Website Development I

HOW TO RESTRICT ACCESS TO STUDENT FOLDERS

Version 1-2, revised Feb. 21, 2022

prepared for

Dave Schneider

prepared by

Jesse M. Heines, Civilian Volunteer

SUMMARY

This document explains how to use the Apache server's built-in authentication features to restrict access to specific folders in a student's directory tree. The advantage of the approach explained here is that one can make some folders public while keeping others private.

This approach has five main steps:

1. configuring Apache via the `httpd.conf` file
2. creating and populating a `users` file of authorized usernames and passwords
3. creating the student folders to be password-protected
4. creating and configuring a `.htaccess` file to restrict access to password-protected folders
5. creating an HTML file to automatically redirect browsers to the student's `public` folder

ASSUMPTIONS

The explanations in this document assume that student folders for this course are set up hierarchically as follows:

```
D:\Dropboxes
└──Website_Development_I
   └──student11
      ├──private
      └──public
```

The explanations further assumes that Apache was installed using XAMPP in a standard manner and is running as a service. This assumption implies that Apache and all of its utilities reside in `C:\xampp` and its subfolders.

Finally, this document assumes that students' main folders for this course have been specified using `alias` directives in the `alias_module` section of file `C:\xampp\apache\conf\httpd.conf`. For example:

```
<IfModule alias_module>
   Alias /student11 "D:/Dropboxes/Website_Development_I/student11"
</IfModule>
```

This section may, of course, contain many other `alias` directives as well.

Step 1: CONFIGURING APACHE

In a standard XAMPP installation, the master `httpd.conf` configuration file can be found at:

```
C:\xampp\apache\conf\httpd.conf
```

The first step is to edit this file to allow access to student folders and to specify the commands that are allowed in `.htaccess` files.

Editing the `Directory` Section

The `Directory` section in the original `httpd.conf` file immediately after Apache installation looks like this:

```
# Deny access to the entirety of your server's filesystem. You must
# explicitly permit access to web content directories in other
# <Directory> blocks below.
#
<Directory />
    AllowOverride none
    Require all denied
</Directory>
```

The lines that begin with hash signs (#) are comments. We will focus on the lines that do not begin with hash signs.

Inside the two `Directory` lines, the first indented line (called a *directive* in the Apache documentation) pertains to what one can "override" in a `.htaccess` file. The default parameter is "`none`," which prevents `.htaccess` files from overriding *any* of the permissions specified in the master `httpd.conf` file.

The second line (or directive) pertains to which servers are allowed to access files on the current server. The default parameters "all denied" prevent any server — ***including the one we're running on!*** — to access files on the current server.

Thus, in both cases, all permissions are essentially turned "off" by default, so we need to turn them "on" for `.htaccess` files to work as we want.

To do this, change the two lines in the `Directory` section to read as follows.

```
<Directory />
    AllowOverride all
    Allow from all
</Directory>
```

The first line (the `AllowOverride` directive) now allows **all** available directives and options to be used in `.htaccess` files.

The second line now allows any server to access the files on the current server. This is OK on a local area network, because only systems on that network have access to the host server. You may not wish to use the "all" option on a server connected to the Internet.

Checking the `Files` Section

While you're editing the `httpd.conf` file, check the `Files` section as well. This section should look like this:

```
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<Files ".ht*">
    Require all denied
</Files>
```

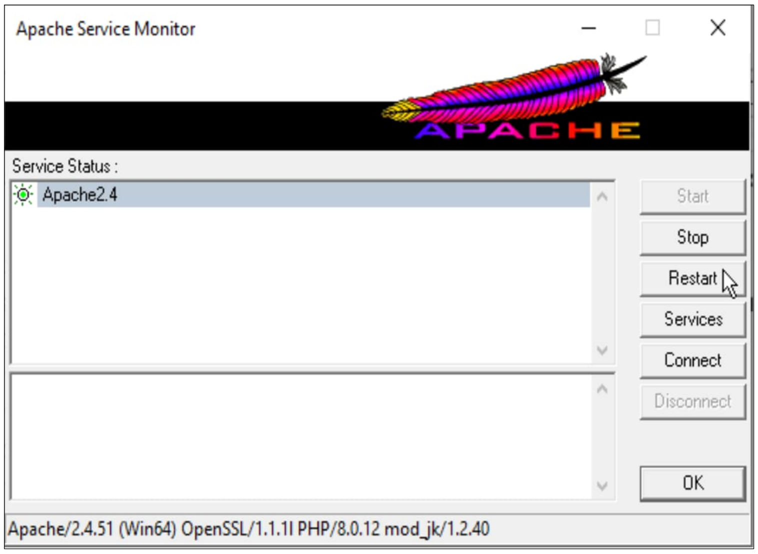
This is the default and is exactly how we want this section to appear, so I'm just recommending that you check it to make sure. As the comment indicates, this code prevents web browsers from being able to view the `.htaccess` files that we will be creating.

