

# Apache and...

**Virtual Hosts ---- aliases  
mod\_rewrite ---- htaccess**

**AFNOG Chix 2011  
Blantyre, Malawi  
31<sup>st</sup> Oct - 4<sup>th</sup> Nov 2011**

**Dorcas Muthoni**

Courtesy: Hervey Allen

# What is Apache?

**Very good overview here:**

[http://en.wikipedia.org/wiki/Apache\\_web\\_server](http://en.wikipedia.org/wiki/Apache_web_server)

The Apache web site is an excellent source of information as well:

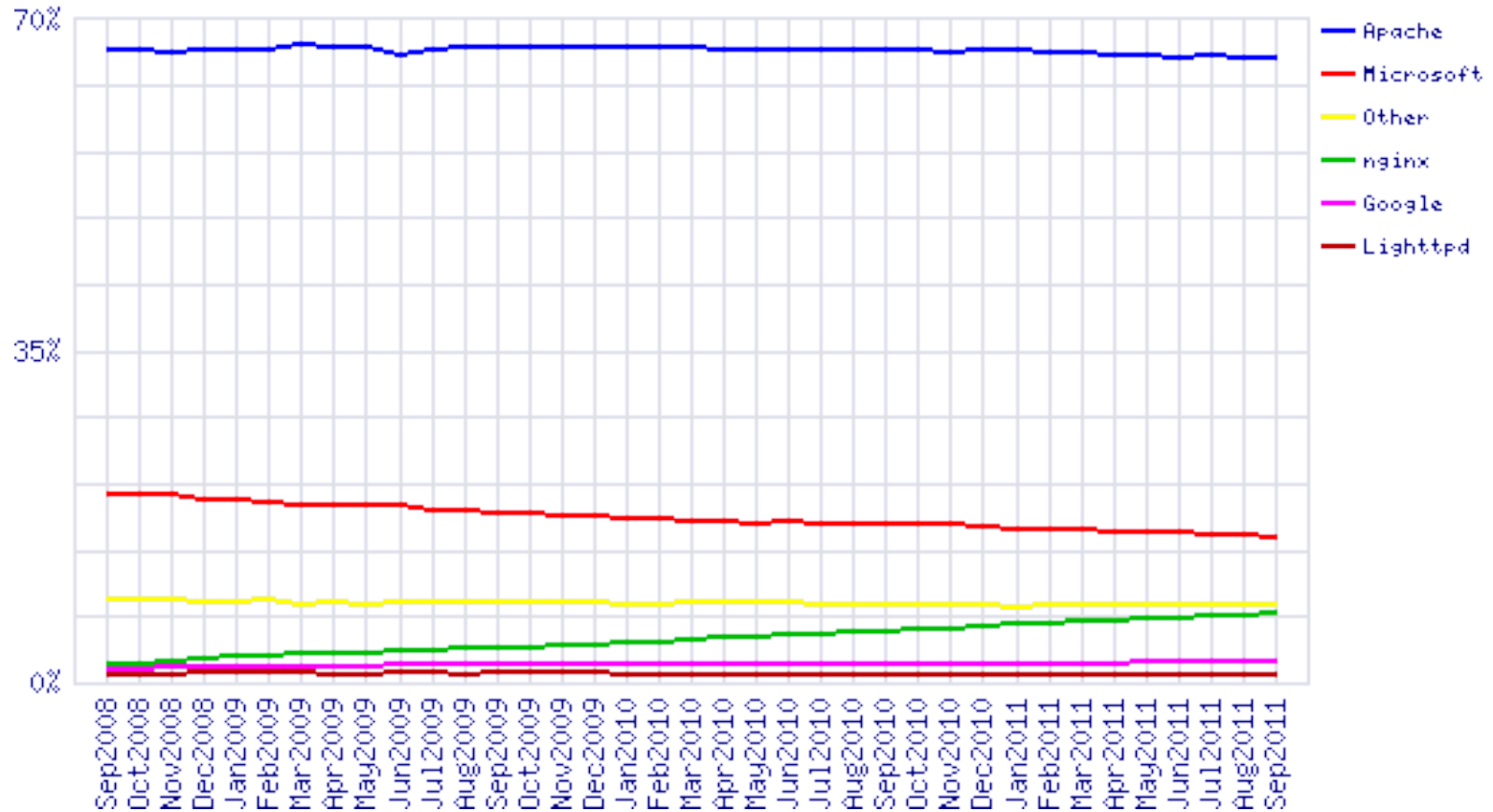
<http://www.apache.org/>



# Quick Facts

- Initially released in 1995
- Used on over 100 million web sites
- 65% market share. Microsoft is 15%.- Oct 2011
- One million *busiest sites*, Apache 66.82%, Microsoft 16.87%
- *Cross platform*: Runs on Unix, Linux, FreeBSD, Solaris, Netware, Mac OS X, Windows, OS/2 and more.
- Licensed under the *Apache License*. Incompatible with GPL version 2, compatible with version 3.

