4.1 Apache Virtual Hosting

- Linux Containers
- What is Virtual Hosting?
- Apache Differences on Ubuntu
- My Virtual Hosting Strategy
- Virtual Hosting
    - Setup Virtual Hosting Directories
    - Create Virtual Host Configuration File
    - Enable Virtual Host
    - Disable Virtual Host
    - Under the Covers
- Testing
    - Without Real Domain Names
    - With Real Domain Names
- Cleanup
- Next
- Resources

Linux Containers

Linux containers (LXC) may be a game changer in this area that finally does what I want in terms of isolation by user communities as ACL's do not work as expected. I'm going over how I can make this all work.

What is Virtual Hosting?

Virtual hosting configures Apache to be able to host more than one website on the same computer. As an example, let's say this Apache Web Server will host both www.earth.com and www.krypton.com. Here is how the process works,

1. Register domain names (in this example) www.earth.com and www.krypton.com (You may skip this step if you just testing)
2. Point to the server's ip address.
3. User types in either url into their browser which would send a request to Apache.
4. Apache reads the request header (which contains the url) where the user wanted to go.
5. Based on the url, apache checks for a matching virtual host entry and directs the user to that virtual server's home directory.

> In the future, I will put a diagram to illustrate.

Apache Differences on Ubuntu

Ubuntu/Debian organizes things slightly different than other systems when it comes to Apache. If you read other website that talk about Apache you might get confused. So here's how Ubuntu does things. You have the following key files,

/etc/apache2/apache2.conf - this is the default file provided during install and contains the default settings. If possible do not modify this file.
/etc/apache2/httpd.conf - location for global user configured options.
/etc/apache2/site-available/* - this is where you store your different virtual hosts.
/etc/apache2/sites-enabled/* - symbolic links here used to enable sites from site-available.

This is enough to get us started, but feel free to read more details at Control-Escape.

My Virtual Hosting Strategy

There are so many different ways of doing this it's quite mind boggling. Here's my overall strategy:

- I want clients to come in and manage the pure html aspects of their website, so each website will have it's own group and clients will belong to the website's group.
- Logging will be Virtual Host Based Logging. The pros and cons are are discussed in Apache Log Management.
- Each virtual host will also have a,
    - browse-able "shared" folder to easily distribute files
    - "shared.private' where .htaccess is enabled so users can set their own authentication parameters and indexing
- Most people only have one IP address to I'll use the "name based" hosting approach.

I find this approach is complex enough to address the needs of most applications and at the same simple enough to implement for the intermediate level user.

To segregate users I've tried ACLs for a long time but it's not workable. Instead for a simple system, I use basic Linux permissions and more advanced systems, I use a using Linux Containers or Docker.

Virtual Hosting

> There is another approach to this (provided most virtual hosts have the same requirements) where virtual hosts are created through mod rewrite, convention and customization is achieved using .htaccess or <Directory>.

Looking in /etc/apache2/apache2.conf you will see a reference to the directory, /etc/apache2/sites-enabled/. Apache will look in this directory and load any virtual host file configurations.

Setup Virtual Hosting Directories

Now we setup the directories to be used by Apache where your html files are kept.

> The entire sprit of what I'm trying to do below follows least access privileges using base Linux permissions. I tried using advanced Linux permissions, ACLs, but even that technology is too limiting. However, there is a much better approach if you can use container technology as you're then limiting the entire OS experience for a user to their particular directory. I'll write this up in the future, but only if there is someone interested.

Assuming you are logged in as a member of the staff group, we will be creating groups and users with reserved ids as mentioned in the basic setup,

```
cd /opt/web
sudo mkdir www.krypton.com # Home directory for the website.

cd /opt/web/www.krypton.com
sudo mkdir www # Folder for static content
sudo addgroup --gid 3100 wgkryptonian # Special work group to distinguish
users who should have access to the website.

cd /opt/web
sudo chown -R serveradmin:wgkryptonian ./www.krypton.com/
sudo chmod -R o-w ./www.krypton.com/ # Make sure others can't change files.
# Ensure setgid bit is setup so new files created will have same groups.

sudo find ./www.krypton.com/ -type d | sudo xargs -I{} chmod g+s {}


# Repeat for www.earth.com
cd /opt/web
sudo mkdir -p www.earth.com/www # Makes both directories with one command
sudo addgroup --gid 3101 wgearthling
sudo chown -R serveradmin:wgearthling ./www.earth.com/
sudo chmod -R o-w ./www.earth.com/ # Make sure others can't change files.
sudo find ./www.earth.com/ -type d | sudo xargs -I{} chmod g+s {}
```

The basic file permissions are pretty straight-forward. Enforcing group permissions of newly created files is not so straight forward. If you do not understand the limitations or how the command works, read the Bonsaframework setgid specifically the section around folders.

