <br/></body></
10:    }
11:  }
12:  HTTP::respond 200 content $response "Content-Type" "text/html"
13: }
```

This code works well, but it's very limiting because if a pools contain more then 2 members it will only provide a status of the active member the ADC has chosen and not report on the other pool member. Therefore, this iRule code can ONLY provide you UP/DOWN at the pool level.

So we need to take it up a notch. We know that LB::status can provide status of a node address or pool member specifically providing UP/DOW, Session enabled or Disabled. We also know it makes more sense to provide this status in one location so the App owner doesn't have to enter anything more then http://www.f5status.com or whichever URL that can be used. The following example uses **/status** as the switch in the URL to run the iRule hosted on a Virtual address.

```
1: when HTTP_REQUEST {
2:   if { [HTTP::uri] eq "/status" } {
3:     set response "<html><head><title>BIGIP Pool Member Status - \
4:     [clock format [clock seconds]]</title><meta http-equiv='refresh' content='300;\
5:     url=http://[HTTP::host]/status'></head><h1>BIGIP Pool Member Status - [clock format [clock seconds]]</h1>\
6:     <table border='1'><tr><th>Status</th><th>Pool Name</th><th>Member</th><th>Port</th></tr>"
7:     foreach { selectedpool } [class get pool_member_status_list] {
8:       if { [catch {
9:         scan $selectedpool {%[^/]/%[^:]:%s} poolname addr port
10:         switch -glob [LB::status pool $poolname member $addr $port] {
11:           "up" {
12:             append response "<tr><td><font style=\"color:green\"><b>UP</b></font></td>\
13:             <td>[string tolower $poolname]</td><td>$addr</td><td>$port</td><tr>"
14:           }
15:           "down" {
16:             append response "<tr><td><font style=\"color:red\"><b>DOWN</b></font></td>\
17:             <td>[string tolower $poolname]</<d><td>$addr</td><td>$port</td><tr>"
18:           }
19:           "session_enabled" {
20:             append response "<tr><td><font style=\"color:blue\"><b>ENABLED</b></font></td>\
21:             <td>[string tolower $poolname]</td><td>$addr</td><td>$port</td><tr>"
22:           }
23:           "session_disabled" {
24:             append response "<tr><td><font style=\"color:black\"><b>DISABLED</b></font></td>\
25:             <td>[string tolower $poolname]</td><td>$addr</td><td>$port</td><tr>"
26:           }
27:           Default {
28:             append response "<tr><td><font style=\"color:orange\"><b>INVALID</b></font></td>\
29:             <td>[string tolower $poolname]</td><td>$addr</td><td>$port</td><tr>"
30:           }
```

```
31:         }
32:         #SWITCH END
33:       } errmsg] } {
34:         append response "<tr><td><font style=\"color:orange\"><b>INVALID</b></font></td>\
35:           <td>[string tolower $poolname]</td><td>$addr</td><td>$port</td><tr>"
36:         }
37:         #SECOND CONDITIONAL STATEMENT END - Catch end error in an invalid named or non-existent pool
38:     }
39:     #FOR LOOP END
40:     append response "</table></body></html>"
41:     HTTP::respond 200 content $response "Content-Type" "text/html"
42:   }
```

How does this work? Well it's quite simple once you break down the logic. Lines 3 to 6 create a variable called "response" which will be the first part of the HTML page that will be sent back to the client requesting a status. Line 7 starts the for loop which will loop through each item within the datagroup called "pool_member_status_list". The datagroup is external datagroup which contains the list of all current pools, their members and port number. Within the loop it will use LB::status to determine each members status and after the loop is finished will generate a HTML response from variable "response" which started with a HTML header and additional HTML code appended over the course of the FOR Loop. What does it look like?

BIGIP Member Status - Mon Sep 20 06:53:08 PDT 2010

Status	Pool Name	Member	Port
DOWN	pool_something	192.168.1.100	80
UP	pool_something	192.168.1.101	80
DISABLED	pool_blah	192.168.1.102	80

The beauty here is that not only can the App owners can view the pool status at the member level, but they can view it without authentication and on their mobile phones (if they can reach the website at all).

What does the pool member list contain?

The following is an example of what the entries would look like:

```
"pool_something/192.168.1.100:80",
"pool_something/192.168.1.101:80",
"pool_blah/192.168.1.102:80",
```

Now you could do this manually but what fun would it be if we can't somehow automate this. How?

This where the beauty of Linux Cron Scripts and bigpipe comes handy. From the Bigpipe command you know you could create the entries in the following shell command:

```
b pool all |grep "POOL MEMBER" | awk '{sub(":any",":0",$4);print "\""$4"\","}' | sort
```

Now let's assume with trial and error you came up with the following shell cron script:

```
#!/bin/bash
b db bigpipe.displayservicenames false
b pool all |grep "POOL MEMBER" | awk '{sub(":any",":0",$4);print "\""$4"\","}' | sort >/var/class/pool_member_status_list.class
if [ "`cat /var/prompt/ps1`" == "Active" ]; then
   b load
fi

exit 0
```

How does this work?

The second line is making sure that when you use "`b pool all`" it doesn't output service names but actually port numbers. The third line outputs the sorted pool member list into a file in a specific location in a specific format. Lines 4 and 6 is a conditional statement that checks to see it can run the "`b load`" command on an active ADC unit within a HA pair. Why? Datagroups are compiled and loaded into memory the minute they are defined and updated via the F5 GUI. In this case "pool_member_status_list.class" is dynamic in which the file is only updated. So the datagroup that is pointing to the "pool_member_status_list.class" needs to recompile with the new information. This is where "`b load`" comes i

Of course you need to save this shell script and be able to execute this without your interaction. You save this script under **/etc/cron.daily** directory which is configured to run any script within this directory daily around midnight.

You also need to make it executable so the F5 OS can run the script. Assuming you called the cron script "pool_member_status_list.cron" you issue one of two commands as in the example below:

```
chmod +755 /etc/cron.daily/pool_member_status_list.cron
or
chmod +x /etc/cron.daily/pool_member_status_list.cron
```

Now that is settled how do we tie the datagroup to the file?. Here is an this example instruction of how to do it:

Main >> Local Traffic >> iRules
Select Data Group List
Click create
enter name: pool_member_status_list
enter Type: External
Path / Filename: /var/class/pool_member_status_list.class
File Content: String

So there you have it. An pool status page w/o authentication AND the benefit here is that you can view through your mobile device.

So now you can rest easy knowing that you can sleep easier at night knowing that your tech support is the iRule itself.

The code has been posted on the codeshare section with several enhancement done by the F5 community.

http://devcentral.f5.com/wiki/default.aspx/iRules/Pool_Member_Status_Page_on_a_Virtual_Server.html

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com