Restarting Apache

Any time you change anything in the httpd.conf file you need to restart the Apache service. The easiest way to do this is the run the Apache Monitor program:

```
C:\xampp\apache\bin\ApacheMonitor.exe
```

This will place an icon in your taskbar that looks like this: . Double-clicking this icon will bring up the Apache Service Monitor. Click the `Restart` button to restart Apache.



Apache/2.4.51 (Win64) OpenSSL/1.1.1l PHP/8.0.12 mod_jk/1.2.40

Step 2: SPECIFYING AUTHORIZED USERS

A `users` file contains login information for people who are authorized to access the files in a folder via a web browser. This login information consists of a username in plain text and a password that has been hashed, which means that it looks like gibberish to a human being trying to read the file, but Apache knows how to convert the hashed text back to a form that it can use to compare it to whatever the user entered.

The standard Apache installation provides a program named `htpasswd.exe` to both create a `users` file and to add login information to that file. This program is located in the Apache `bin` folder:

```
C:\xampp\apache\bin\htpasswd.exe
```

The program has a number of *switches* — letters preceded by hyphens (`-`), such as `-b` and `-c` — that can be used to control is functionality, but we only need to learn about a few of them. (To see the other switches and a brief explanation of what they do, simply run the `htpasswd.exe` program with no parameters.)

> *Note:* The instructions in this section all involve working in a DOS command prompt window. To open this window, click the Windows Start button, enter `cmd`, and press the Enter key.

Adding the Apache `bin` Folder to the System Path

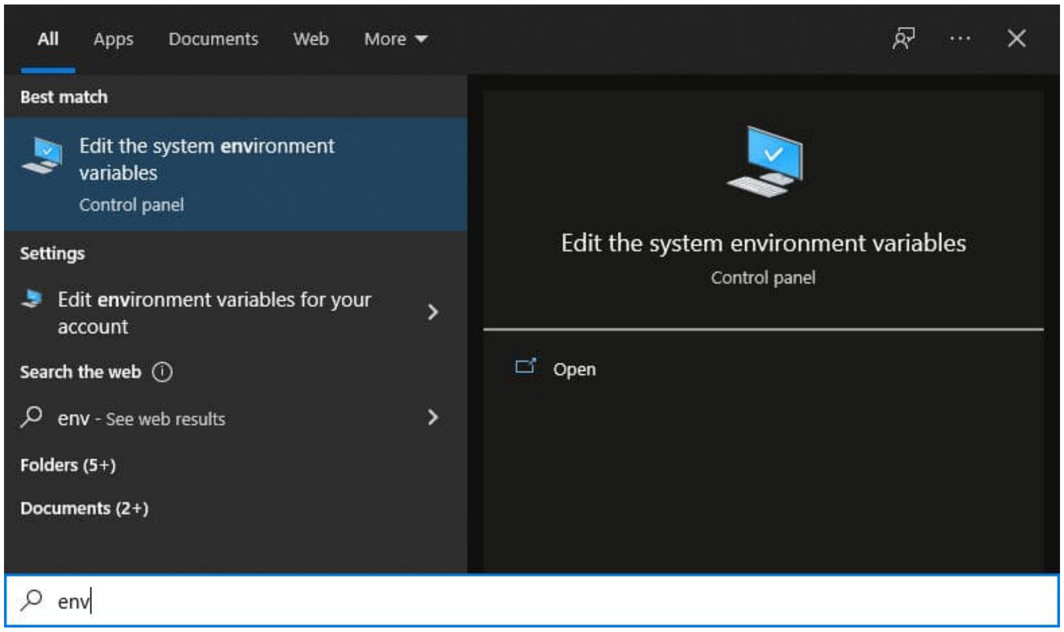
To simplify using the `htpasswd.exe` program so that you don't have to type is full path as shown above, add the Apache `bin` folder to the system's `path` environment variable. This variable specifies where DOS will look for programs when you type a program name on the command line.

