Apache Web Server Setup

Introduction:

What is the Apache Software Foundation? They are a non-profit corporation, an extension of The Apache Group, that in 1995 collaborated to develop the Apache HTTP Server. The foundation was created to supply hardware, communication, and business infrastructure to collaborative software development projects. To provide legal means for the individual volunteers and to protect the Apache brand name from misuse.

The management of the Foundation is overseen by an annually elected board of directors. The board appoints managers to oversee the foundations operations and projects. Projects are managed by technical experts that actively contribute to their projects.

The latest August 2002 Netcraft Web Server Survey found that the Apache Web Server (AWS) is still the most popular HTTP server on the internet. A total of 63% of web sites out there on the web are using Apache. Microsoft, with their Internet Information Server (IIS) are trailing way in the back with just over 30% Another interesting note is that Microsoft have been actually losing ground lately.

Why would an individual use the AWS?

A webserver can be a great solution as an interface to content or resources sitting on a LAN for any sized network: a networked home up to a corporate sized company. You can use widely available html editors to whip up the web pages in no time and and direct users to content. From personal experience a well organized web interface requires zero training to your users, which is always a major plus. As you will see setting up restrictions access rights for the various users is a breeze. Regarding access from the outside into your intranet, one has to be very cautious and take security issues well into consideration before doing so.

Why is AWS so popular?

For one, it's absolutely free, whereas Microsoft's IIS Windows 2000 Server with 5 Client Access Licenses costs around 1000usd.The source code is open and thoroughly tested by unbiased (well...) developers and users. All the latest protocols are implemented including HTTP1.1 etc. Multiplatforms are supported (Win32 and the various UNIX platforms such as Linux, Solaris and FreeBSD). Security concerns/vulnerabilities are not downplayed or blatantly ignored.

More information about the Apache Web Server can be found on the Apache Web Server website. The name "Apache" was said to have come from the phrase "A Patchy Server", at least they are comfortable with the inevitable patching practice!

What we will be looking at in this article is how to get up and running with your own Apache Web Server on a Win32 platform within 10 minutes of downloading the software. No prior Web Server knowledge is required, but working knowledge of regular file management and editing text configuration files will be useful. Since you are over reading this at MHW, I am sure you will find this whole procedure a breeze :-)

Prerequisites:

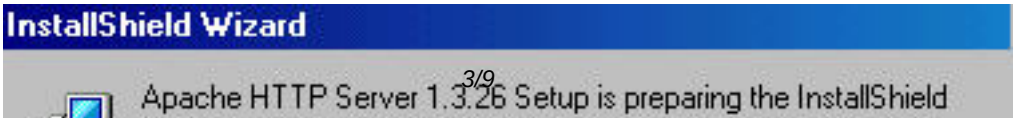
- Windows operating system(Win95 - make sure you read this.)
- Web content (your index web page...or we can use the built in test HTML page)
- firewall restrictions: Make sure you exclude your Apache server ports, usually port 80, from the list of ports your firewall is blocking.
- Celebration beverage of your choice. (Editor's note; Mountain Dew of course.)

