HTTP: LAMP

This material is based on work supported by the National Science Foundation under Grant No. 0802551

C3L9S1

Lesson Overview

As Linux continues to gain market share, administrators need to understand the various server components available for Linux and how each may be implemented to meet business needs.

One example of server components combined with Linux is LAMP—an acronym that describes the combination of the **L**inux operating system, **A**pache Web Server, **M**ySql database, and **P**HP scripting capabilities in one bundle for easy installation configuration, and maintenance.

LAMP services are also versatile enough to accommodate multiple domain name hosting through proper virtual domain configuration. Savvy students will need to be familiar with all aspects of LAMP and Virtual Domain configurations to be competitive with their peers in a fierce job market.

Select **PLAY** below to review the lesson introduction:

View Video
VideoLesson9Intro(C3L9S2)
.mp4

Objective

You should know what will be expected of you when you complete this lesson. These expectations are presented as objectives. Objectives are short statements of expectations that tell you what you must be able to do, perform, learn, or adjust after reviewing the lesson.

Lesson Objective:

Given the need to create a LAMP server in a virtual domain, the student will configure a LAMP server in a virtual domain.

Lesson Outline

In this lesson, you will explore:

- LAMP
  - Linux
  - Apache
  - MySQL
  - PHP
  - LAMP use and comparison
  - XAMPP
- Virtual Hosting
  - Setup & configuration
  - Testing

LAMP

LAMP is an acronym for **L**inux, **A**pache, **M**ySql and **P**HP. Some people argue that the "P"may represent Python or Perl, but they are incorrect. A properly configured LAMP system can provide web hosting companies an opportunity to host and deliver both dynamic and database driven web sites. These sites can include shopping carts, educational media, medical technologies, and real estate sites. As such, LAMP is considered by many to be the platform of choice for development and deployment of high performance web applications.

LAMP is a proven robust combination of applications driven on the powerful Linux operating system. Preferred by a majority of the system administrators in the IT field, this combination of technologies has a proven track record of being secure and efficient. Additionally, all major components are absolutely free!

When compared to proprietary applications, LAMP far exceeds most of their components and sets the benchmark for industry standards. The Linux/Apache combination currently serves more web pages than all other commercial and open source solutions combined. MySql is the fastest open source database available and easily compares to the popular Oracle database application. PHP is also recognized as the fastest server-side scripting program on the planet and exceeds the capabilities of ASP, JAVA, .NET and Cold Fusion. PHP can be used for any application including web carts, email services, blogs, and online photo albums. PHP is very dynamic and powerful.

Select **PLAY** below to learn about Apache:

View Video VideoLesson9Lamp(C3L9S5).mp4

Linux

Linux is the operating system used to run the applications (Apache, MySql and PHP).

Linux is recognized for speed, minimum hardware requirements, security and remote administration capabilities.

Linux can be streamlined by limiting users to only command line management or a GUI can be installed depending on the preference of end users.

The most attractive aspect of Linux is the accompanying free price tag!

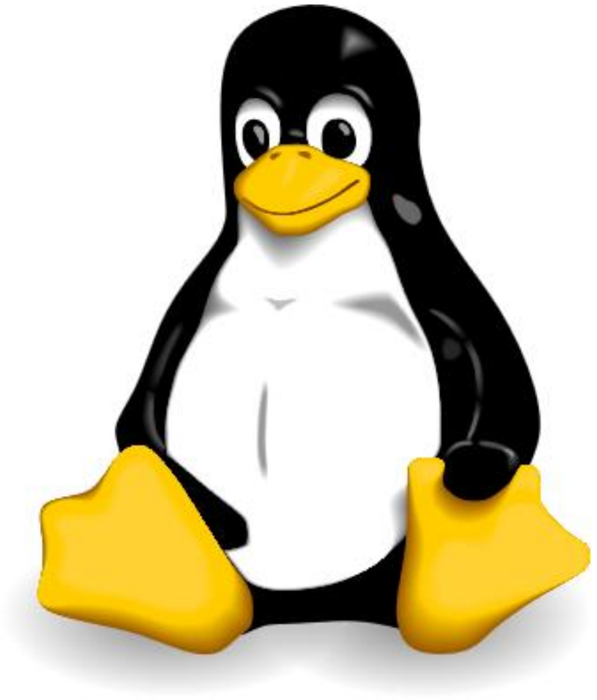


Image Credit: Larry Ewing, Simon Budig, Anja Gerwinski

For Review:
- Linux
- Introduction to Linux

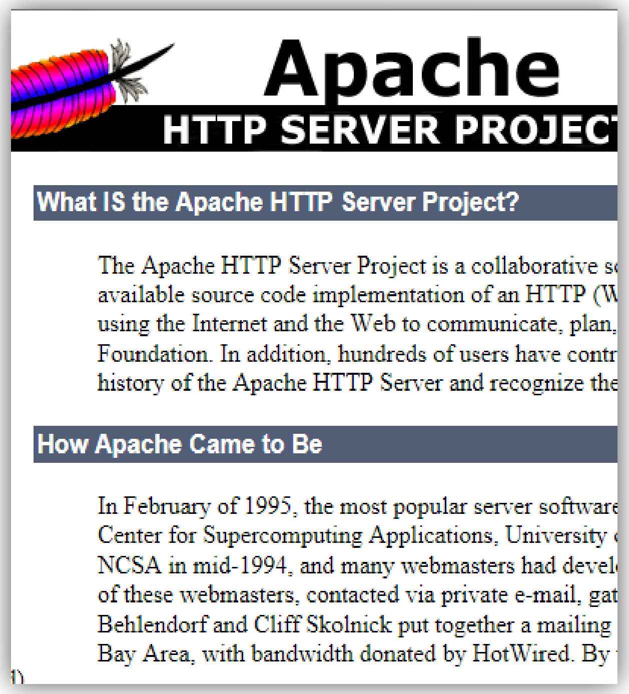
Apache

Developed by the Apache Software Foundation (ASF), Apache is an open source web server solution packed with features. Apache is also extremely fast and works well with Linux. Apache may also be used within the Windows operating system environment but the overhead required by the Windows limits the potential of Apache.

Apache has a virtual hosting function that allows you to host several websites from a single server. In 1996, Apache took the number one spot as the world's most popular web server and has held that position ever since.