Now we create users that will have access to their respective websites,

```
sudo useradd -d /opt/web/kalel -m -u 2500 -G wgkryptonian -c "Client" -s
/bin/bash kalel # Add user kalel with additional group and make the user's
home directory
sudo useradd -d /opt/web/jorel -m -u 2501 -G wgkryptonian -c "Client" -s
/bin/bash jorel

sudo passwd kalel
sudo passwd jorel

sudo useradd -d /opt/web/loislane -u 2502 -m -G wgearthling -c "Client" -s
/bin/bash loislane
sudo useradd -d /opt/web/jimmyolsen -u 2503 -m -G wgearthling -c "Client"
-s /bin/bash jimmyolsen

sudo passwd loislane
sudo passwd jimmyolsen
```

Create Virtual Host Configuration File

Next you create your physical virtual host file in /etc/apache2/sites-available and then create a symbolic link in /etc/apache2/sites-enabled/. The file must have a .conf extension.

Start with the default virtual host file as a template as it changes over time,

```
cd /etc/apache2/sites-available
sudo cp 000-default.conf www.krypton.com.conf
sudo cp 000-default.conf www.earth.com.conf
```

sudo Edit **www.krypton.com.conf** and remove all the extra lines and modify the matching lines,

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost

    ServerName www.krypton.com
    ServerAlias krypton.com

    DocumentRoot /opt/www.krypton.com/www

    # This restrictive a precedence for ALL directory blocks.
    <Directory />
        Options FollowSymLinks
        # This prevents use of .htaccess
        AllowOverride None
    </Directory>

    # Main location of static content for the websites.
    <Directory /opt/www.krypton.com/www/>
        Options +MultiViews
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/www.krypton.com.error.log

    # Possible values include: debug, info, notice, warn, error, crit,
alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/www.krypton.com.access.log combined

</VirtualHost>
```

Apache 2.2 and older
In Apache 2.2 and older

```
Require all granted
```

is changed to

```
Order Allow,Deny
Allow from all
```

Some notes on not so obvious entries in the virtual host file,

- MultiViews - uses Content Negotiation choose the best representation of a resource based on the browser-supplied preferences for media type, languages, character set and encoding.
- A keen eye will notice the + symbol in front of MultiViews. The plus symbol indicates we are adding to the existing Options inherited from parent blocks rather than resetting.
- combined - This is a predefined log format set by LogFormat in apache2.conf

The log files are stored in /var/log/apache2/ where Ubuntu already has script to manage log rotation.

There is a disadvantage with specifying specific log files per virtual hosts because you can run out of file descriptors. The pro of course is simplicity and easy separation of your logs. There may be alternatives but don't hold your breath for me to find a solution and publish it. My clients rarely keep more than 3 virtual sites on the same machine.

Repeat for www.earth.com changing the directory and domain names values as required.

> Somebody drop in here command line to do this quickly with sed.

Enable Virtual Host

Just because you created the virtual host does not mean it is enabled. To enable the virtual hosting,

```
cd /etc/apache2/sites-available/ # must be in this directory for a2ensite
to work
sudo a2ensite www.krypton.com.conf # enable a virtual host
sudo a2ensite www.earth.com.conf # enable a virtual host
```

Notice the **conf** extension to the file. If you do not put that, you will recieve the error "ERROR: Site www.krypton.com does not exist!".

> As an side note, a2ensite is a Ubuntu shortcut command which creates a symbolic link. It is exactly the same things as doing this,
>
> cd /etc/apache2/sites-enabled/
> sudo ln -s ../sites-available/krypton.com ./krypton.com

You will then be prompted to reload the Apache configuration file for changes to take effect. This command is useful because it does not affect users currently browsing your other sites. However, I have found this sometimes does not work for me. In that case, I usually issue a full restart,

```
sudo service apache2 stop
sudo service apache2 start
```

Before Ubuntu 12 introduced the service approach you would execute,

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

Type in your browser, www.krypton.com. Because directory listing is enabled and there is no default html page usually index.html you should see an directory page listing the contents of **/opt/web/serveradmin/www.krypton.com/**.

Disable Virtual Host

You can disable your reference using the equivalent *sudo a2dissite* .Again you must must restart Apache for the changes to take effect.

Under the Covers

The Ubuntu packaging enabled certain directives for you. If you are using a vanilla implementation of Apache you might need to do some more work namely using the Listen and NameVirtualHost directives.

Ubuntu already added these directives as shown,

```
tpham@krypton:/etc/apache2$ find . -type f | xargs grep -i listen
./ports.conf:Listen 80
./ports.conf:    Listen 443
tpham@krypton:/etc/apache2$ find . -type f | xargs grep -i namevirtualhost
./ports.conf:NameVirtualHost *:80
./ports.conf:    # NameVirtualHost statement here
tpham@krypton:/etc/apache2$
```

Testing

Without Real Domain Names

... host file ... Roderick will fill in the details.

With Real Domain Names

... put steps on verifying domain name points to proper ip, also link to article on how to setup DNS... Roderick to fill in.

Cleanup

You should do the following,

- Get rid of favicon error in your logs.
- ...

Next

There's much more to Apache than this. For example, we could set up public and private Online Shares. If you are serving real traffic you might want to read the next step about configuring logging. If you are just playing around then you can skip to setting up an application server.

Resources

http://httpd.apache.org/docs/2.0/vhosts/examples.html - official examples from Apache
http://mail-archives.apache.org/mod_mbox/httpd-users/200603.mbox/%3C200603161214.10191.mymaillists@gmx.at%3E - good working example of how to do virtual hosting with different ports.
http://httpd.apache.org/docs/2.4/upgrading.html#access - New Virtual host configuration Apache 2.4