Step 1: Download the Software

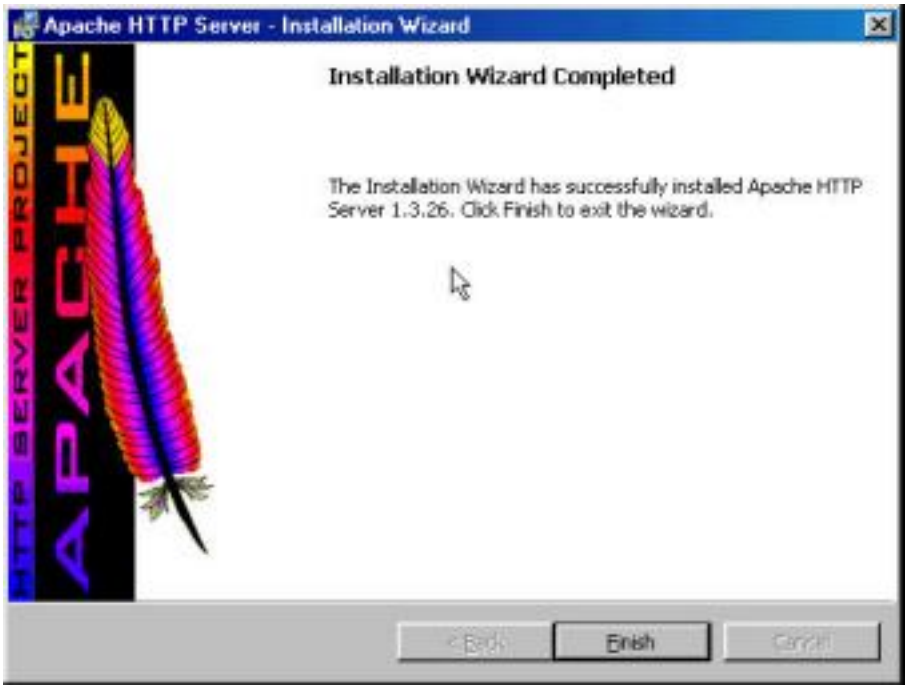
We have two options: Download only the Web server source code or download the source code and a pre built binary. I opt for the Pre built source code 1.3.26 (a new version 2.0.40 is available), so go here and download from a http or ftp site of your choice the following 5.1MB installer package: apache_1.3.26-win32-x86-no_src.exe We will not be bothering with any accompanying signature files which can be used to verify that the packages are authentic Apache.Org releases. But feel free to do so if inclined, just note you may need additional software to read these files.

Step 2: Install the Apache Server

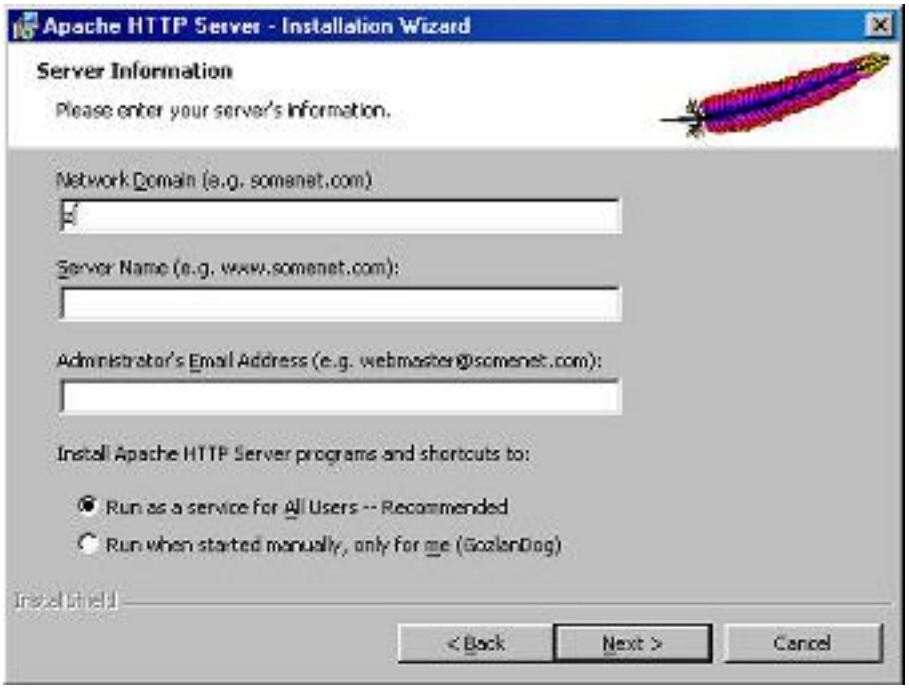
Note: If you have a previous installation of AWS then now is a very good time to shut down the previous server and uninstall it through the Windows 'Add/Remove Programs'. Remember - Configuration settings will remain!!! Double click the apache_1.3.26-win32-x86-no_src.exe file you previously downloaded. The server software will be installed to the following default directories:


InstallShield Wizard
Apache HTTP Server 1.3.26 Setup is preparing the InstallShield

C:Program FilesApache GroupApachesrc
C:Program FilesApache GroupApache
C:Program FilesApache GroupApachehtdocs



Unless you really have good reasons to - I suggest not to change these defaults yet, for two reasons: Firstly if you later re-install AWS in the same default location, the new server installation will use the settings you had in the previous installation and secondly, troubleshooting will be obviously easier. Now the quick installation process starts. Agree to the license and read the security warnings regarding Win32 platforms. Now you will be greeted by a screen requesting your server's information.



