WEEK

3

COMP2241

Web Server Administration

LAB MANUL

- Book: Sams Teach Yourself PHP, MySQL and Apache All in One, 5th Edition
- Week: 3
- Chapters: 3, 25, 29
- Author/s: Rafal Chachura & Maziar Masoudi

- External Sources:
  - Textbook: Sams Teach Yourself PHP, MySQL and Apache All in One
  - https://httpd.apache.org/docs/2.4/mod/quickreference.html
  - https://vim.wikia.com/wiki/Tutorial
  - https://en.wikipedia.org/wiki/Hosts_(file)

Table of Contents

Contents

Working With Apache Directives

This lab will focus on configuring Apache using various directives. For detailed information on the directives used, you can consult the Directive Quick Reference.

The textbook contains a guide describing each of the sections of the directive reference towards the end of chapter 3 (p. 46)

Preparation

To successfully complete this manual, you will need to modify various text based configuration files from the terminal. This may be easier if the text editor you are using has some form of syntax highlighting, a search feature, and easy navigation through large files.

Vim is such an editor. It is not required; you can choose to use any editor that you like. Vim can be installed on your Ubuntu environment using the package manager by typing the following commands:

```
sudo apt-get install vim
```

There are many tutorials that describe how to use vim online. If you are using a different text editor replace all the occurrences of vim in the following steps with any other editor.

In order to continue with the rest of the manual, you must have a working copy of Apache installed. For detailed steps see the lab manual from week 2.

For a better understanding of directives and containers in general, read pages 45 – 50 in the text book.

Most of the commands require root access, so before you start any of the sections, make sure you run:

```
sudo su
```

You will be making changes to Apache's configuration files. It is a good idea to back up the original configuration should you want to revert your changes.

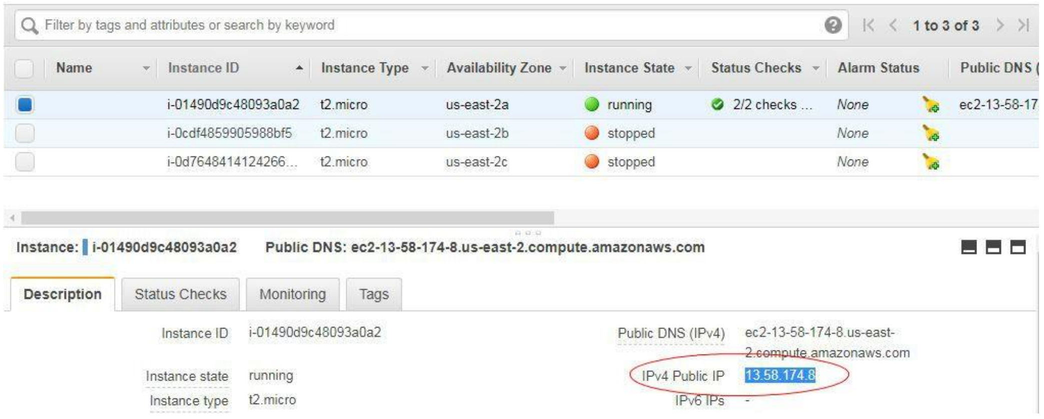
```
cd /usr/local/apache2/conf
cp httpd.conf httpd.conf.bak
```

A. Listen Directive

The Listen directive instructs Apache httpd to listen to only specific IP addresses or ports. If only a port is specified, the server listens to the given port on all interfaces. The default port is 80. Follow the instructions in **Lab Manual 3 - Supplemental** before continuing!

1. First make sure that Apache is running:
   ```
   /usr/local/apache2/bin/apachectl start
   ```

2. Find out the server's Public IP Address in your EC2 control Panel:



Filter by tags and attributes or search by keyword — 1 to 3 of 3

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (
✓		i-01490d9c48093a0a2	t2.micro	us-east-2a	● running	✓ 2/2 checks ...	None	ec2-13-58-17
		i-0cdf4859905988bf5	t2.micro	us-east-2b	● stopped		None	
		i-0d7648414124266...	t2.micro	us-east-2c	● stopped		None	

Instance: i-01490d9c48093a0a2 Public DNS: ec2-13-58-174-8.us-east-2.compute.amazonaws.com

Instance ID	i-01490d9c48093a0a2	Public DNS (IPv4)	ec2-13-58-174-8.us-east-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.58.174.8
Instance type	t2.micro	IPv6 IPs	-

3. Make sure that you can access your server from outside by typing the highlighted IP address into your favourite browser. Keep the browser open, and go back to your Ubuntu environment.

