Apache htaccess Files

What Are They

An htaccess file is the default name that the Apache web server uses for local access control. The name is actually *.htaccess* so that the files are hidden on Unix/Linux systems.

A .htaccess file is placed in a directory to control the access to that directory and *all subdirectories* of that directory. If there are .htaccess files in the subdirectories, they take precedence. In other words, the nearest .htaccess file makes the rules.

A .htaccess file is a text file and it must be world-readable since the web server (running as apache for example) needs to be able to read that file. Since you don't normally want outsiders to see the contents of the file, the master configuration file should specifically disallow web access (see the Apache Basics document).

There are dozens of Apache configuration directives and many can be used in .htaccess files. In general, they are the same directives and have the same effect. The directives are intended to be on one line in most cases, so make sure your text editor doesn't wrap lines.

The Testbed

For the following examples, assume that you are operating in the context of a user home directory (/home/testbed/www) and that there is a document root that contains directories for icons (/icons/) and global errors (/errors/). Under the www directory, you have directories named *pages*, and *error_pages*. One level above the www (user root) directory, you have a directory named wwwsafe, which is hidden from web access by virtue of being above the local document root.

Local Error Handling

If you want to provide customized error handling for a directory, you might try this:

```
ErrorDocument 401 ../error_pages/authorization_error.html
ErrorDocument 404 ../error_pages/notfound_error.html
ErrorDocument 403 "<body bgcolor=#ffffff><H1>You can't do that Bumpkin</H1>"
```

The numbers are standard error codes for Apache. As you can see, you can reference a file to be displayed, or simply provide the HTML desired.

Basic Authentication

If you want to password a directory tree, you have a number of choices. We will discuss two here; Basic Authentication and using a Perl module.

For a situation where you want to limit access to a small number of people, you would start by creating a password file. For example,

```
htpasswd -c /home/clowns/safedir/.htpasswd bozo
htpasswd /home/clowns/safedir/.htpasswd blimpy
```

creates a file and adds two users. The passwords will be prompted for. Then create a .htaccess file to use that file:

```
AuthUserFile /home/clowns/safedir/.htpasswd
AuthGroupFile /dev/null
AuthName SomeDomain
AuthType Basic
require valid-user
```

AuthUserFile names the password file for web users. *AuthGroupFile* names the group file, which is not used in this case. *AuthName* names the realm that is being passworded, and its primary purpose is to be displayed in the login prompt. *AuthType Basic* indicates that the type of authentication is the fundamental username/password form. Another type is *Digest* which uses a shared secret challenge, but few, if any, browsers support it.

require is the directive that actually generates password checking. It has the form:

require [user user1 user2 ...] [group group1 group2 ...] [valid-user]

valid-user simply allows any legal login to enter, while the others limit entry to a selected group. If you have a site with complex access requirements, you might have a single large password file, but only certain set of users are allowed in any situation.

Any three types or a combination can be used. For example:

require user bob joe mary group admins

AuthUserFile has lines of the form:

name:password

where *name* is the user name and *password* is the MD5 encrypted password. This file can be created in wwwsafe with:

htpasswd -c *password filename username*

and added to with:

htpasswd *password filename username*

You will be prompted for the password unless you provide the *-b* option and put the password (in the clear) on the command line.

AuthGroupFile has lines of the form:

groupname:username1 username2 ...

where *name* is the user name and *username* is the username that belongs to the group. For example:

retired:bozo blimpy

If used, the Group file restricts access to users in the groups allowed by the require directive, as in:

require group retired

Advanced Authentication

There are methods of Apache authentication for ASP, Perl, PHP and JavaScript, but we will consider only Perl. As it happens, the clever Perl folks created a module Apache::Authen which handles authentication for Apache for a variety of different types of situations. This includes but is not limited to:

Apache::AuthenNIS	Authenticate against an NIS database.
Apache::AuthenPasswd	Authenticate against /etc/passwd (or shadow).
Apache::AuthenSmb	Authenticate against an NT server.
Apache::AuthenLDAP	Authenticate against an LDAP database.
Apache::AuthenRadius	Authenticate against a RADIUS server.

For example, suppose that instead of creating a separate database, you simply want to authenticate with the normal user login. You would do the following:

```
AuthName "System Authentication"
AuthType Basic
PerlAuthenHandler Apache::AuthenPasswd
require valid-user
```

This requires that the mod_perl plugin for Apache is installed and that the Perl Authen:: module has to be installed.

Filtering Index Listing

If you want to limit the files that users can see when they list directories, you can do something like this:

```
Options Indexes+
IndexIgnore *.tex *.jpg *.doc
```

Deny Access by IP Address

If you want to limit access by IP address you can use the deny/allow pair.

```
order allow,deny
allow from all
deny from 192.168.1.1
```

or

```
order deny, allow
deny from all
allow from 192.168.1.1
allow from ! 222.222.222.222
```

You can also restrict by domain name by using the domain name in place of the IP address.

Redirects

If you want to redirect a request to another URL, use something like this:

```
Redirect /olddir/file.html http://site/newdir/newfile.html
```

This causes a reference to a specific directory and file to another file. You can also use only directories to redirect all access to a directory. For example:

```
Redirect /clowns/bozo/face.jpg http://www.clowns.org/retired/bozo.jpg
```

Setting Default Pages

In some instances, you might want to change the default page for a directory tree. This could be as simple as an active dislike for the name *index.html* or it could be an attempt to get control of certain directories. For example, suppose you have a directory that has a list of images, but you want to control the access to the files and provide annotations. You might put a *DirectoryIndex* directive in the .htaccess file.

```
DirectoryIndex index.pl
```

So that instead of simply giving a directory listing, it executes the Perl program *index.pl*.

For a directory tree, you might want to create an organization that has several different default pages depending on the situation:

```
DirectoryIndex index.html index.pl home.cgi noaccess.html
```

The Apache server will look for an index.html, then and index.pl, then a home.cgi and finally for noaccess.html in finding a default page to display.

Indexing

You can do quite a bit with indexing (file listing) with Apache. The most common directive is *IndexOptions* and there are a number of options:

FancyIndexing	Turns on fancy indexing, which adds icons and other data to the listing and allows the user to resort.
NameWidth=[n—*]	Specifies the width of the filenames in bytes or to the length of the longest filename if "*".
ScanHTMLTitles	Scan the html files for <TITLE> tags or use the AddDescription values.
SuppressColumnSorting	Suppresses column sorting for Fancy Indexing.
SuppressDescription	Suppresses descriptions in Fancy Indexing.
SuppressLastModified	Suppresses last modified data in Fancy Indexing.
SuppressSize	Suppresses file size Fancy Indexing.

Other important directives are:

AddDescription	Provides a description for a file.
AddIconByType	Specifies an icon to be used for a MIME type.
DefaultIcon	Set the default file icon.
HeaderName	Specifies the header to display at the top of the listing.
ReadMeName	Specifies the name of a file to be displayed.

For example,

```
IndexOptions FancyIndexing
IndexOptions NameWidth=16
IndexOptions SuppressSize
IndexOptions SuppressLastModified
AddDescription "File Catalog" filelist.html
AddIconByType (TXT,/icons/text.gif) text/*
HeaderName title.html
```