Using the Windows Environment Variable Editor

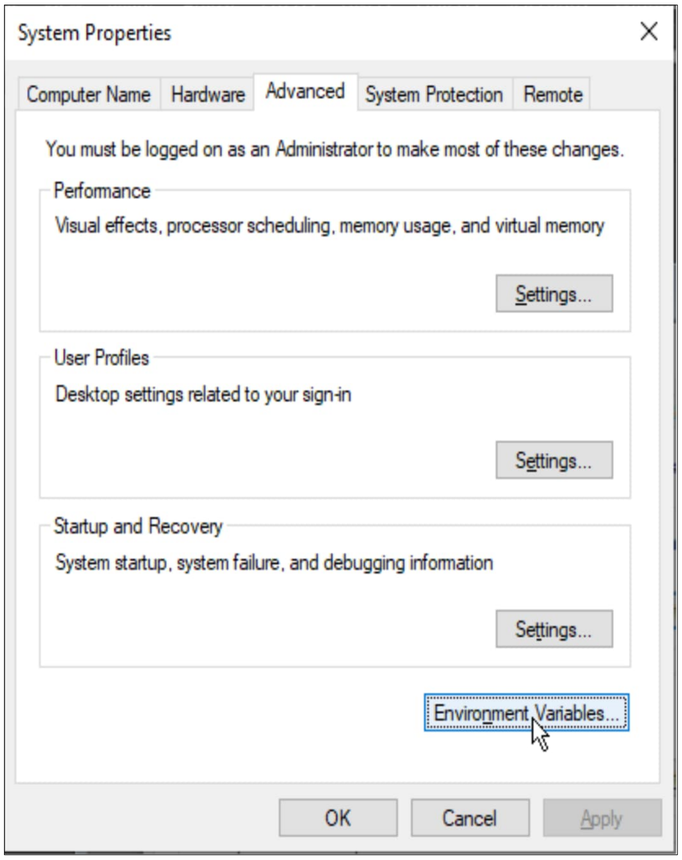
One way to edit the `path` environment variable is using the built-in Windows Environment Variable Editor. There are other ways, but this way is the least error-prone. When editing system variables as we are about to do, you really don't want to make errors, as doing so may "break" your system or make it behave in undesirable ways.

You can get to the Environment Variable Editor through the Windows Control Panel, but the sequence of clicks to do so varies greatly depending upon your version of Windows.

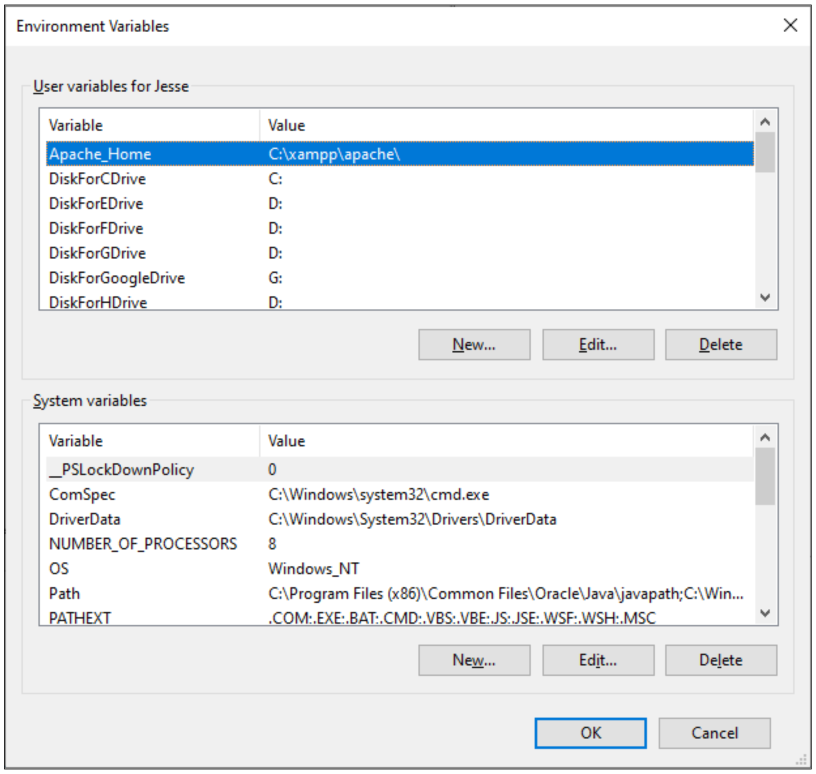
I therefore find it easiest to click the Windows `Start` button and then to start typing the word "environment" until the "Edit the system environment variables" option appears as shown in the screen capture at the top of the next page.