Apache's features include:

- Enhanced logging for each host
- Bandwidth throttling
- Directory access protection
- Common Gateway Interface (CGI) support
- Secure Sockets Layer (SSL) support
- IP or name-based virtual hosts
- Individual document root for each host
- Folder sharing between hosts using alias directives
- Override capabilities per directory or virtual host



Apache
HTTP SERVER PROJEC

What IS the Apache HTTP Server Project?

The Apache HTTP Server Project is a collaborative s
available source code implementation of an HTTP (W
using the Internet and the Web to communicate, plan,
Foundation. In addition, hundreds of users have contr
history of the Apache HTTP Server and recognize the

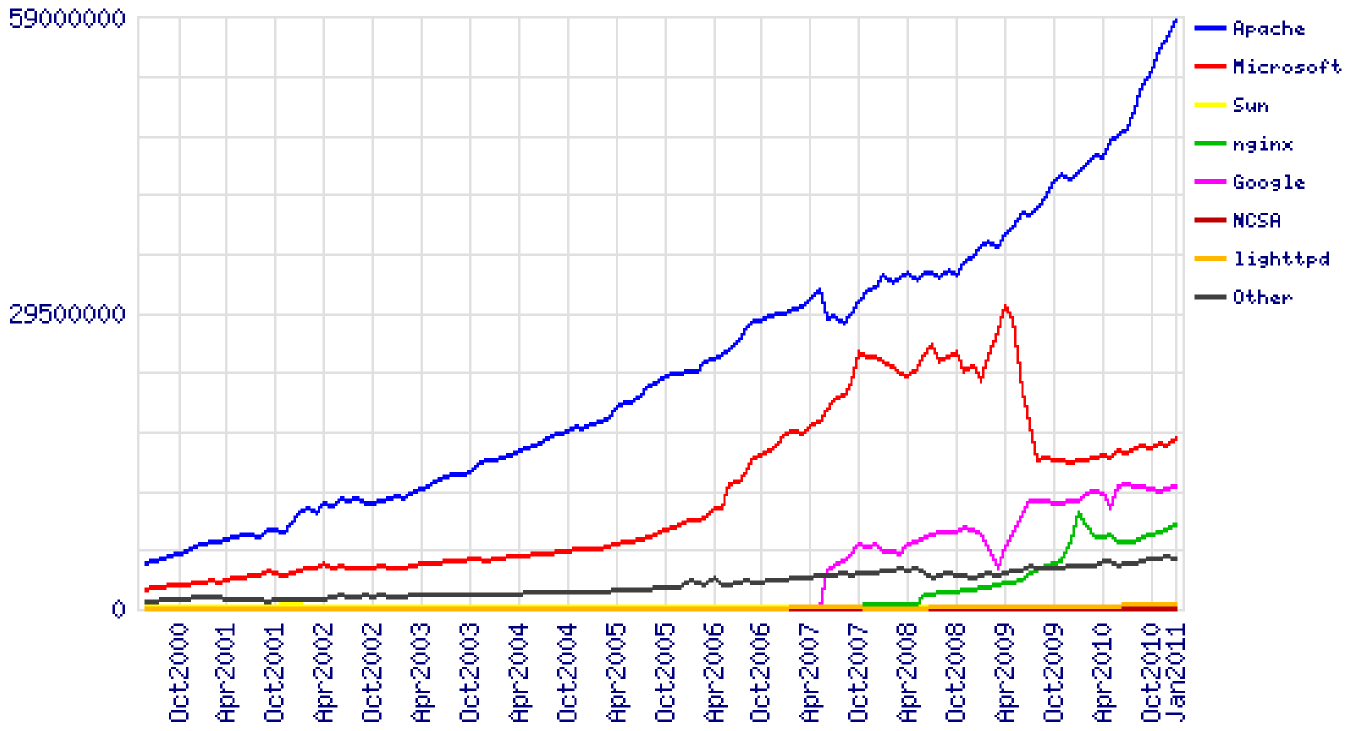
How Apache Came to Be

In February of 1995, the most popular server software
Center for Supercomputing Applications, University
NCSA in mid-1994, and many webmasters had devel
of these webmasters, contacted via private e-mail, gat
Behlendorf and Cliff Skolnick put together a mailing
Bay Area, with bandwidth donated by HotWired. By

For Review:

- What is Apache?
- About Apache

Apache Statistics



Totals for active servers across all domains June 2000 - January 2011

Apache Statistics (Contd)

Developer	December 2010	Percent	January 2011	Percent	Change
Apache	56,810,393	57.38%	58,623,115	57.57%	0.19
Microsoft	16,388,264	16.55%	17,070,240	16.76%	0.21
Google	11,959,840	12.08%	12,115,707	11.90%	-0.18
nginx	7,948,163	8.03%	8,376,958	8.23%	0.20
lighttpd	540,212	0.55%	527,225	0.52%	-0.03

Adoption rates of web server software across all domains December 2010 - January 2011

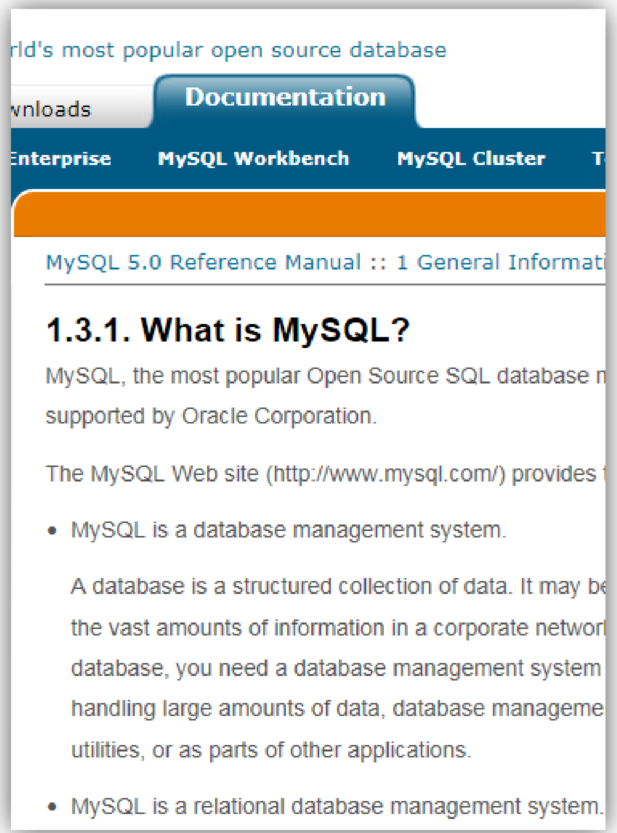
MySQL

MySQL is a robust, open source database solution that enables the web designer to store and retrieve data using PHP scripting. This powerful combination has led to the creation of "dynamic" websites.

