4.0 Setup Apache Web Server on Ubuntu Linux

Install

With Ubuntu installing is very straightforward,

```
sudo apt-get install apache2
```

Apache 2.x is now installed.

> What about Zero Footprint Apache? Definitely doable, but practically with virtualization, and how rarely Apache actually changes right now I'm leaning towards just scripting configuration files only inside of a container.
>
> Having said that, if time permits I might build a BonsaiFramework version.

Test

Verify that the Apache Web Server is running first by hitting your server's IP Address. If you do not know your ip address, at the console type,

```
ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UNKNOWN qlen 1000
    link/ether 40:40:39:1b:ec:30 brd ff:ff:ff:ff:ff:ff
    inet 173.203.126.225/24 brd 173.203.126.255 scope global eth0
    inet6 fe80::4240:39ff:fe1b:ec30/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UNKNOWN qlen 1000
    link/ether 40:40:33:6c:9d:19 brd ff:ff:ff:ff:ff:ff
    inet 10.179.62.235/19 brd 10.179.63.255 scope global eth1
    inet6 fe80::4240:33ff:fe6c:9d19/64 scope link
       valid_lft forever preferred_lft forever
```

Sometimes you may get back more than one IP address if you have more than one Ethernet card. If you are unsure, just try them one at a time in

the next step. In this case mine is **173.203.126.225**.

Launch a browser and enter your ip address into the browser.

You should see a default Apache webpage.

Stopping, Starting, Restarting and Reload

You should know the basic commands to running Apache 2. Go ahead and try them. Note ignore the warning message about "fully qualified domain name" as that is covered in the next section.

As of Ubuntu 12, the following the basic commands to manage Apache2 are,

```
sudo service apache2 stop
sudo service apache2 start
sudo service apache2 restart # restart will restart the service (safer, as
not all services support reload)
sudo service apache2 reload # reload will re-load the configuration files,
with little or no downtime.  Not all services support it (source:
http://askubuntu.com/questions/105200/what-is-the-difference-between-servi
ce-restart-and-service-reload)
```

Before Ubuntu12,

```
sudo /etc/init.d/apache2 stop
sudo /etc/init.d/apache2 start
sudo /etc/init.d/apache2 restart
sudo /etc/init.d/apache2 reload
```

Provide Server Name

> This is now corrected as part of Apache 2.4.18 and onwards.

Apache is working fine, but during restart you will get the warning message, "apache2: Could not reliably determine the server's fully qualified domain name, using ...".

Most websites have a domain name attached to them. Apache is looking for this on start-up. Depending on the version of Apache and Ubuntu this can be resolved by adding the ServerName Directive with the hostname.

You can determine hostname by typing,

```
hostname
```

Adding an entry into the Global Configuration ensures that the change will persist even if Apache is upgraded,

```
# create the configuration file in the "available" section
echo "ServerName localhost" | sudo tee
/etc/apache2/conf-available/servername.conf
# enable it by creating a symlink to it from the "enabled" section
sudo a2enconf servername
```

This concept has changed over time and look here for legacy versions of Apache.

Restart Apache to confirm you do not get the warning messages,

```
sudo service apache2 restart
```

Apache Basic Server Hardening

Before hardening your Web Server you should make sure it works with it's intended integrated purpose in a test environment. Otherwise you may spend lots of wasted time trouble-shooting.

So, assuming that your Web Server passes testing of it's intended purpose, you may perform "Basic Hardening". Because this is "basic" I often perform these all at once and then test.

Here are some of the basic hardening steps I take today by default,

> As with any security notes, I will write a disclaimer that there are more advanced ways to secure Apache. You can go as far as compiling your own custom version but that's out of scope for now.

Disable Server Information Banner

By default Apache provides extra information about your server when 403, 404, 502 or similar error pages are invoked. The information could be used to look up vulnerabilities on the particular version of Apache you are running.

If you visit a page that does not exist you will invoke a 404 error resulting in a page Not Found similar to below,

> Not Found
>
> The requested URL /invalidpage.html was not found on this server.
>
> Apache/2.4.18 (Ubuntu) Server at www.bonsaiframework.com Port 80

Edit **/etc/apache2/conf-available/security.conf**,

set ServerTokens Prod - This turns off all the extra header information sent by Apache.

set ServerSignatures Off - Removes footer information from default apache pages. For example, page not found.

> Older versions of Apache use /etc/apache2/conf.d/security

Restart Apache to take effect and verify by invoking a 404 again.

Disabling Unnecessary Modules

Less loaded, less vulnerabilities and you will also get performance increases too.

Disable Status Module

I found that you can save about 3MB of memory if the status apache module is disabled. Here's how to disable interactively,

```
sudo a2dismod
Your choices are: alias auth_basic authn_file authz_default authz_groupfile
authz_host authz_user autoindex cgid
                  deflate dir env filter jk mime negotiation proxy
proxy_http rewrite setenvif status substitute
Which module(s) do you want to disable (wildcards ok)?
NOTE: make sure you only disable the following one ONLY!!! type:
status
Module status disabled.
To activate the new configuration, you need to run:
  service apache2 restart
sudo service apache2 restart
```

Disable More Modules

Will flush this out some more ...

Turn off Default Website

...

Uninstall Apache Completely

.. these instructions need to be improved, and there is nothing here about removing logs.

1. stop apache:

sudo service apache2 stop

sudo /etc/init.d/apache2 stop

2. remove:

sudo apt-get remove apache2

sudo apt-get purge apache2

References

http://cloudservers.mosso.com/index.php/Ubuntu_-_Apache_configuration#Security_Settings - Rackspace wiki on hardening Apache Web Server.

Apache Web Server Hardening Guide - https://geekflare.com/apache-web-server-hardening-security/