As soon as this option appears, press the Enter key to open the System Properties dialog box with the Advanced tab selected.


System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your sign-in
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply

Click the Environment Variables… button in this dialog box to open the Environment Variable Editor.



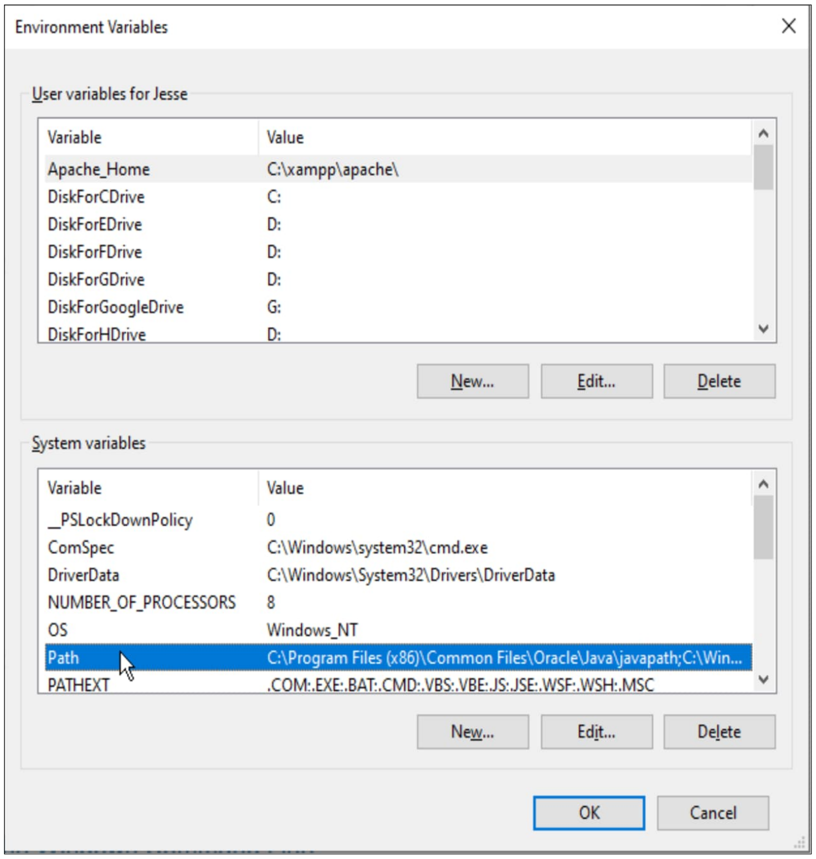
User variables for Jesse

Variable	Value
Apache_Home	C:\xampp\apache\
DiskForCDrive	C:
DiskForEDrive	D:
DiskForFDrive	D:
DiskForGDrive	D:
DiskForGoogleDrive	G:
DiskForHDrive	D:

System variables

Variable	Value
__PSLockDownPolicy	0
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Win...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

Click Path in the System variables section, and then click the Edit… button.