A dynamic website is a single page of code that changes based on the viewers' interaction with the browser. Dynamic sites offer numerous combinations of potential end user experiences. The data that can be "manipulated" includes:

- Boolean operators
- Text
- Integers
- Images
- Binary digits

It should be noted that it is not only the MySQL aspect of the programming that creates this unique ability, but the extensive capabilities of PHP scripting and MySQL data management processes combined.



MySQL 5.0 Reference Manual :: 1 General Informati

1.3.1. What is MySQL?

MySQL, the most popular Open Source SQL database m
supported by Oracle Corporation.

The MySQL Web site (http://www.mysql.com/) provides

- MySQL is a database management system.

  A database is a structured collection of data. It may be
  the vast amounts of information in a corporate networ
  database, you need a database management system
  handling large amounts of data, database manageme
  utilities, or as parts of other applications.

- MySQL is a relational database management system.

Required Reading:

- What is MySQL?

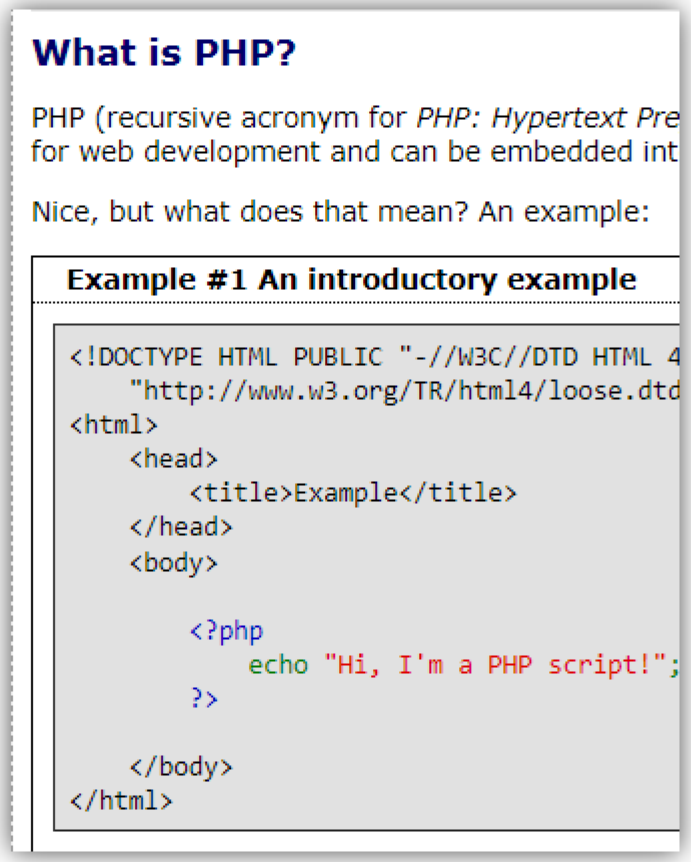
PHP

PHP is a widely used scripting language that can be easily embedded in to HTML code. This easy interaction allows web designers to create dynamic web applications that output different results from different user inputs.

The dynamic interaction afforded by PHP and other platforms gives designers more opportunities to develop creativity and enhancements over traditional HTML static web pages and has resulted in a stampede of web sites designed with PHP scripting.

Required Reading:

- What is PHP?



What is PHP?

PHP (recursive acronym for *PHP: Hypertext Pre*
for web development and can be embedded int

Nice, but what does that mean? An example:

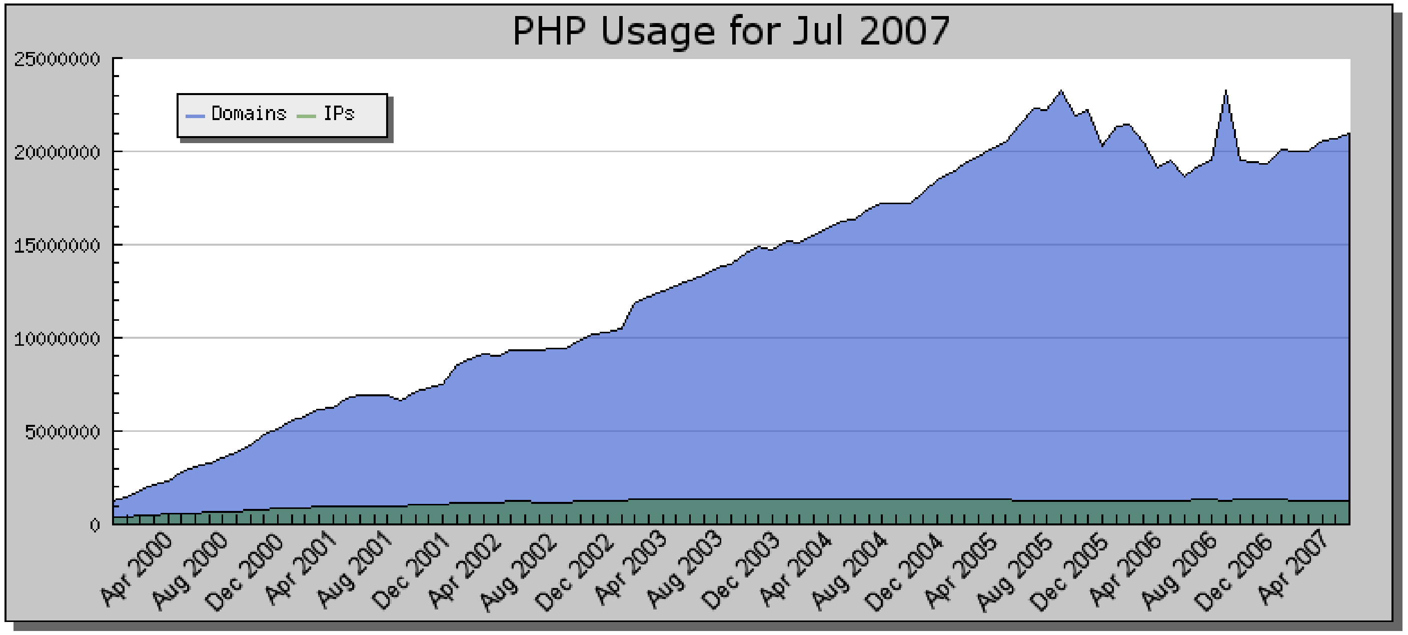
Example #1 An introductory example

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4
    "http://www.w3.org/TR/html4/loose.dtd
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

PHP Use



Usage Stats for April 2007 from www.php.net/usage.php

LAMP Use

Which web server and associated applications you choose depends on your objectives, training, comfort zone and resources (including money). If you critically examine the cost of a web server, it becomes obvious that LAMP presents a favorable argument.