Network Domain - Fill in your company or private network domain (mycompany.com), if you do not have a domain, or unsure I suggest to fill in "Network_Domain". This can be edited later on if need be.

Server Name - Here you have to put your registered DNS name. If you have a dynamic IP then enter it here.

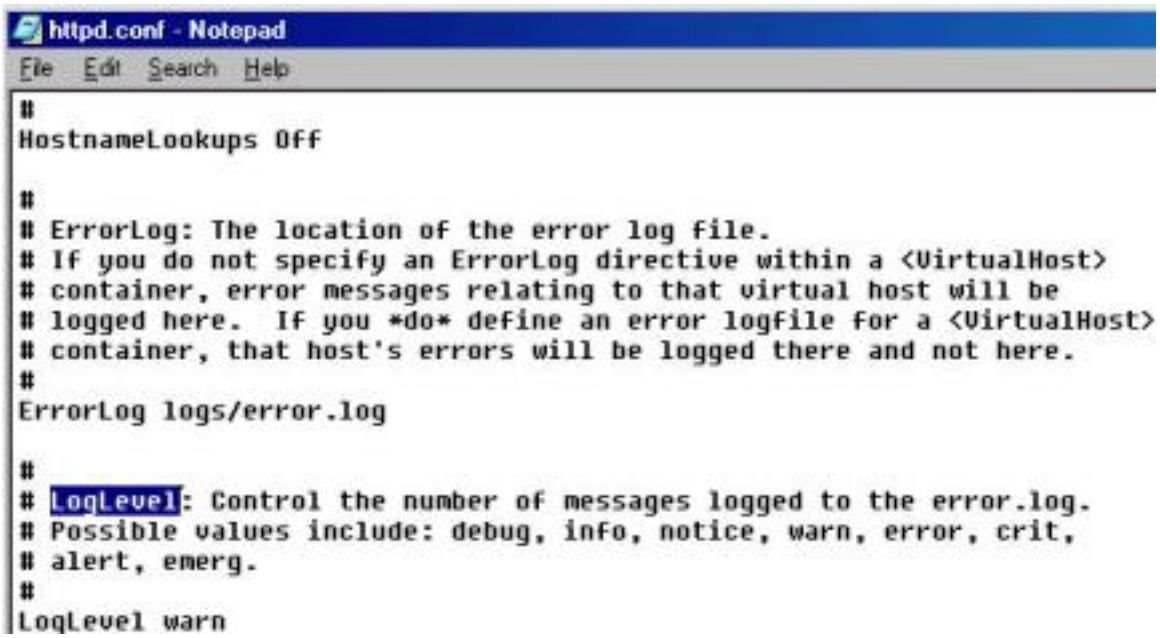
Network Administrator's Email - This is used for feedback from disgruntled users if they receive server error messages: They will be displayed with the option to report the error to this email address. Go ahead and fill in an appropriate email address.

Install Apache HTTP Server programs and shortcuts to: Here you will have two options. choose what suits your computer setup regarding users etc:
-Run as service for all users.OR
-Run when started manually, only for me.

The next screen asks you which setup you would like to perform: "Complete" or "Custom". The Custom option will allow you to change the previously mentioned default directories and also save some space (21MB for complete installation is required) by waiving the documentation files. Lets go with the "Complete" installation. Click 'Next' twice and you will see the installation process zip by and you will be presented with the "Installation Wizard Complete" screen. Hit 'Finish' to complete the process.