4. Open the main Apache configuration file:
   ```
   vim /usr/local/apache2/conf/httpd.conf
   ```

5. Look for the Listen directive, and change the port that is specified to some other number

   `Listen 80` ←Original

   `Listen 899` ←Changed

6. Save your changes and restart apache
   ```
   /usr/local/apache2/bin/apachectl restart
   ```

7. Go back to your open browser and refresh the page. You should see a 404 error. Append the port number that you specified in step 5 after the IP address in the address bar. For example:

   192.168.6.128:**899** *(replace 192.168.6.128 with your server's IP)*

8. You should see the contents of index.html
9. Open httpd.conf and change the listen directive back to 80
10. Don't forget to restart apache.

B. Authentication and Authorization

It is possible to password protect various directories on your server. You can do this with the help of a password file that holds the credentials that you would like to allow on your server and a few directives.

1. Create a directory in your web root that you would like to protect
   ```
   mkdir /usr/local/apache2/htdocs/restricted
   ```

2. Add a sample index.html file that will be loaded whenever someone navigates to the restricted folder
   ```
   cd /usr/local/apache2/htdocs/restricted
   echo this is a restricted area > index.html
   ```

3. Navigate to the not quite restricted folder you just created using your favourite browser. Leave the browser open to the address below and return to Ubuntu for the next step

   192.168.6.128/restricted *(replace 192.168.6.128 with your server's IP)*

4. Create a password file using the htpasswd binary. You can replace *username* with any user name ex: John. You will be prompted for the password twice after the following command.

```
/usr/local/apache2/bin/htpasswd -c /usr/local/apache2/conf/.htusers username
```

5. Open the main Apache configuration file
   ```
   vim /usr/local/apache2/conf/httpd.conf
   ```

6. Navigate to the bottom of the file and add a new directory container:
   ```
   <Directory /usr/local/apache2/htdocs/restricted>
   </Directory>
   ```

7. Inside the container add the following directives:

```
AuthType Basic
AuthName "Restricted Area"
AuthUserFile /usr/local/apache2/conf/.htusers
Require valid-user
```

8. Save the file and restart Apache
9. Attempt to access the restricted area with the open browser. Notice that you are now required to provide login credentials to gain access.

C. Authentication and Authorization (Directory Level Override)

The above authentication can be applied on a per directory basis without modifying httpd.conf. Before this process works, we need to enable directory overrides inside the main configuration file.

1. Open the main Apache configuration file

```
vim /usr/local/apache2/conf/httpd.conf
```

2. Look for the directory container that describes the webroot:

```
<Directory /usr/local/apache2/htdocs>
…
</Directory>
```

3. Find the directive AllowOverride and change the value to the following, then save the file and exit the text editor.

```
AllowOverride All
```

4. Create and navigate to a new directory that you would like to protect

```
mkdir /usr/local/apache2/htdocs/area51
cd /usr/local/apache2/htdocs/area51
```

5. Create a directory override file

```
vim .htaccess
```

6. Inside the text editor repeat the directives that were used in step B7

```
        AuthType Basic
       AuthName "Area 51"
       AuthUserFile /usr/local/apache2/conf/.htusers
       Require valid-user
```

7. Save the file and restart Apache

8. Try accessing the folder in your browser by going to:

   192.168.6.128/area51   *(replace 192.168.6.128 with your server's IP)*

   You should be prompted for a username and password.

D. Controlling Access Based on Referrer

Using certain apache directives, it is possible to allow or deny access to resources based on where the request originated. The origin can be defined by an IP address or domain name. In the following exercise you will disallow access from your mobile phone.

1. Disconnect your phone from Wi-Fi.
2. On your phone, open a web browser and go to www.whatismyip.com.
3. Make note of your IP address (i.e. 192.168.56.123). This address will be referred to as <IP> in the next steps.

4. In the Ubuntu environment create a private directory inside your webroot

```
       mkdir /usr/local/apache2/htdocs/private
```

```
echo Private area > /usr/local/apache2/htdocs/private/index.html
```

5. open Apache's main configuration file

```
       vim /usr/local/apache2/conf/httpd.conf
```

6. Create a new location container at the end of the file that points to the private path you just created

```
        <Location /private>
       </Location>
```

7. Place the following directives inside the location container. Where <IP> refers to the host environment's IP you found in step 1

   ```
   Order Allow,Deny
   Allow from all
   Deny from <IP>
   ```

8. Save the file and restart Apache.
9. Attempt to access /private from your phone. If you entered the IP address correctly, you should not be allowed.
10. Go back to Apache's configuration file and remove the last line in the location container you just created. Save the file and restart Apache. You are now allowed to view the private location from your phone.