User variables for Jesse

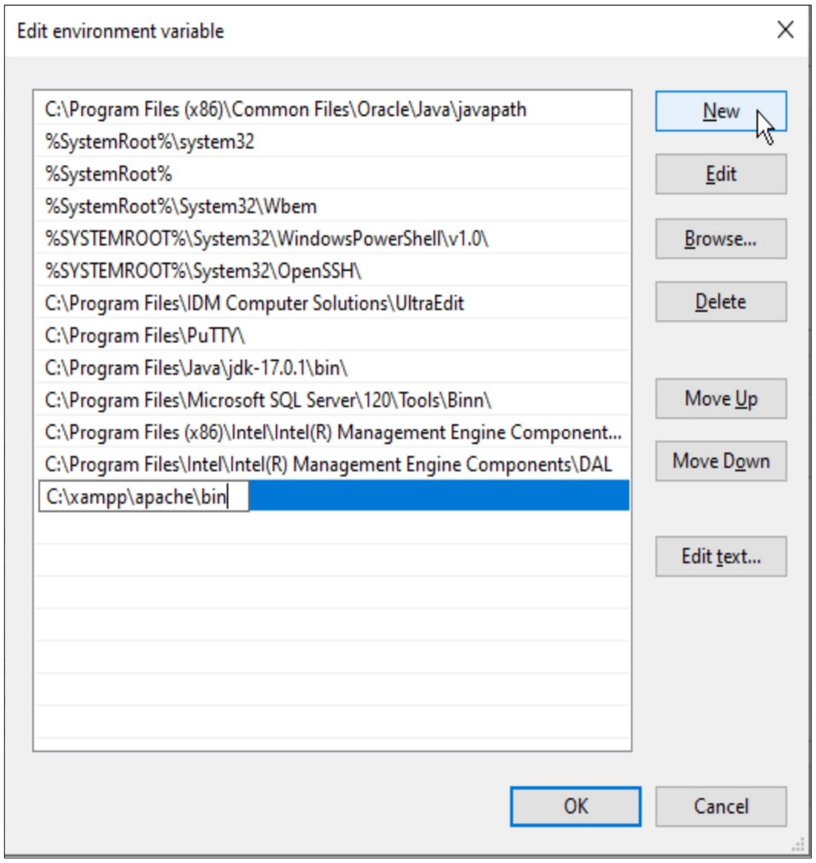
Variable	Value
Apache_Home	C:\xampp\apache\
DiskForCDrive	C:
DiskForEDrive	D:
DiskForFDrive	D:
DiskForGDrive	D:
DiskForGoogleDrive	G:
DiskForHDrive	D:

System variables

Variable	Value
__PSLockDownPolicy	0
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Win...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

This brings up the `Edit environment variable` dialog box. Here, click the `New…` button. This will highlight the last line in the list and allow you to add a new folder to the path environment variable.

Type `C:\xampp\apache\bin` in the space provided and then click the `OK` button.


Edit environment variable

```
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\IDM Computer Solutions\UltraEdit
C:\Program Files\PuTTY\
C:\Program Files\Java\jdk-17.0.1\bin\
C:\Program Files\Microsoft SQL Server\120\Tools\Binn\
C:\Program Files (x86)\Intel\Intel(R) Management Engine Component...
C:\Program Files\Intel\Intel(R) Management Engine Components\DAL
C:\xampp\apache\bin
```

At this point, the `Environment Variables` and `System Properties` dialog boxes will still be open. Click the `OK` button in each of these dialog boxes to close them.

You should then be able to enter simply `htpasswd` on the command line and DOS should find and execute the `C:\xampp\apache\bin\htpasswd.exe` program.

> *Note:* When you change environment variables in this manner, the change is permanent. That is, the changed `path` environment variable will still have its new value even after you reboot your system.

Creating a New `users` File

Open a DOS command window and use the `cd` command to get to the student's main folder for this course. For example:

```
> cd D:\Dropboxes\Website_Development_I\student11
```

You are now ready to create the `users` file to control access to this student's folders.

> *Note:* You want to create the users file in the student's top-level folder so that it can be easily accessed by all the folders in his directory tree.

If the `users` file does not yet exist, enter a command similar to the following.

```
> htpasswd -c -b users student11 password
```

Let's look at the last three elements on this command line first.

