Microsoft®

Apache 2.2 on Windows: A Primer

Published by the Open Source Software Lab at Microsoft. February 2008.
Special thanks to Chris Travers, Contributing Author to the Open Source Software Lab. Most current version will be maintained at http://port25.technet.com.

PORT25

Abstract:
By many estimates, Apache is the world's most popular web server software, hosting more than half of active domains according to Netcraft. Typically, Apache is run on Linux or UNIX, but it runs quite well on Windows. This paper provides an introduction to running this software on Windows and provides a framework for understanding how Apache on Windows is fundamentally different from Apache on Linux.

1.1 Introduction

By many estimates, Apache is the world's most popular web server software, hosting more than half of active domains according to Netcraft. Typically, Apache is run on Linux or UNIX, but it runs quite well on Windows. This paper provides an introduction to running this software on Windows and provides a framework for understanding how Apache on Windows is fundamentally different from Apache on Linux.

This paper is written for users with general familiarity with the Windows operating system including some experience administrating Windows servers. I do not assume, however, any prior experience with Apache.

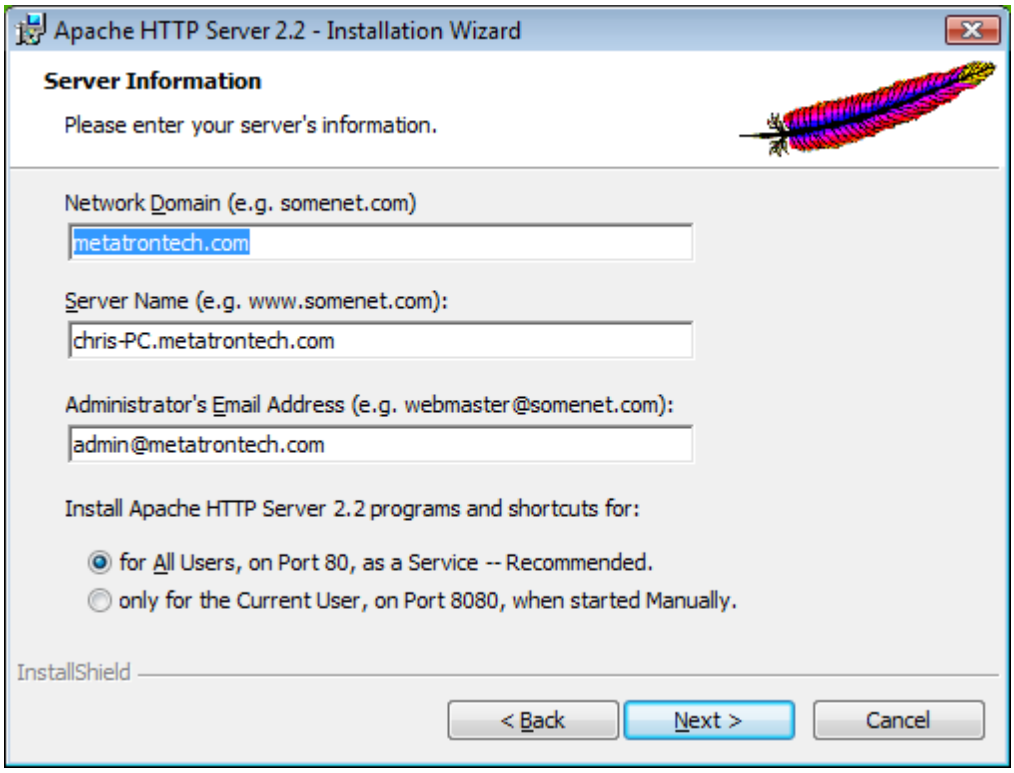
1.1.1 Windows-specific Notes

Since 2.0, Apache has used a modular structure aimed at allowing for better performance across multiple platforms. When the software is compiled, a Multi-Process Module (MPM) is selected. By default on UNIX and Linux, the process-based "Prefork" MPM is selected, while on Windows the default MPM uses the Windows preferred thread-based model. For many applications, this change results in a sizable performance gain in Windows compared to older versions of the software, but problems can occur if a module is used which is not thread-safe. In these cases, the application must run as a CGI script and this inflicts greater performance costs than the old architecture. Always check the documentation of any modules not distributed with Apache before installing into the web server process.

