



Mastering Apache

Tim Anderson explores **how to configure this web server**, and applies some protection.

Apache marches on. The latest surveys suggest that around 60 per cent of Internet web servers use this free-to-deploy web server. Most of these are Unix-like operating systems, although there is a Windows version as well. It comes with warnings emphasising that it lacks the quality of the Unix release, but it still keeps a high standard. Even so, Internet Information Server (IIS) is the natural choice for Windows NT or 2000, but it is on other operating systems that Apache is such a fantastic bonus. What follows is based on Apache on Linux, but the Windows version is very similar.

Definitive documentation for Apache is difficult because it is extensible. Apache is extended by modules – supplementary code that is either compiled into the server executable, or else dynamically loaded. This last option is only available if your Apache was built with support for DSO (Dynamic Shared Objects), which are the Unix equivalent to Windows Dynamic Link Libraries. You can get useful information about your version of Apache by running:

```
httpd -v
```

for the version number and build date (or -V for a list of compile options).

Another handy command is:

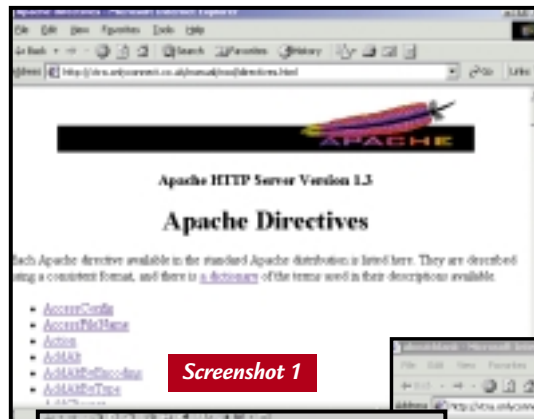
```
httpd -l
```

which lists statically compiled modules. If the list includes mod_so.c then DSO support is available.

Customising directories

Apache is configured by editing text files. The location of these varies, but httpd -V will show the location for your build. The key file is httpd.conf, which is heavily commented. In addition, each directive is explained in the Apache documentation (screenshot 1).

A common question is how to set up password-protected access. Before doing this, you need to understand how Apache works out the options applicable to a particular directory. The starting point is the DocumentRoot directive, which sets up the root directory. For example, if this is set as:



Screenshot 1



Screenshot 2

```
DocumentRoot /usr/local/  
apache/htdocs
```

(Key: ✓ code string continues)

and if the server is called pcw, then browsing to <http://pcw/mydoc.html> would look for the file /usr/local/apache/htdocs/mydoc.html.

DirectoryIndex is the directive that sets which web page will be served if the user enters a URL ending in a slash, in other words, a directory rather than a specific page.

Next, the Directory directive sets options for a specific directory and its sub-directories. If no directory is specified, then it sets default options for the root directory and all its subdirectories. In either case, a later Directory directive can supplement or overrule an earlier, less specific one.

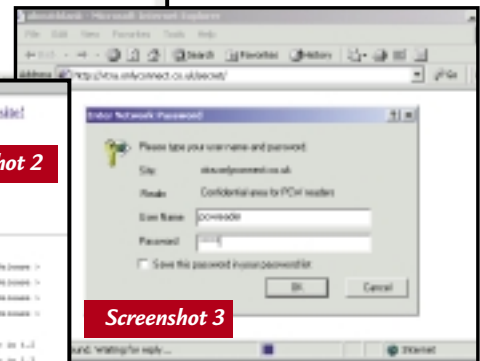
One of the directives valid in a Directory section is AllowOverride, for example:

```
AllowOverride All
```

Screenshot 1: The Apache documentation includes an explanation of all standard directives

Screenshot 2: You can download Apache from www.apache.org

Screenshot 3: The password-protected directory in action



Screenshot 3

This allows the use of an .htaccess file (the name can be changed by modifying AccessFileName) to control access and set other options for the directory. So if the directory 'secret' needs password protection, create a file called .htaccess in that directory. This file is not just for password protection; it can hold other directives as well. If AllowOverride is set to None, then .htaccess will have no effect.

Creating a password file

To create a password file, you need to use an Apache support executable called htpasswd. The password file itself must not be in the directory it protects, and is normally in a location such as /etc, well away from the web server files.

```
./htpasswd -cb /etc/htusers ✓  
.pwd pcwreader webdev
```

The -c argument creates a new password file and is not used if you want to append to an existing file. The -b argument reads the password from the command line. If you look at the file you created, you will find that it has the user name followed by an encrypted password.

Applying the protection

Now you can go back to .htaccess in the 'secret' directory and apply the protection. Here is a simple example:

```
AuthUserFile ✓  
/etc/htusers.pwd  
AuthName "Confidential ✓  
area for PCW readers"  
AuthType Basic  
require user pcwreader  
DirectoryIndex ✓  
special.html
```

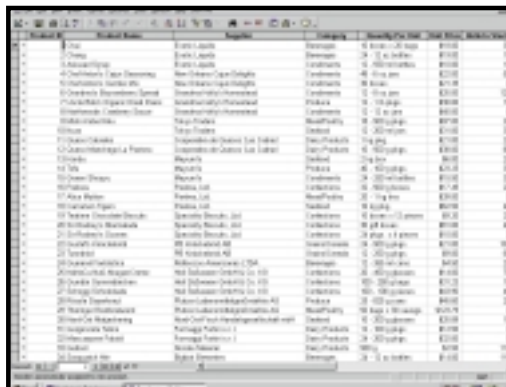
The DirectoryIndex directive is not required, but is included to show how you can use an access file for things other than password protection (screenshot 3).

For greater flexibility, you will probably want to define an AuthGroupFile that lets you define groups of users like this:

```
pcwgroup: tim niall pcwreader
```

Save this as, say, /etc/htgroups.grp and then amend .htaccess like this:

```
AuthUserFile /etc/htusers.pwd
```



Product Name	Supplier	Price	Stock
1. Apple	Apple Inc.	100.00	1000
2. Banana	Banana Ltd.	50.00	500
3. Cherry	Cherry Corp.	75.00	750
4. Date	Date Co.	120.00	1200
5. Fig	Fig Farm	80.00	800
6. Grape	Grape Growers	60.00	600
7. Lemon	Lemon Ltd.	40.00	400
8. Mango	Mango Inc.	90.00	900
9. Orange	Orange Orchard	30.00	300
10. Peach	Peach Perfection	55.00	550
11. Pineapple	Pineapple Pro	110.00	1100
12. Raspberry	Raspberry Ridge	150.00	1500
13. Strawberry	Strawberry Fields	65.00	650
14. Tangerine	Tangerine Tree	45.00	450
15. Watermelon	Watermelon Works	200.00	2000

Store image files externally with Access 2000

```
AuthGroupFile ✓  
/etc/htgroups.grp  
AuthName "Confidential area ✓  
for PCW readers"  
AuthType Basic  
require group pcwgroup  
DirectoryIndex special.html
```

For more on Apache see this month's *Hands On Workshop*.

Images in Data Access

Steve Dalzell asks: 'I have an Access 2000 database with an OLE field in which is stored an image. How can I display it on a Data Access Page?'

Data Access Pages do not support image controls bound to OLE image fields. You have to bind an image control to a field that contains a path to an image. This means amending the Access 2000 database so that the images are stored externally, with only the path to the image stored in the MDB.

CONTACTS

Tim Anderson welcomes your comments on the Web Development column. Contact him via the PCW editorial office or email:

webdev@pcw.co.uk

Working examples from this column are

posted at **www.onlyconnect.co.uk**

All things Apache start at **www.apache.org**