LAMP is free, works with minimal hardware resources, covers most web page requirements from web carts to blogs, and is easily implemented and managed after very little initial training. Furthermore, there are abundant web sites for free technical support as well as numerous sites offering traditional “paid” service and support. Redhat, for example, has made a fortune offering support and customized solutions for web administrators and businesses.

LAMP meets the needs of both large and small web service providers alike. The author of this lesson switched to a LAMP platform after hosting with Microsoft products for over ten years. The switch was long in coming but necessary to compete with larger hosting companies while staying profitable. A Windows-based hosting setup for 50 clients could run as much as $1500 to $2000 a year in licensing fees alone while a LAMP-based solution is free. Additionally, the switch from Microsoft forced me to be more aggressive in learning Linux, which led to new business opportunities and a whole new network of experienced technicians and engineers.

LAMP Compared

From the operating perspective, Linux offers much more security than Microsoft products and is much faster. Linux can be run efficiently on older and cheaper hardware, and uses less memory than similar Microsoft systems.

On the other hand, Microsoft dominates the home market, so new IT technicians tend to favor their products due to an excellent marketing campaign by the Microsoft team and specifically Bill Gates. Mr. Gates has offered the Microsoft product line on home computers for nearly 20 years at a cost of pennies on the dollar. This allowed Microsoft to gain public trust and following in both the home and corporate environments while encouraging businesses to invest on the Microsoft platform because their employees were already familiar with Microsoft products and operating systems.

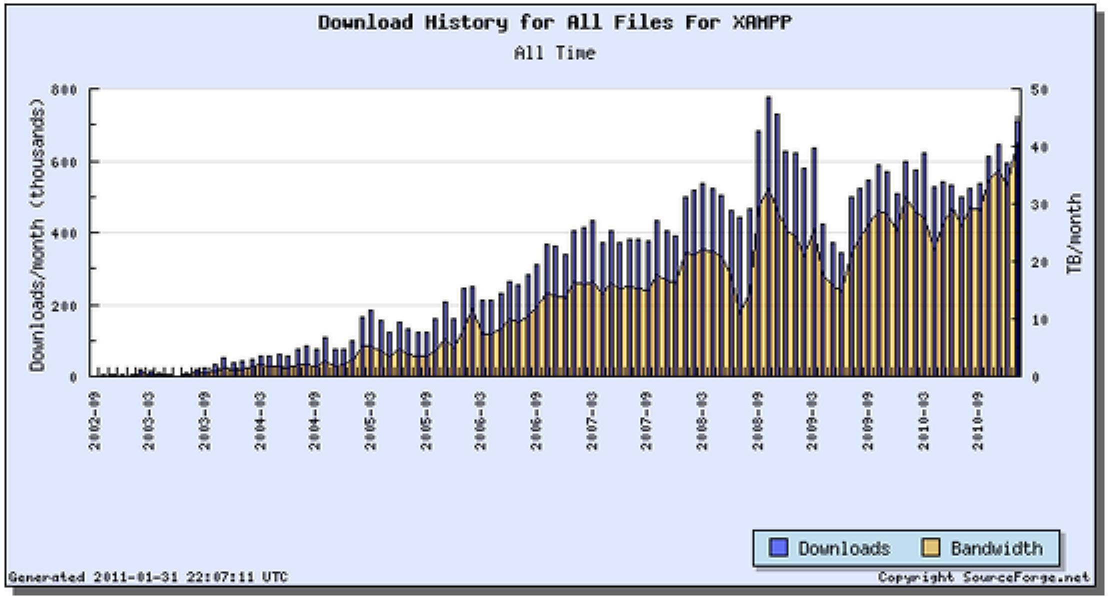
Times are now changing and training programs are being developed to familiarize IT technicians with not just Microsoft but Linux-based systems and applications as well. An Apache-based web hosting system on a Linux server offers faster speeds and security at a reduced cost when compared to Microsoft Internet Information Server (IIS). Additionally, PHP is many times faster than Microsoft or Sun Microsystems' Java platform. PHP also allows for command-line scripting that can lead to customized scheduling of cron jobs on your Linux based system.

Chances are a large portion of web sites you use run on some aspect of LAMP. Nearly 60% of all web sites use Linux as their base operating system. Of those, most use MySQL and PHP for additional functionality.