1.2 Obtaining and Installing Apache 2.2

Apache can be obtained from http://httpd.apache.org/download.cgi.

The installation wizard is fairly simple and asks for very little user input. The following screen is an exception and requires some discussion:



Please note that one cannot simply make up names and expect them to work. If in doubt, you can enter "localdomain" to the network domain section and "localhost" to the server name section. The administrator email should be valid.

Vista users should note that User Account Control must be turned off for the software to install properly. I also found that the Apache Server Monitor did not work as advertised on Vista as of 2.2.4. This is not a big problem because the service can be checked through the Services snap-in in the MMC. Another way to access the same functionality in Windows is to right-click on "Computer" and select "Manage."

Of course, running a computer for an extended period of time with UAC turned off is not recommended. If you are running on Vista, you will want to turn it back on after the install is finished and the httpd.conf is properly configured.

Once the software is installed and running, you should be able to go to "http://localhost/" and get a page which simply says, "It Works!"

1.3 Introducing the httpd.conf

Apache, like most UNIX-based software, is configured using a series of text files. As of current versions, the primary file is httpd.conf, though other files may be included into this one. All of the configuration options in for the server can be specified here. The syntax is similar to XML but has its own format.

The actual file is well commented but these comments are intended to serve more as notes than as instructions. For simple deployments the default configuration may be sufficient.

The file can be found in the Start menu, near other Apache files. Clicking on it in the start menu opens it. However, this does not work properly with Vista when UAC is active.

1.3.1 Loadable Modules

Apache, by itself, is a fairly simple web server with very little functionality. Modules provide a means of extending that functionality without making the software "everything to everybody." Modules can provide means of authentication (including single sign-on or single password integration with Active Directory), server-side includes, server-side scripting languages, and the like. Modules distributed with Apache are all thread-safe and can be used on Windows without problem, but additional modules may pose problems for the threaded back-end. Always check your documentation before loading such modules on Windows.

1.3.2 Useful Main Directives

Although the default installation is usable, there are a number of main directives which can be particularly helpful. These include:

DirectoryIndex specifies which files to serve by default from a directory.
Alias allows you to create a virtual directory
ScriptAlias allows you to create a virtual directory with suitable permissions for running CGI scripts.
DefaultType allows you to specify a default MIME type for files of unknown type. This can help browsers handle commonly used content correctly.
include allows you to include other files into the main configuration file.

Note that where file paths are used, you must use forward slashes (/) instead of backslashes (\).

Comments are noted by the # sign.

