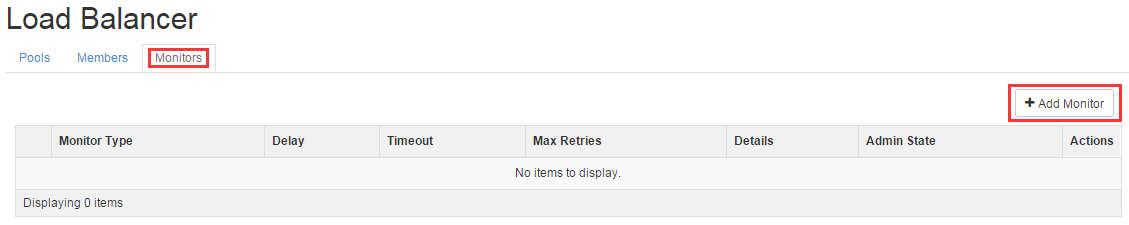


前面我们创建了 Pool，VIP 并添加了 Member。今天将创建 Monitor，然后测试 LBaaS 是否能够正常工作。

**创建 Monitor**

LBaaS 可以创建 monitor，用于监控 Pool Member 健康状态。   
如果某个 member 不能正常工作，monitor 会将其状态设置为 down，从而避免将后续请求转发给它。

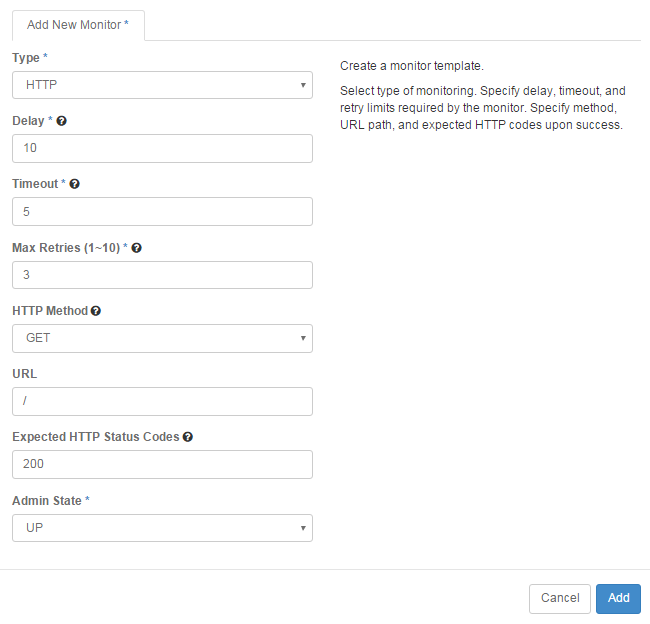
下面我们为 Pool 添加一个 monitor。   
在 Monitors 标签页中点击 “Add Monitor” 按钮



Type 选择 “HTTP”，含义是通过 HTTP 检查 member 的健康状态。   
Delay 设置为 “10”，含义是 10 秒检查一次 member 的状态。   
Timeout 设置为 “5”，含义是如果 member 在 5 秒内无法应答，则超时。   
Max Reties 设置为 “3”，含义是如果尝试 3 次都超时或者失败，则将 member 状态设置为 down。

HTTP Method 设置为 “GET”   
URL 设置为 “/”   
Expected HTTP Status Codes 设置为 “200”

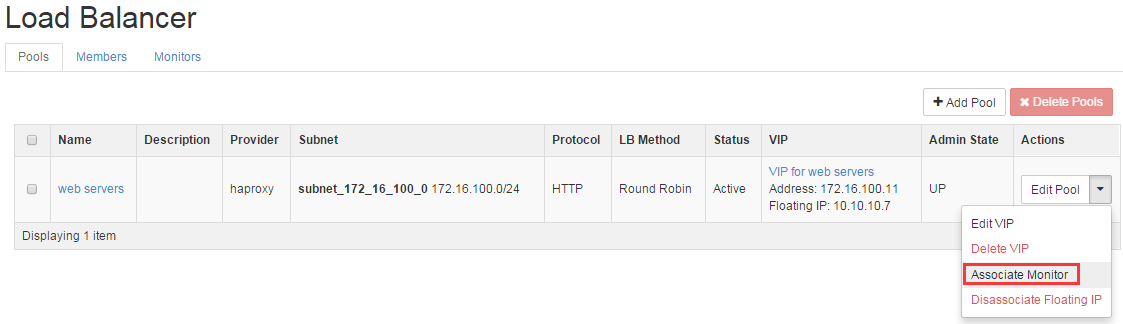
上面三项的含义是通过 HTTP GET 请求 member “/” URL，如果返回码为 200，则认为 member 状态正常。



