How to Configure Apache Load Balancer

Apache load balancer helps your websites handle large traffic loads without any performance issues. Here's how to configure Apache Load balancer for your website.

Apache Load Balancer Configuration

Here are the steps to configure Apache load balancer for your website, on Ubuntu/Debian systems.

1. Install Required Apache modules

We require 4 Apache modules to configure Apache load balancer – *mod_proxy*, *mod_proxy_http*, *mod_proxy_balancer*, *mod_lbmethod_byrequests*

- *mod_proxy* is the main proxy module that redirects requests and allows Apache to act as gateway to backend servers
- *mod_proxy_http* allows support for proxying HTTP requests
- *mod_proxy_balancer* and *mod_lbmethod_byrequests* add load balancing capabilities to Apache web server.

Open terminal and run the following commands to install pre-requisite Apache modules.

```
$ sudo a2enmod proxy
$ sudo a2enmod proxy_http
$ sudo a2enmod proxy_balancer
$ sudo a2enmod lbmethod_byrequests
```

Bonus Read : Apache Reverse Proxy Configuration (http://ubiq.co/tech-blog/apache-reverse-proxy-configuration-step-step/)

2. Restart Apache Server

Restart Apache Server to apply changes

```
$ sudo service apache2 restart
```

3. Setup backend servers

To configure Apache Load Balancer, we will install flask & setup 2 servers to run on port 8080 and port 8081 as a backend server.

```
$ sudo apt-get update
$ sudo apt-get -y install python3-pip
$ sudo pip3 install flask
```

Flask comes with a readymade ~/*backend.py* file that returns "Hello World" on requesting home page.

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
    return 'Hello world!'
```

Bonus Read : How to Install and Configure mod_pagespeed (http://ubiq.co/tech-blog/install-configure-mod_pagespeed-apache/)

We will create a copy of it for our second server.

$ sudo cp ~/backend.py ~/backend1.py

Open backend1.py and change the "Hello World" message in last line to "Hello World 2"

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
    return 'Hello world 2!'
```

Bonus Read : How to Check if mod_expires is enabled (http://ubiq.co/tech-blog/check-mod_expires-enabled/)

We will run the first flask server

```
$ FLASK_APP=~/backend.py flask run --port=8080 >/dev/null 2>&1 &
```

You can test this server by running *curl* command

```
$ curl http://127.0.0.1:8080/
```

You will see the output as **Hello World!**

We will run the second flask server

```
$ FLASK_APP=~/backend1.py flask run --port=8081 >/dev/null 2>&1 &
```

You can test this server by running *curl* command

```
$ curl http://127.0.0.1:8081/
```

You will see the output as **Hello World 2!**

Now we have 2 backend servers ready to handle the load. We will be distributing load between these 2 servers.

Bonus Read : How to Secure Apache with Let's Encrypt on Debian 10 (http://ubiq.co/tech-blog/secure-apache-lets-encrypt-debian-10/)

4. Configure Apache Load Balancer

We need to modify Apache's default configuration file to configure Apache load balancer. Open Apache configuration in text editor

```
$ sudo vi /etc/apache2/sites-available/000-default.conf
```

Add the following lines to *VirtualHost* tag in Apache configuration file.

```
    <Proxy balancer://mycluster>
      BalancerMember http://127.0.0.1:8080
      BalancerMember http://127.0.0.1:8081
     </Proxy>
     ProxyPreserveHost On
     ProxyPass / balancer://mycluster/
     ProxyPassReverse / balancer://mycluster/
```

Let's look at the 3 directives.

- *ProxyPreserveHost* causes Apache to preserve original host header and pass it to back-end servers.
- *ProxyPass* is the main proxy directive which states that everything under root (/) should be directed to back-end cluster (we have named it *mycluster*) of servers. If Apache gets request for /example, then it will send the request to *http://your_backend_server/example*
- *ProxyPassReverse* tells Apache to modify response header in the response received from back-end server. This is useful in case the back-end server returns a location redirect response, then the client will be redirected to Apache proxy server, instead of back-end server.
- We list our backend servers in *Proxy* tag named *balancer://mycluster* . You can change it to anything else. Inside this proxy tag, we list each backend server as *BalancerMember.* So your cluster's *Proxy* tag can have one or more *BalancerMember*

```
<VirtualHost *:80>
...
   <Proxy balancer://mycluster>
      BalancerMember http://127.0.0.1:8080
      BalancerMember http://127.0.0.1:8081
    </Proxy>
    ProxyPreserveHost On
    ProxyPass / balancer://mycluster/
    ProxyPassReverse / balancer://mycluster/
...
</VirtualHost>
```

Bonus Read : How to Enable mod_headers in Apache (http://ubiq.co/tech-blog/how-to-enable-mod_headers-in-apache/)

5. Restart Apache Server

Restart Apache Server to apply changes

```
$ sudo service apache2 restart
```

Hopefully, the above Apache load balancer configuration will help you setup Apache load balancer for your website.

Ubiq (http://ubiq.co/) makes it easy to visualize data in minutes, and monitor in real-time dashboards. Try it (http://ubiq.co/accounts/register) Today!

Related posts:

How to Disable Directory Listing in Apache

(https://ubiq.co/tech-blog/disable-apache-directory-listing/)

How To Install mod_security on CentOS 7

(https://ubiq.co/tech-blog/how-to-install-mod_security-on-centos-7/)

Apache Deny Access to URL, Files & Directory

(https://ubiq.co/tech-blog/apache-deny-access-to-url-files-directory/)

How to Remove Trailing Slash in Apache

(https://ubiq.co/tech-blog/how-to-remove-trailing-slash-in-apache/)

About Author

About Ubiq

Ubiq (https://ubiq.co/) is a powerful dashboard & reporting platform for small & medium businesses. Build dashboards, charts & reports for your business in minutes. Get insights from data quickly. Try it (https://ubiq.co/accounts/register) for free today!

Apache Reverse Proxy Configuration For Ubuntu/Debian Linux (https://ubiq.co/tech-blog/apache-reverse-proxy-configuration-step-step/)

Six Examples of Poor UX Design and What to Avoid (https://ubiq.co/tech-blog/six-examples-of-poor-ux-design-and-what-to-avoid/)