1.4 Sample httpd.conf

```
#
# This is the main Apache HTTP server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See <URL:http://httpd.apache.org/docs/2.2/> for detailed information.
# In particular, see
# <URL:http://httpd.apache.org/docs/2.2/mod/directives.html>
# for a discussion of each configuration directive.
#
# Do NOT simply read the instructions in here without understanding
# what they do.  They're here only as hints or reminders.  If you are unsure
# consult the online docs. You have been warned.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/" for Win32), the
# server will use that explicit path.  If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "C:/Program Files/Apache Software Foundation/Apache2.2" will be
interpreted by the
# server as "C:/Program Files/Apache Software Foundation/Apache2.2/logs/foo.log".
#
# NOTE: Where filenames are specified, you must use forward slashes
# instead of backslashes (e.g., "c:/apache" instead of "c:\apache").
# If a drive letter is omitted, the drive on which Apache.exe is located
# will be used by default.  It is recommended that you always supply
# an explicit drive letter in absolute paths, however, to avoid
# confusion.
#

# ThreadsPerChild: constant number of worker threads in the server process
# MaxRequestsPerChild: maximum  number of requests a server process serves
```

```
ThreadsPerChild 250
MaxRequestsPerChild  0

#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do not add a slash at the end of the directory path.  If you point
# ServerRoot at a non-local disk, be sure to point the LockFile directive
# at a local disk.  If you wish to share the same ServerRoot for multiple
# httpd daemons, you will need to change at least LockFile and PidFile.
#
ServerRoot "C:/Program Files/Apache Software Foundation/Apache2.2"

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 80

#
# Dynamic Shared Object (DSO) Support
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding `LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Statically compiled modules (those listed by `httpd -l') do not need
# to be loaded here.
#
# Example:
# LoadModule foo_module modules/mod_foo.so
#
LoadModule actions_module modules/mod_actions.so
LoadModule alias_module modules/mod_alias.so
LoadModule asis_module modules/mod_asis.so
LoadModule auth_basic_module modules/mod_auth_basic.so
#LoadModule auth_digest_module modules/mod_auth_digest.so
#LoadModule authn_anon_module modules/mod_authn_anon.so
#LoadModule authn_dbm_module modules/mod_authn_dbm.so
LoadModule authn_default_module modules/mod_authn_default.so
LoadModule authn_file_module modules/mod_authn_file.so
#LoadModule authz_dbm_module modules/mod_authz_dbm.so
LoadModule authz_default_module modules/mod_authz_default.so
LoadModule authz_groupfile_module modules/mod_authz_groupfile.so
LoadModule authz_host_module modules/mod_authz_host.so
LoadModule authz_user_module modules/mod_authz_user.so
LoadModule autoindex_module modules/mod_autoindex.so
#LoadModule cern_meta_module modules/mod_cern_meta.so
LoadModule cgi_module modules/mod_cgi.so
#LoadModule dav_module modules/mod_dav.so
#LoadModule dav_fs_module modules/mod_dav_fs.so
#LoadModule deflate_module modules/mod_deflate.so
LoadModule dir_module modules/mod_dir.so
LoadModule env_module modules/mod_env.so
#LoadModule expires_module modules/mod_expires.so
#LoadModule file_cache_module modules/mod_file_cache.so
#LoadModule headers_module modules/mod_headers.so
LoadModule imagemap_module modules/mod_imagemap.so
LoadModule include_module modules/mod_include.so
#LoadModule info_module modules/mod_info.so
LoadModule isapi_module modules/mod_isapi.so
```

```
LoadModule log_config_module modules/mod_log_config.so
LoadModule mime_module modules/mod_mime.so
#LoadModule mime_magic_module modules/mod_mime_magic.so
#LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
#LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
#LoadModule proxy_connect_module modules/mod_proxy_connect.so
#LoadModule proxy_http_module modules/mod_proxy_http.so
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule negotiation_module modules/mod_negotiation.so
#LoadModule rewrite_module modules/mod_rewrite.so
LoadModule setenvif_module modules/mod_setenvif.so
#LoadModule speling_module modules/mod_speling.so
#LoadModule status_module modules/mod_status.so
#LoadModule unique_id_module modules/mod_unique_id.so
LoadModule userdir_module modules/mod_userdir.so
#LoadModule usertrack_module modules/mod_usertrack.so
#LoadModule vhost_alias_module modules/mod_vhost_alias.so
LoadModule ssl_module modules/mod_ssl.so

# 'Main' server configuration
#
# The directives in this section set up the values used by the 'main'
# server, which responds to any requests that aren't handled by a
# <VirtualHost> definition.  These values also provide defaults for
# any <VirtualHost> containers you may define later in the file.
#
# All of these directives may appear inside <VirtualHost> containers,
# in which case these default settings will be overridden for the
# virtual host being defined.
#

#
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.  e.g. admin@your-domain.com
#
ServerAdmin admin@metatrontech.com

#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
ServerName chris-PC.metatrontech.com:80