# Sept 2010-2011 Statistics



<http://news.netcraft.com/archives/category/web-server-survey/>

# What is a Virtual Host?

There are two types:

- Name-based
- IP-based

We will be configuring named-based virtual hosts.

This allows a **single IP address to serve many web sites from a single server**. This is possible because the web client sends the name of the site it wishes to connect to as part of its initial connection request.

# IP-based Hosting

- This requires a separate IP address for each hostname on a web server.
- IP-based hosting works with current SSL implementations.
- IP-based hosting (can) work even if DNS has failed.
- However, **requires an IP address for each site**. This may not be possible and requires more effort to implement.

# Configuration Details: Apache

<b>Primary Configuration file</b>	<code>/usr/local/etc/apache22/httpd.conf</code>
<b>Where your website files are stored</b>	DocumentRoot Default is usually <code>"/usr/local/www/apache22/data"</code>
<b>File that Apache will serve if a directory is requested</b>	DirectoryIndex Default is usually <b>index.html</b> Others can be <code>index.php</code> or <code>index.htm</code> etc
<b>Listen port</b>	<b>Listen 80</b> You can also bind apache to a port, IP or both e.g. <code>Listen 12.34.56.78:80</code>
<b>Supplemental configuration</b>	The configuration files in the <code>etc/apache22/extra/</code> directory can be included to <b>add extra features or to modify the default configuration</b>  Include <code>etc/apache22/extra/httpd-vhosts.conf</code>

# Configuration Considerations: Apache

- **Directory naming conventions.** Decide upon one from the start:
  - **/usr/local**/www/share/?? (FreeBSD)
  - **/var**/www/share/?? (Linux)
- What to do about default actions? We'll give an example in our exercises.
- Must deal with directory permissions in more detail.



**Questions?**

**?**

# Other Popular Apache Items

**Three include:**

- aliases
- mod\_rewrite
- htaccess

# Aliases

Allows you to specify a web directory name that maps to a separate directory *outside* the file structure of a web site.

## For example:

Your site is `http://www.example.com/`

The site resides in `/usr/local/www/share/default/`, but you want the files in `/usr/local/www/books/` to be available at `http://www.example.com/books/`

## How would you do this?

# Aliases continued

In the file `httpd.conf...`

```
Alias /books /usr/local/www/share/books
```

But, you must set Directory permissions as well. For instance:

```
<Directory "/usr/local/www/share/books">  
    Options Indexes FollowSymLinks  
    AllowOverride None  
    Order allow,deny  
    Allow from all  
</Directory>
```

**Remember, case counts in Apache configuration files!**

# mod\_rewrite

Allows you to redirect requests from a page, or a pattern of pages to another page, or another pattern of pages.

- Extremely powerful
- Uses regular expression language
- Can save you if

In order to use `mod_rewrite` the rewrite module must be part of your Apache install (it is in FreeBSD 8.2 and Apache 2.2), and it must be loaded in the `httpd.conf` file:

```
LoadModule rewrite_module modules/mod_rewrite.so
```

# mod\_rewrite continued

Here is some sample code where `mod_rewrite` is actually used (from `httpd.conf`):

```
# turn on the use of the mod_rewrite module
    RewriteEngine on

# Redirect old style ISO=NN requests
    RewriteRule ^\s/db\s/lookup\s/ISO=([A-Z]) \s
    /db/lookup/redirect.php
```

The end result of this is the redirect reforms the requests in to the form:

`http://nsrc.org/db/lookup/country.php?ISO=eg`

or

`http://nsrc.org/db/lookup/provider.php? \`

`id=89733450039&fromISO=eg`

# htaccess

Perhaps the most common use of `mod_rewrite` is to force the use of `https` for a set of pages – such as a site login page.

**Here is an example:**

```
# Turn on the rewrite engine.

# If we are not using port 443 (ssl) AND
# We are trying to access something under the /trac directory AND
# We are NOT trying to open the initial index.php file (to avoid
# infinite redirects), THEN keep the URI and force the user to use
# SSL. Too many passwords and sensitive info are thrown around on
# the trac project pages.

RewriteEngine on

RewriteCond %{SERVER_PORT} !443
RewriteCond %{REQUEST_URI} ^/trac
RewriteCond %{REQUEST_URI} !^/trac/index.php
RewriteRule ^(.*)$ https://ws.edu.isoc.org$1 [R=301]
```

# htaccess continued

Then you must create a file “.htaccess” in the directory you wish to protect. In that file you might have something like this:

```
AuthName "AfNOGChix 2011 Malawi, Trac Access"  
AuthType Basic  
AuthUserFile /var/www/html/trac/afnogchix2011/.htpasswd  
require user afnogchix
```

Note the file “.htpasswd” above. This is where you store user/password information. You do this by running and using the `htpasswd` command.



# htpasswd command

To create an initial `.htpasswd` file with a user and password you do:

```
# htpasswd -c .htpasswd username
```

The “-c” parameter says to create the file. Enter in the password when prompted. For the next user do:

```
# htpasswd .htpasswd username
```

To change a password just run the command again.

And, in the end you'll see a prompt like this...

# htaccess



## Questions?