XAMPP

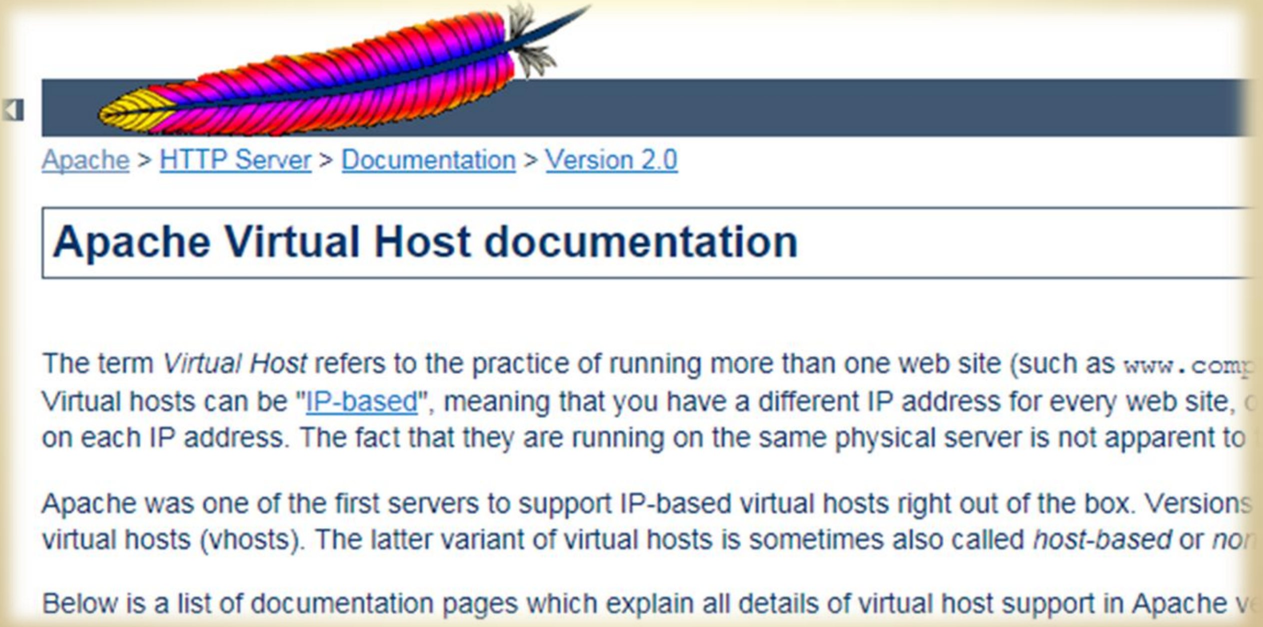
XAMPP is a modernized version of LAMP and is increasing in popularity. Both utilize the Apache, MySQL, PHP attitude and functionality, but XAMPP has added more opportunities and additional security features. XAMPP is gaining market share on LAMP and has passed the 50% threshold but with the increased functionality comes an aggressive learning curve and potential for security vulnerabilities through configuration mistakes.

New administrators are heavily cautioned on using XAMMP and should steer towards LAMP due to the simple and proven nature of the LAMP package.



Graph taken from Apachefriends.org

Virtual Hosting



Apache > HTTP Server > Documentation > Version 2.0

Apache Virtual Host documentation

The term *Virtual Host* refers to the practice of running more than one web site (such as `www.comp`
Virtual hosts can be "IP-based", meaning that you have a different IP address for every web site, c
on each IP address. The fact that they are running on the same physical server is not apparent to

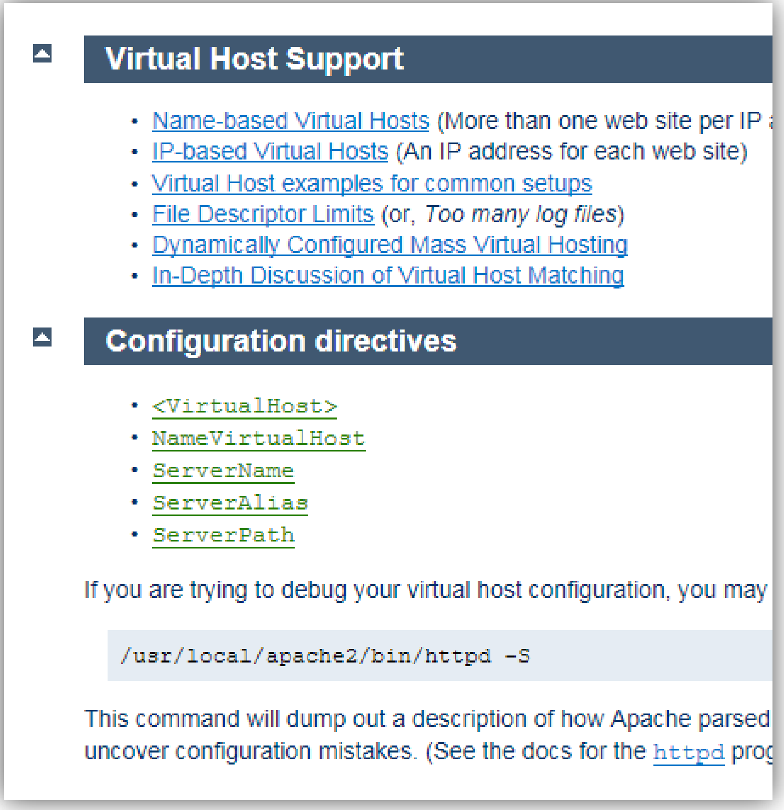
Apache was one of the first servers to support IP-based virtual hosts right out of the box. Versions
virtual hosts (vhosts). The latter variant of virtual hosts is sometimes also called *host-based* or *nor*

Below is a list of documentation pages which explain all details of virtual host support in Apache v

Virtual Hosting

Apache has a magnificent built-in feature called virtual hosts. This feature allows you to host web sites by either IP address or domain name.

Virtual hosting allows system administrators to serve web pages consisting of several domain names through the use of a single server. In fact, most savvy hosting companies do this because it saves money and decreases the amount of resources needed to host sites.



Virtual Host Support

- Name-based Virtual Hosts (More than one web site per IP
- IP-based Virtual Hosts (An IP address for each web site)
- Virtual Host examples for common setups
- File Descriptor Limits (or, *Too many log files*)
- Dynamically Configured Mass Virtual Hosting
- In-Depth Discussion of Virtual Host Matching

Configuration directives

- `<VirtualHost>`
- `NameVirtualHost`
- `ServerName`
- `ServerAlias`
- `ServerPath`

If you are trying to debug your virtual host configuration, you may

```
/usr/local/apache2/bin/httpd -S
```

This command will dump out a description of how Apache parsed uncover configuration mistakes. (See the docs for the `httpd` prog

Virtual Hosting on LAMP

There are several options for setting up and installing a LAMP server. There is even a boot CD distribution of Debian called Lamppix that will auto-boot and can be easily installed on an older computer with minimal memory and processing power.