#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "C:/Program Files/Apache Software Foundation/Apache2.2/htdocs"

#
# Each directory to which Apache has access can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# features.
#
<Directory />
    Options FollowSymLinks
    AllowOverride None
```

```
    Order deny,allow
    Deny from all
    Satisfy all
</Directory>

#
# Note that from this point forward you must specifically allow
# particular features to be enabled - so if something's not working as
# you might expect, make sure that you have specifically enabled it
# below.
#

#
# This should be changed to whatever you set DocumentRoot to.
#
<Directory "C:/Program Files/Apache Software Foundation/Apache2.2/htdocs">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important.  Please see
    # http://httpd.apache.org/docs/2.2/mod/core.html#options
    # for more information.
    #
    Options Indexes FollowSymLinks

    #
    # AllowOverride controls what directives may be placed in .htaccess files.
    # It can be "All", "None", or any combination of the keywords:
    #   Options FileInfo AuthConfig Limit
    #
    AllowOverride None

    #
    # Controls who can get stuff from this server.
    #
    Order allow,deny
    Allow from all

</Directory>

#
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
#
<IfModule dir_module>
    DirectoryIndex index.html
</IfModule>

#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
</FilesMatch>

#
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
```

```
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog logs/error.log

#
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

<IfModule log_config_module>
    #
    # The following directives define some format nicknames for use with
    # a CustomLog directive (see below).
    #
    LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
    LogFormat "%h %l %u %t \"%r\" %>s %b" common

    <IfModule logio_module>
      # You need to enable mod_logio.c to use %I and %O
      LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio
    </IfModule>

    #
    # The location and format of the access logfile (Common Logfile Format).
    # If you do not define any access logfiles within a <VirtualHost>
    # container, they will be logged here.  Contrariwise, if you *do*
    # define per-<VirtualHost> access logfiles, transactions will be
    # logged therein and *not* in this file.
    #
    CustomLog logs/access.log common

    #
    # If you prefer a logfile with access, agent, and referer information
    # (Combined Logfile Format) you can use the following directive.
    #
    #CustomLog logs/access.log combined
</IfModule>

<IfModule alias_module>
    #
    # Redirect: Allows you to tell clients about documents that used to
    # exist in your server's namespace, but do not anymore. The client
    # will make a new request for the document at its new location.
    # Example:
    # Redirect permanent /foo http://chris-PC.metatrontech.com/bar

    #
    # Alias: Maps web paths into filesystem paths and is used to
    # access content that does not live under the DocumentRoot.
    # Example:
    # Alias /webpath /full/filesystem/path
    #
    # If you include a trailing / on /webpath then the server will
    # require it to be present in the URL.  You will also likely
    # need to provide a <Directory> section to allow access to
    # the filesystem path.

    #
    # ScriptAlias: This controls which directories contain server scripts.
    # ScriptAliases are essentially the same as Aliases, except that
    # documents in the target directory are treated as applications and
    # run by the server when requested rather than as documents sent to the
    # client.  The same rules about trailing "/" apply to ScriptAlias
```

```
    # directives as to Alias.
    #
    ScriptAlias /cgi-bin/ "C:/Program Files/Apache Software Foundation/Apache2.2/cgi-bin/"

</IfModule>

#
# "C:/Program Files/Apache Software Foundation/Apache2.2/cgi-bin" should be changed to whatever
your ScriptAliased
# CGI directory exists, if you have that configured.
#
<Directory "C:/Program Files/Apache Software Foundation/Apache2.2/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

#
# Apache parses all CGI scripts for the shebang line by default.
# This comment line, the first line of the script, consists of the symbols
# pound (#) and exclamation (!) followed by the path of the program that
# can execute this specific script.  For a perl script, with perl.exe in
# the C:\Program Files\Perl directory, the shebang line should be:

   #!c:/program files/perl/perl

# Note you _must_not_ indent the actual shebang line, and it must be the
# first line of the file.  Of course, CGI processing must be enabled by
# the appropriate ScriptAlias or Options ExecCGI directives for the files
# or directory in question.
#
# However, Apache on Windows allows either the Unix behavior above, or can
# use the Registry to match files by extention.  The command to execute
# a file of this type is retrieved from the registry by the same method as
# the Windows Explorer would use to handle double-clicking on a file.
# These script actions can be configured from the Windows Explorer View menu,
# 'Folder Options', and reviewing the 'File Types' tab.  Clicking the Edit
# button allows you to modify the Actions, of which Apache 1.3 attempts to
# perform the 'Open' Action, and failing that it will try the shebang line.
# This behavior is subject to change in Apache release 2.0.
#
# Each mechanism has it's own specific security weaknesses, from the means
# to run a program you didn't intend the website owner to invoke, and the
# best method is a matter of great debate.
#
# To enable the this Windows specific behavior (and therefore -disable- the
# equivilant Unix behavior), uncomment the following directive:
#
#ScriptInterpreterSource registry