E. Creating Name-Based Virtual Hosts

Instead of always using the IP address of your apache server in the address bar, you can configure apache to listen to a domain name that you define. This requires configuring Apache, and your host environment.

1. Create a new directory in your web root that you would like to point to with virtual host

   ```
   mkdir /usr/local/apache2/htdocs/mysite
   ```

```
echo my first virtual host > /usr/local/apache2/htdocs/mysite/index.html
```

2. Open up Apache's main configuration file

   ```
   vim /usr/local/apache2/conf/httpd.conf
   ```

3. Look for and uncomment the following line:

   ```
   #include conf/extra/httpd-vhosts.conf
   ```

4. Save and close this file and open up the virtual host configuration file

   ```
   vim /usr/local/apache2/conf/extra/httpd-
   vhosts.conf
   ```

5. You can use this file to add as many virtual host directives as you like. First comment the two sample directives and create your own virtual host container

   ```
   <VirtualHost *:80>
   </VirtualHost>
   ```

6. Inside the container, you need to insert at least two directives to define your virtual host:

```
DocumentRoot /usr/local/apache2/htdocs/mysite
ServerName mysite.local
```

7. Save and close this file, and restart Apache.

Inside your host environment, you will have to modify your hosts file so that mysite.local points to the correct IP address. You can visit https://www.howtogeek.com/howto/27350/beginner-geek-how-to-edit-your-hosts-file/ for instructions on modifying the hosts file in Windows, Mac and Linux.

8. Add the following line to the hosts file, where <SERVIP> is the Pubic IP address of your server.

```
<SERVIP> mysite.local
```

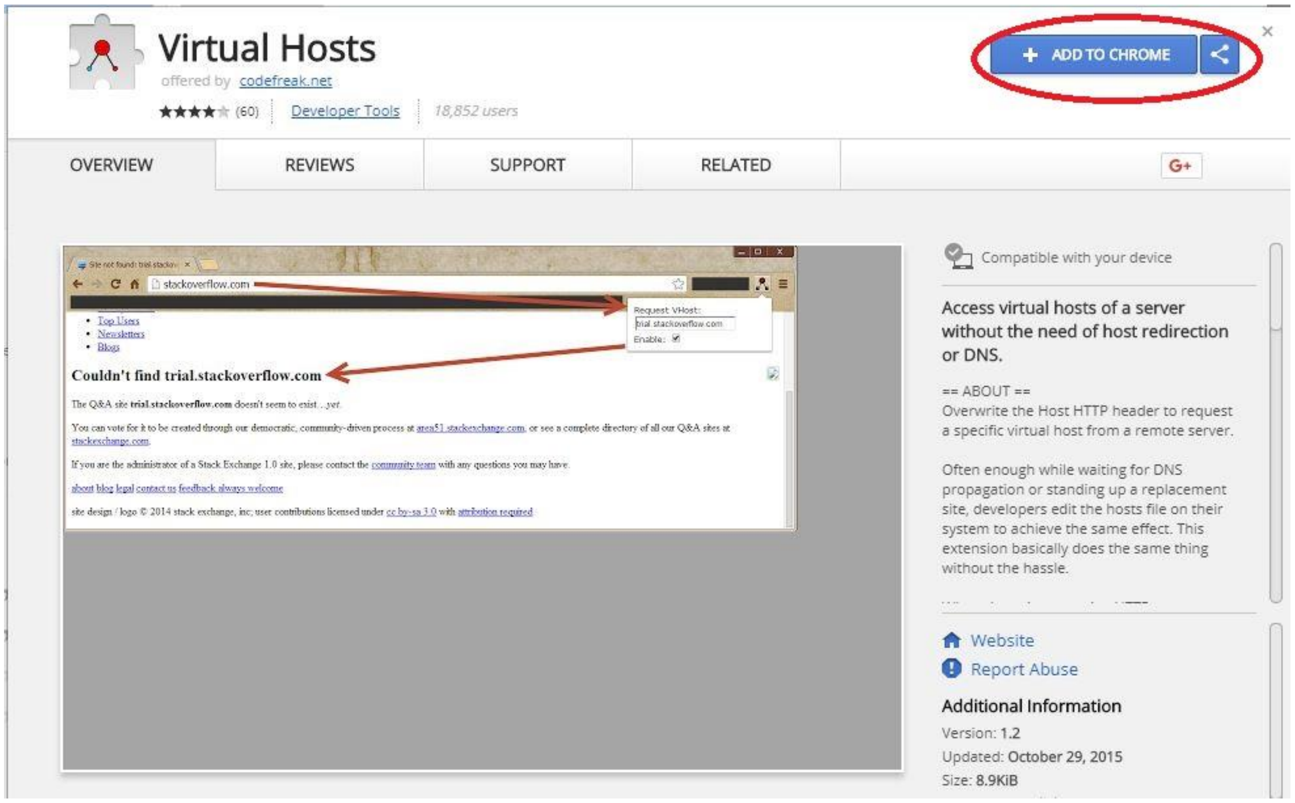
If you have administrator access to your computer's hosts file, and were able to add the line above, skip to step 14.