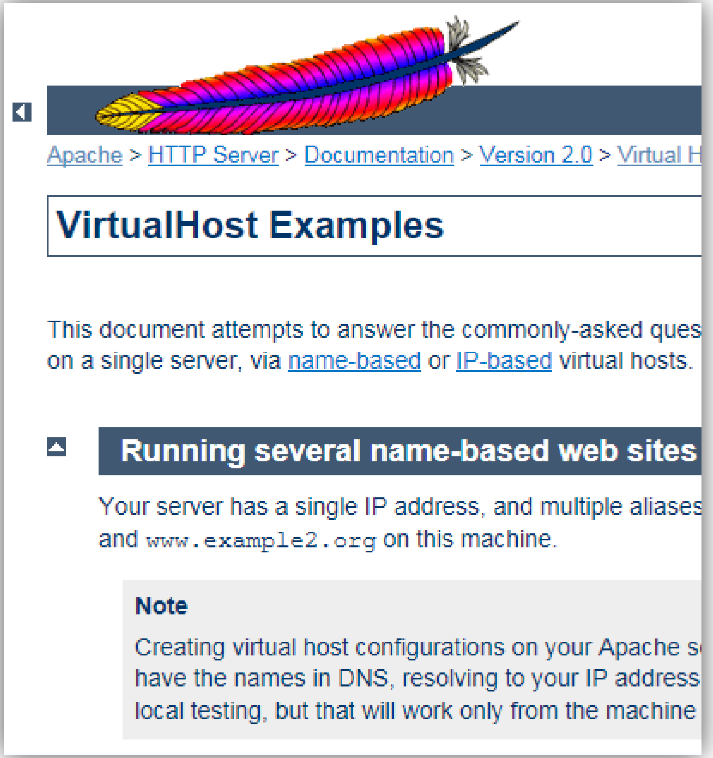
Install Options

- Lamppix Boot Cd
- Debian net installation
- Ubuntu Server installation boot cd
- Regular Ubuntu Desktop installation followed by additions of Apache, MySQL and PHP

Each virtual install option has advantages and disadvantages that will be discussed on the next slide.

Required Reading:

• Lamppix



Apache > HTTP Server > Documentation > Version 2.0 > Virtual H

VirtualHost Examples

This document attempts to answer the commonly-asked ques
on a single server, via name-based or IP-based virtual hosts.

Running several name-based web sites

Your server has a single IP address, and multiple aliases
and `www.example2.org` on this machine.

Note

Creating virtual host configurations on your Apache s
have the names in DNS, resolving to your IP address
local testing, but that will work only from the machine

Virtual Hosting Install Options

Lamppix

Lamppix is designed to run directly from the original boot CD. It also has an option for installation with minimal administrator interaction. The second install option is a better choice that also works well on older computers with minimal resources. If you choose to run Lamppix directly from the boot cd (live CD mode that can be used for testing), configuration can be tricky or confusing and is not recommended for real time or virtual hosting. However, the live CD mode may be useful in a small test environment or in a home network.

Debian

A fresh Debian install is probably your best option. Keep your installations small and add additional the components through the command line using Apt-get. Gnome could be added, but it will consume more resources. Try to stick with minimal installation if possible for improved system performance. The initial Debian installation process will include options for installing DNS, LAMP, SAMBA and other services with minimal administrator interaction. The best choice is to select LAMP to minimize resource usage. Once selected, Debian will auto load and install all the components required for a functional LAMP system. The installation process will pause when necessary to allow input such as user IDs, passwords and other configuration options. A complete setup in this manner including updating can be performed in as little as one hour!

Virtual Hosting Install Options (Contd)

Ubuntu Server Installation

Ubuntu Server installation is similar to a Debian installation in which the core components are installed to produce a functional server. You will use the Ubuntu server installation method for the labs in this lesson. Keep in mind that you have to be careful which services you install as part of the default installation. Many of the default options are not required. A video and PowerPoint presentation will be provided to demonstrate the selection process.

Ubuntu Desktop Installation

Ubuntu desktop installation is a regular install of the desktop operating system followed by additional installation of LAMP server components. While this option works, it should be used as a very last resort because the desktop installation adds many applications and services that are not required. The additional components also demand extra maintenance and management time.

The only advantages of using the desktop install is that it is “dummy proof” and provides a GUI-based desktop application.

Virtual Hosting Configuration

Setting up DNS

To set up your DNS, edit your computer hosts file to trick your system into resolving a fake name for your server's IP address. On Linux, the HOSTS file is located in /etc/hosts, and you will need to add the following lines to it, assuming your server's IP address is 10.0.2.100.

```
10.0.2.100 testdomain.com www.testdomain.com
10.0.2.100 testotherdomain.com www.testotherdomain.com
```

NOTE: The IP addresses used above are fictional and will not work on a real server. You *must* substitute *10.0.2.100* with your real IP address to create a fully functional server!

After adding the lines above to your HOSTS file, your system will be tricked into resolving the domain names testdomain.com, www.testdomain.com, testotherdomain.com, and www.testotherdomain.com to the IP address of the server.

TIP: Keep this HOSTS file trick in mind the next time you register a domain name. You can set up your HOSTS files so you can start building your website while you wait for DNS to register and resolve your domain name.

Virtual Hosting Configuration: DNS

What exactly is DNS?

DNS is short for Doman Name System. DNS is a database system that translates a domain name such as www.google.com into a matching or associated IP address such as 74.125.47.147. The domain name or IP address leads to the same address.

In the old days of the Internet, administrators maintained DNS information by sending their daily additions to one centralized person who added the new entries to a file which pointed future requests to the IP address that hosted the web pages. As the web grew, a system was developed that allowed these changes to be configured on the fly. As a new domain name is issued or changed, the information is populated to root servers which translate the domain name information to an IP address. The request for the web page information is then sent to the hosting server.

Can you imagine how complicated the manual process for maintaining the “translation” information must have been in the early days of the Internet?

Required Reading:
- What is DNS?

Preparing for Virtual Hosting (Step 1)

Before you edit configuration files, you need to prepare the system for virtual hosting by creating the *DocumentRoot* and *Logging* directories. A rule of thumb that usually works well is to use the domain name for the parent directory. You can store your *DocumentRoot* directories wherever you wish, but the /home directory works well. Follow these steps:

Step 1. Create the directories:

```
mkdir -p /home/www.testdomain.com/public_html/cgi-bin
mkdir -p /home/www.testdomain.com/logs
mkdir -p /home/www.testotherdomain.com/public_html/cgi-bin
mkdir -p /home/www.testotherdomain.com/logs
```

NOTE: The traditional method of using cgi-bin directories is to place them outside the document root and use the cgi-bin ScriptAlias and Alias directives. However, this method works and can be used if you accept there might be minimal risks involved.

Preparing for Virtual Hosting (Step 2)

Step 2. Change Permissions

Now that the directories are created, you should change the permissions of these directories because you are probably running as root.