#
# The directive above can be placed in individual <Directory> blocks or the
# .htaccess file, with either the 'registry' (Windows behavior) or 'script'
# (Unix behavior) option, and will override this server default option.
#

#
# DefaultType: the default MIME type the server will use for a document
# if it cannot otherwise determine one, such as from filename extensions.
# If your server contains mostly text or HTML documents, "text/plain" is
# a good value.  If most of your content is binary, such as applications
# or images, you may want to use "application/octet-stream" instead to
# keep browsers from trying to display binary files as though they are
# text.
#
DefaultType text/plain
```

```
<IfModule mime_module>
    #
    # TypesConfig points to the file containing the list of mappings from
    # filename extension to MIME-type.
    #
    TypesConfig conf/mime.types

    #
    # AddType allows you to add to or override the MIME configuration
    # file specified in TypesConfig for specific file types.
    #
    #AddType application/x-gzip .tgz
    #
    # AddEncoding allows you to have certain browsers uncompress
    # information on the fly. Note: Not all browsers support this.
    #
    #AddEncoding x-compress .Z
    #AddEncoding x-gzip .gz .tgz
    #
    # If the AddEncoding directives above are commented-out, then you
    # probably should define those extensions to indicate media types:
    #
    AddType application/x-compress .Z
    AddType application/x-gzip .gz .tgz

    #
    # AddHandler allows you to map certain file extensions to "handlers":
    # actions unrelated to filetype. These can be either built into the server
    # or added with the Action directive (see below)
    #
    # To use CGI scripts outside of ScriptAliased directories:
    # (You will also need to add "ExecCGI" to the "Options" directive.)
    #
    #AddHandler cgi-script .cgi

    # For type maps (negotiated resources):
    #AddHandler type-map var

    #
    # Filters allow you to process content before it is sent to the client.
    #
    # To parse .shtml files for server-side includes (SSI):
    # (You will also need to add "Includes" to the "Options" directive.)
    #
    #AddType text/html .shtml
    #AddOutputFilter INCLUDES .shtml
</IfModule>

#
# The mod_mime_magic module allows the server to use various hints from the
# contents of the file itself to determine its type.  The MIMEMagicFile
# directive tells the module where the hint definitions are located.
#
#MIMEMagicFile conf/magic

#
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
#
# Some examples:
#ErrorDocument 500 "The server made a boo boo."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://chris-PC.metatrontech.com/subscription_info.html
#
```

```
#
# EnableMMAP and EnableSendfile: On systems that support it,
# memory-mapping or the sendfile syscall is used to deliver
# files.  This usually improves server performance, but must
# be turned off when serving from networked-mounted
# filesystems or if support for these functions is otherwise
# broken on your system.
#
#EnableMMAP off
#EnableSendfile off

# Supplemental configuration
#
# The configuration files in the conf/extra/ directory can be
# included to add extra features or to modify the default configuration of
# the server, or you may simply copy their contents here and change as
# necessary.

# Server-pool management (MPM specific)
#Include conf/extra/httpd-mpm.conf

# Multi-language error messages
#Include conf/extra/httpd-multilang-errordoc.conf

# Fancy directory listings
#Include conf/extra/httpd-autoindex.conf

# Language settings
#Include conf/extra/httpd-languages.conf

# User home directories
#Include conf/extra/httpd-userdir.conf

# Real-time info on requests and configuration
#Include conf/extra/httpd-info.conf

# Virtual hosts
#Include conf/extra/httpd-vhosts.conf

# Local access to the Apache HTTP Server Manual
#Include conf/extra/httpd-manual.conf

# Distributed authoring and versioning (WebDAV)
#Include conf/extra/httpd-dav.conf

# Various default settings
#Include conf/extra/httpd-default.conf

# Secure (SSL/TLS) connections
#Include conf/extra/httpd-ssl.conf
#
# Note: The following must must be present to support
#       starting without SSL on platforms with no /dev/random equivalent
#       but a statically compiled-in mod_ssl.
#
<IfModule ssl_module>
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
</IfModule>
```

1.5 Final Thoughts

The above description should be sufficient to help you get a basic installation of Apache running on Windows while avoiding many common problems.

1.6 About the Author

Chris Travers is the owner of Metatron Technology Consulting, a firm dedicated to helping customers utilize open source software. He has over 7 years experience with Apache on both Windows and Linux.