Step 3: Verify Configuration Settings:

The first item we are going to verify is the Server name. With Notepad or any other text file editor open up the AWS main configuration file, the httpd.conf file. Perform a search for "ServerName" (the 2nd one found from the top of the file) and make sure you have a valid name or IP address after the ServerName (you should, because you just entered one during the Installation Wizard process!). Note that AWS will not start up unless the Server Name is valid. You may use 127.0.0.1 or "localhost" for testing purposes. **Important Note:** Apache config files require the use of UNIX style slashes: Use "C:/mydocs" as opposed to "C:mydocs"



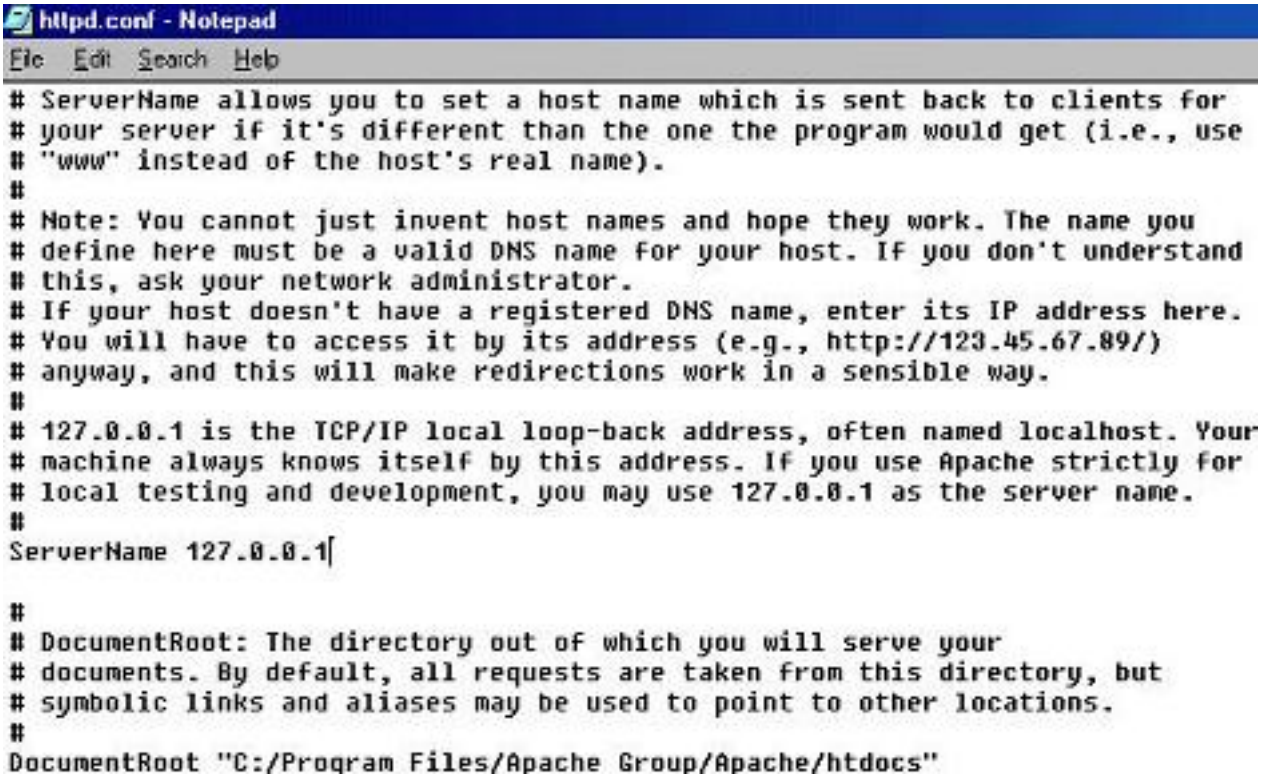
The next thing we are going to do is start up the server. Go to Start->Programs->Apache Web Server->Start Apache When the server is started (in Win 9x you will be running a DOS console) open up your web browser and type in the address HTTP://ServerName (use the Server Name you designated in step 1) or use HTTP://127.0.0.1

You should now see the 'Apache has been successfully installed!' webpage (this page is the index.html.xx found in the C:/Program Files/Apache Group/Apache/htdocs directory). It is recommended that you get rid this page ASAP - Delete it or save it in a different directory. If this page is served to one of your users they might think they had just installed AWS on their machine and that may cause unwanted helpdesk confusions...


t the installation of the Apache web server software on this system was successful.
page.