Changing permissions prevents users from logging in to the server as root with permission to edit files and manage websites.

Change the permissions as follows:

```
chown -R someuser.somegroup /home/www.testdomain.com/
chown -R someuser.somegroup /home/www.testotherdomain.com/
```

Preparing for Virtual Hosting (Steps 3-4)

Step 3: Create directory to store virtual hosts

You need to create a directory that will store your virtual host files. You may simply append the virtual host configurations directly to the httpd.conf file. However, you can also have Apache include a directory of configuration files when it starts. Keeping a separate directory of individual virtual host files seems more practical than editing a configuration file of 500+ lines each time you need to manage one of those hosts. Make the directory now:

```
mkdir /www/conf/vhosts
```

Step 4: Modify the httpd.conf file

Modify the httpd.conf file to include this new directory of configuration files you will be creating. Open the /www/conf/httpd.conf file and add the following line to the end of it:

```
include conf/vhosts
```

Preparing for Virtual Hosting: Step 5

Step 5: Edit "NameVirtualHost"

Because you are editing the httpd.conf file, you need to enable an additional virtual hosts setting.

Locate the NameVirtualHost directive line and remove the comment symbol (#). Change the line to the following setting:

```
NameVirtualHost *
```

The previous setting enables virtual hosts to be configured for any IP address. You should be all set to create your virtual host configurations now.

Virtual Host Configuration Files

Creating separate files for each host makes virtual hosts much easier to manage. Let's set up the first file for *www.testdomain.com* and name this file *www.testdomain.com.conf* in the */www/conf/vhosts* directory.

The file will contain the following contents (if it does not, modify it to match):

```
<VirtualHost *>
ServerName www.testdomain.com
ServerAlias testdomain.com
DocumentRoot /home/www.testdomain.com/public_html
CustomLog /home/www.testdomain.com/logs/access_log combined
ErrorLog /home/www.testdomain.com/logs/error_log
</VirtualHost>
```

Each line above will be described in detail on the next few pages.

NOTE: When using the backslash to continue directives, you should avoid allowing any additional spaces or characters after the backslash (\), which could cause parse errors in the configuration files.

Description of Configuration Files

Let’s break down this configuration file for better understanding.

<VirtualHost *>

<VirtualHost *> is the opening tag for the virtual host. Everything between this line and the last line, </VirtualHost>, will contain the settings for the virtual host you are configuring. Take special notice of the asterisk (*) in the open tag. The asterisk (*) indicates that this virtual host will respond to any request for this domain name on any IP address on which it is used.

ServerName www.testdomain.com

ServerName is the name of your server, hence the name ServerName.

ServerAlias testdomain.com

The ServerAlias directive enables you to point testdomain.com to the same virtual host as www.testdomain.com. When the server alias is configured properly, a person who enters testdomain.com or www.testdomain.com will access the same site. As such, the server alias is important because web users may access your site without using www before the domain name. If you do not configure the server alias properly, you will lose users who do not access your site using the standard www.testdomain.com.

Description of Config Files (Contd)

DocumentRoot /home/www.testdomain.com/public_html

The DocumentRoot directive is the path to the directory where the website's files are located.

CustomLog /home/www.testdomain.com/logs/access_log \combined

The CustomLog directive is the path to the log file that will contain access information about the virtual host.

ErrorLog /home/www.testdomain.com/logs/error_log

The ErrorLog directive contains error information regarding the domain

</VirtualHost>

Finally, you close the Virtual Host configuration for this domain name by using the closing tag </VirtualHost>.

Virtual Host Configuration (Version 2)

Now that you have a better understanding of the virtual host configuration, let's make another virtual host for your other domain name.

Create a file at www/conf/www.testotherdomain.com.conf and enter the following information:

```
<VirtualHost *>
ServerName www.testotherdomain.com
ServerAlias testotherdomain.com
DocumentRoot /home/www.testotherdomain.com/public_html
CustomLog /home/www.testotherdomain.com/logs/access_log\combined
ErrorLog /home/www.testotherdomain.com/logs/error_log
</VirtualHost>
```

Create Index File

Your next task is to create an index.html file that will display a message indicating which virtual host is being displayed when you access the domain name.

In your /home/www.testdomain.com/public_html directory, create a file named index.html with the following contents:

```
 <html>
<head><title>Testdomain.com</title></head>
<body>Welcome to Testdomain.com</body>
</html>
```

In your /home/www.testotherdomain.com/public_html directory, create a file named index.html with the following contents:

```
<html>
<head><title>TestotherDomain.com</title></head>
<body>Welcome to TestotherDomain.com</body>
</html>
```

Testing Configuration Changes

Adding your virtual hosts has been the first real editing you have done with your configuration files. Consequently, you need to test your changes by running a few commands that can be performed without taking your server offline.

The command “apachectl” contains a special setting that enables you to test your configuration files while the server is still running and prevent you from taking your server offline.

Enter this command (in bold) and check the output:

```
apachectl configtest
```

Good syntax output:

```
Processing config directory: /www/conf/vhosts/
Processing config file: /www/conf/vhosts/www.testdomain.com.conf
Syntax OK
```

In this example, everything parsed as expected by the server, which indicates you are clear to start, stop, or restart your server as needed to make the changes take effect. See the next slide for an example of bad syntax output.

Testing Config Changes (Contd)

Bad syntax output:

```
Processing config directory: /www/conf/vhosts/
Processing config file: /www/conf/vhosts/www.testdomain.com.conf
Syntax error on line 2 of /www/conf/vhosts/www.testdomain.com.conf:
Invalid command 'brknmchn', perhaps mis-spelled or defined by a module not included in the server configuration
```

In the example above, you have a bad token on line 2 of your www.testdomain.com.conf file. (We purposely entered the text *brknmchn* because we knew that it would cause a syntax error in Apache.)

After you get the "Syntax OK" output from your configuration files, you are clear to restart the server. We prefer to completely stop the server and then start it instead of using the restart command. Changes do not always take effect using the restart command. A full restart is often required. See below:

```
apachectl stop
apachectl start
```

Alternatively, if you are using SSL:

```
apachectl stop
apachectl startssl
```

Apache Config Files

On Ubuntu Server, the Apache server uses a few configuration files that define its operations. These files are in the /etc/apache2 directory. Let’s start with a short overview:

/etc/apache2/apache2.conf
This is the main configuration file for your Apache server. It contains the generic configuration for your server, such as the specification of the directory where the server can find its configuration files (the so-called ServerRoot) and much more. If you want to tune the performance of your Apache server, this is the file you should look examine. “Include” directives are used in this file to include all other configuration files.