If you do not have access to your computer's hosts file, please follow the instructions below to add a Virtual Hosts Extension to Chrome:

9. Open Chrome and do a search for "Virtual Hosts Extension". Alternatively, you can also click on this link to take you directly to the Chrome Web Store:

https://chrome.google.com/webstore/detail/virtual-hosts/aiehidpclglccialeifedhajckcpedom?hl=en

10. Click the **ADD TO CHROME** button.



Couldn't find trial.stackoverflow.com

The Q&A site **trial.stackoverflow.com** doesn't seem to exist... *yet*.

You can vote for it to be created through our democratic, community-driven process at area51.stackexchange.com, or see a complete directory of all our Q&A sites at stackexchange.com.

If you are the administrator of a Stack Exchange 1.0 site, please contact the community team with any questions you may have.

about blog legal contact us feedback always welcome

site design / logo © 2014 stack exchange, inc; user contributions licensed under cc by-sa 3.0 with attribution required

Compatible with your device

Access virtual hosts of a server without the need of host redirection or DNS.

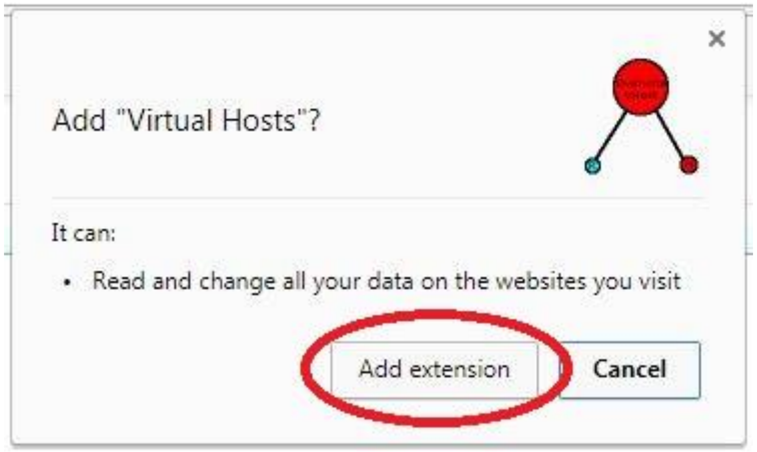
== ABOUT ==
Overwrite the Host HTTP header to request a specific virtual host from a remote server.

Often enough while waiting for DNS propagation or standing up a replacement site, developers edit the hosts file on their system to achieve the same effect. This extension basically does the same thing without the hassle.

Website
Report Abuse

Additional Information
Version: 1.2
Updated: October 29, 2015
Size: 8.9KiB

11. Click **Add Extension:**



Add "Virtual Hosts"?

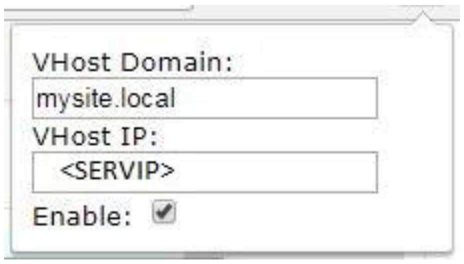
It can:

- Read and change all your data on the websites you visit

Add extension   Cancel

12. Click on the new Virtual Hosts icon beside the address bar in Chrome.

13. Type in the following information, replacing <SERVIP> with the Public IP address of your server:



VHost Domain: mysite.local
VHost IP: <SERVIP>
Enable: ✓

14. If you are using the **Virtual Hosts** Extension, type in the following URL in Chrome. If you edited the hosts file, you can use the browser of your choice (remember to include **http://**).

```
http://mysite.local
```

15. You can add as many virtual hosts as you like. Just remember that for each virtual host, you need to create a VirtualHost container in apache, and modify the hosts file or Virtual Hosts Extension in Chrome.