The default page that AWS will serve to a user logging on to your HTTP server will be the index.html or index.htm file. Now, if you did not see the 'Apache has been successfully installed!' webpage, note what the error screen says and head over to the Apache FAQ and you will most likely find out the reason for the error.

Another configuration setting you might be interested in is changing the default directory for your web content. The default is "C:/Program Files/Apache Group/Apache/htdocs" Lets say you want to have all your web pages in the folder C:mywebpages. Open up the httpd.conf file again and search for the string "DocumentRoot". Change the default entry to "C:/mywebpages" There are two places within the httpd.conf file you will need to edit thedefault "DocumentRoot" entry.



Important Note:
After making changes to the AWS config files (such as httpd.conf) you will need to restart the server to have the changes take effect. Do so through a DOS command line: C:Program FilesApache GroupAPACHE>apache -k restart

Debugging:
You can change the "LogLevel" in the httpd.conf file from "warn" to "debug". All error information will be then logged into the err.log file. Easy access to this file is provided via the Start menu: Start ->Program Files -> Apache HTTP Server -> Review server logs files.

Fine tuning:
Most settings are good enough to start off with, but I suggest you make sure the following entries in the httpd.conf file are correct:
- KeepAlive - It should be set to "on". This helps speed up execution.
- TypeOfServer - make sure it is set to "Standalone". This also will speed up execution times.
- Options Multiviews - Default is "off". If you are not serving multiple languages then leave it off. this will reduce negotiation traffic.
- TimeOut - Default is 300 seconds. This is the timeframe the server will wait for your server to send a packet and the client to return an acknowledge. You can safely change this to 75 sec. 300 seconds is way to much.

Directives:
The Apache Org documentation is very helpful for new AWS users uncertain that their server configurations are optimal or even correct. A good resource for these cases can be found at http://httpd.apache.org/docs/mod/directives.html. You can find a listing of available directives (configurations) with detailed explanations and examples.

Step 4: Restricting Access

AWS has two main methods to control access on a per-directory level:
- containers in the server-wide configuration files.
- .htaccess files in directories where they are required.

Each method has its advantages and its disadvantages.

.htacces files are used to allow the server to handle all the documents in a particular directory, or directory tree in the same manner, for example- requiring a password before access is given to files in a particular directory or prohibiting directory listings altogether.

How is this done? AWS configuration groups documents by directory. To apply controls/restrictions to a certain directory tree you can use the container directive in the server's config files. An example:

AllowOverride None
Options None

Note: AllowOverride None disables .htaccess files.

The server will search for .htaccess files, starting at "C:" then all the way down, through all the directories until it reaches the one containing the requested document, processing all .htaccess files found that the config files say should be processed. Why are their so many .htaccess files? One of the reasons is so you don't need to restart the server after making a change. This means you can grant or withdraw access on the fly.

Due to difficulties in persuading certain OSes to allow you to create or edit file names beginning with a "." you can configure AWS to look for files named ht.acl instead of .htaccess by changing the "AccessFileName" httpd.conf entry to: AccessFileName ht.acl

Final Words:

If this was your first attempt at a web server installation hopefully something did go wrong and hopefully you resolved the issue and gained valuable experience. Although we did not cover any really 'advanced' Apache Web Server configurations or options such as securing web pages, cgi or Perl scripting, SSI or configuring various "aliases". But there are plenty of resources for these advanced options on the web. Some good sites I can recommend are: The Scripts, Apache Week, Dot.Com Builder.

Gozlandog

This page comes from
Monster-Hardware:

http://www.monster-hardware.com

The URL for this page is:
http://www.monster-hardware.com/modules.php?name=Content&pa=showpage&pid=17