/etc/apache2/httpd.conf
Httpd.conf is empty by default. If you want to create additional configuration parameters that are not in the apache2.conf file, put them here.

/etc/apache2/envvars
Place environment variables to tune the operation of your Apache server in this file.

/etc/apache2/ports.conf
This file contains the port numbers on which the Apache server will listen. By default, the server listens on just port 80.

Apache Config Files (Contd)

/etc/apache2/conf.d/
You can put additional Apache configuration files in this directory. After a default installation, the directory contains only the charset file, which specifies the character set to use. Additional files can be placed here as well. From /etc/apache2/apache2.conf, all files in this directory are included in the Apache configuration automatically.

/etc/apache2/mods-available/
As stated before, you can extend the functionality of your Apache web server by using modules. In mods-available directory, you will find all the modules that are installed for your server. These are just the available modules; not all of them are necessarily used by default.

/etc/apache2/mods-enabled/
To enable a given module, you must create a symbolic link in this directory that refers to the module file in /etc/apache2/mods-available.

So, if in /etc/apache2/mods-available you have a module by the name ldap.load and you want to include it in your Apache configuration, create a symbolic link. Use:

```
cd /etc/apache2/mods-enabled
ln -s ../mods-available/ldap.load ldap.load.
```

This action loads the module automatically the next time you restart your Apache web server.

Additional Resources

LAMP Links

- Building a LAMP Server
- Easy LAMP Server Installation
- LAMP on CentOS
- LAMP on Ubuntu
- Apache MySQL PHP

XAMPP Links

- XAMPP for Linux
- XAMPP

PHP Links

- PHP
- PHP Tutorial

VirtualBox Manual

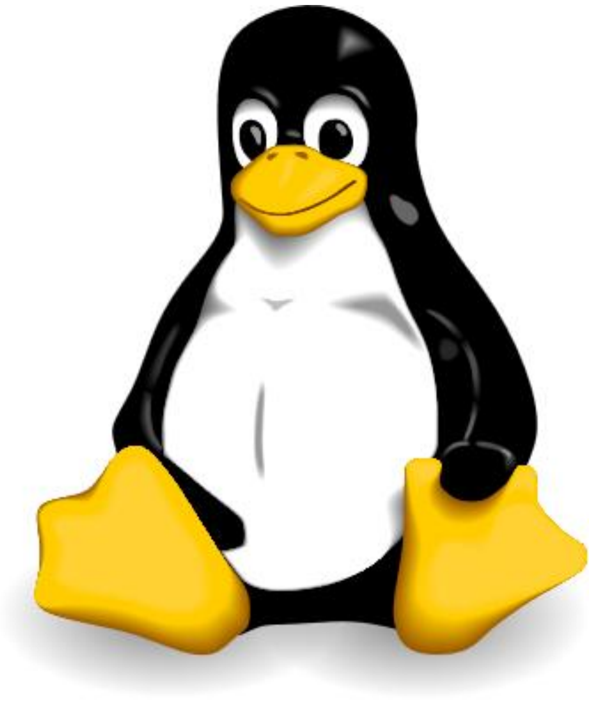


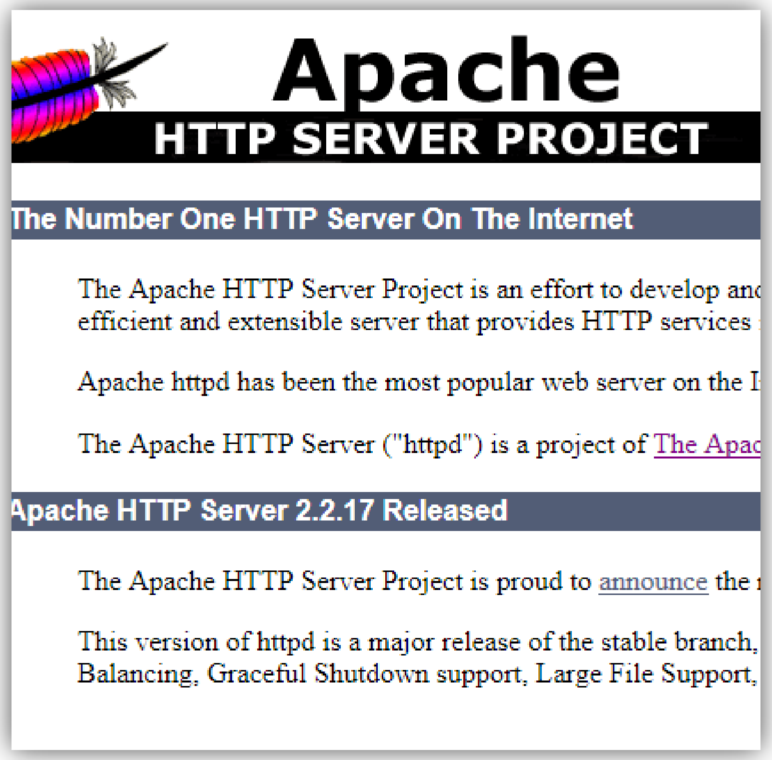
Image Credit: Larry Ewing, Simon Budig, Anja Gerwinski

Lesson Summary

A virtual hosting environment riding on top of a LAMP foundation creates an affordable and easy-to-manage web service.

With over half the world's servers using Apache, the additional services provided by MySQL and PHP create a functional equivalent to licensed commercial hosting services such as Microsoft IIS.

Technicians who are able to master the LAMP web solution are certain to obtain employment in a secure work environment.


Apache
HTTP SERVER PROJECT

The Number One HTTP Server On The Internet

The Apache HTTP Server Project is an effort to develop anc
efficient and extensible server that provides HTTP services

Apache httpd has been the most popular web server on the I

The Apache HTTP Server ("httpd") is a project of The Apac

Apache HTTP Server 2.2.17 Released

The Apache HTTP Server Project is proud to announce the

This version of httpd is a major release of the stable branch,
Balancing, Graceful Shutdown support, Large File Support,