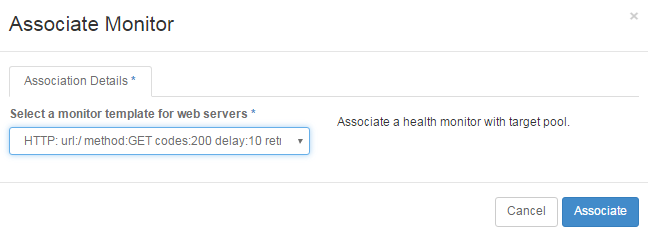
点击 “Add”，monitor 创建成功。



下面将新建的 monitor 添加到 pool 。   
在 “web servers” 的操作列表中点击 “Associate Monitor”



选择我们刚刚创建的 monitor。

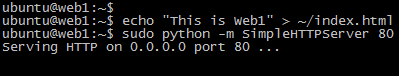


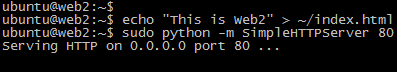
点击 “Associate”。

**测试 LBaaS**

经过上面的设置，我们创建了包含 member “Web1” 和 “Web2” 的 Pool “web servers”，并添加了 monitor。   
准备就绪，可以测试 load balancer 是否正常工作了。

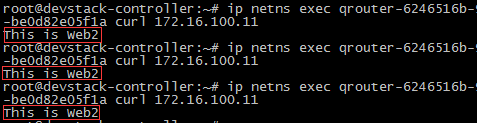
首先在 Web1 和 Web2 中启动 HTTP 服务，在 80 端口监听





这里我们使用 python 提供的 SimpleHTTPServer 模块启动了 HTTP 服务。   
web server 的 index.html 显示当前访问的是哪个 member。

在 router 的 namespace 上多次执行 curl 172.16.100.11（VIP）



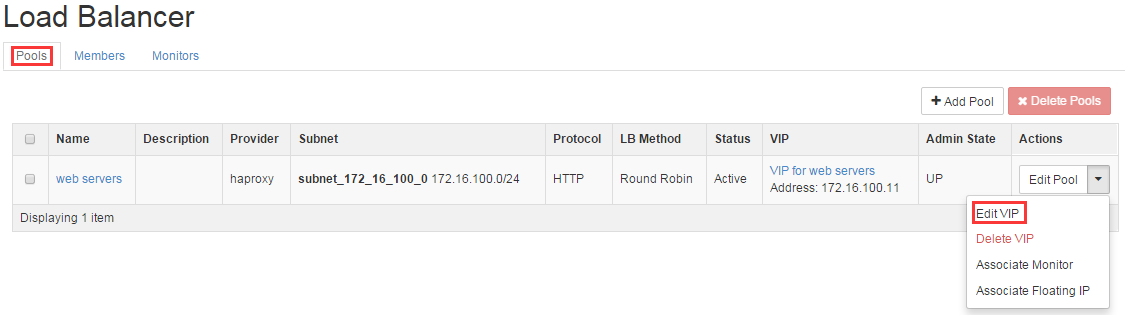
测试结果显示每次访问的都是 Web2 这个 member。   
**为什么没有访问到 Web1 呢？**

还记得我们前面讨论的内容吗：   
Load Balance Method -- ROUND\_ROUBIN   
Session Persistence -- SOURCE\_IP

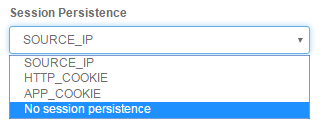
在这种配置下，第一个 curl 请求 HAProxy 通过 ROUND\_ROUBIN 选择了 Web2。  
而后续的请求，HAProxy 则会应用 SOURCE\_IP 机制，仍然选择 Web2。

下面我们修改一下配置。

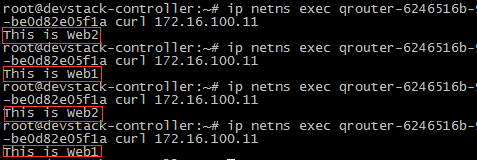
在 “web servers” 的操作列表中点击 “Edit VIP”。



选择 “No session persistence” 并保存。



再进行 curl 测试。



可以看到已经在 “Web1” 和 “Web2” 之间 round robin 了。

下一节我们将分析 LBaaS 的内部实现和工作机制。