- `users` is the name of the file that will contain the login information
- `student11` is the username of the user we are adding to the users file
- `password` is the password for the user we are adding

> *Note:* Usernames and passwords are ***case-sensitive***. That is, if we enter `student11`, this is not the same as `Student11`.

The command line elements that begin with hyphens (`-`) are called *switches*. They control what the `httpasswd.exe` program does with the parameters passed to it.

The `-c` switch causes this command to *create* the authorized users file with a name of `users`. If a `users` file already exists, ***it will be deleted and replaced!*** It is therefore critical that the `-c` switch only be used the *first time* we want to create the `users` file.

The `-b` switch tells the `htpasswd.exe` program that you will be entering the user's password on the command line. This is fine if no one is looking over your shoulder. If you are working in an open environment with people watching, you might want to leave the `-b` switch off and just enter the above command in the following form:

```
> htpasswd -c users student11
```

In this case, the program will then prompt you to enter a password for `student11`. But instead of displaying the password, the program will display asterisk (*). It will then ask you to re-type the new password to make sure that you typed what you wanted to type. If both password entries match, the program will confirm that it added the student's username and password.

```
New password: ********
Re-type new password: ********
Adding password for user student11
```

After the above commands, the contents of the `users` file will look something like this:

```
> type users
student11:$apr1$CTPyV2fm$lrb5GbMnXBCx03fBW8FU20
```

The username is the part before the colon (`:`). The part after the colon is that user's password after being hashed. This prevents anyone who gains access to the file from being able to read the user's password, as it can only be interpreted by the system on which the hash was created.

Adding Authorized Users to an Existing `users` File

To add a new username and password to an *existing* `users` file, be sure to *omit* the `-c` switch from the command line. Thus, you could add `student12` with a password of “password” by typing:

```
> htpasswd -b users student12 password
```

As before, if someone is or may be looking over your shoulder as you type, use the long form of this command instead. To do that, omit the `-b` switch and the password from the command line:

```
> htpasswd users student12
New password: ********
Re-type new password: ********
Adding password for user student12
```

After adding `student12`, contents of the `users` file will look something like this:

```
student11:$apr1$CTPyV2fm$lrb5GbMnXBCx03fBW8FU20
student12:$apr1$RbMYWh1O$1Mw1UOPYjIdK77yOaolnl0
```

Note: Even though I entered “`password`” as the password for both `student11` and `student12`, their two hashes are different. This is because the hashing algorithm uses a variety of factors like the current time when it creates the seemingly random characters that comprise the password hash.

Adding Usernames Containing Spaces

One last thing before we move on to the next step: If you wish to include a space in a username, you must enclose the username in quote marks, like this:

```
> htpasswd -b users "Dave Schneider" password
```

Assigning, Verifying, and Changing Usernames and Passwords

Apache provides no utilities to assign usernames and passwords in the `users` file. You may use whatever naming convention you want for student usernames. For passwords, you may wish to assign a password for each student account and tell him what that is, or, if you want to allow students to choose their own passwords so that they can more easily remember them, they will have to tell you their password so that you can enter in using the `htpasswd.exe` program.

If a student tells you that his password no longer works, you can test that password using the `htpasswd.exe` program with the `-v` switch. In the first case below, the correct password for `student11` was entered. In the second case, an incorrect password was entered.

```
> htpasswd -v users student11
Enter password: ********
Password for user student11 correct.
```

```
> htpasswd -v users student11
Enter password: *********
password verification failed
```

As before, you can use the -b switch to enter the password on the command line if no one is looking over your shoulder.

```
> htpasswd -b -v users student11 password
Password for user student11 correct.

> htpasswd -b -v users student11 incorrect
password verification failed
```

To change (update) a student's password, simply reenter it as if you are adding that user for the first time.

```
> htpasswd -b users student11 newpassword
Updating password for user student11
```

Changing Usernames

While you can change (update) a student's password as explained above, there is no single command to change a student's username. To do that, you delete the student's current entry in the `users` file and then add it again.

To delete an entry, use the `-D` switch. *Note that the `D` in this switch is capitalized.*

```
> type users
student11:$apr1$jQX0a9/l$FJvtwJyBn8.ENpB.RAlgK.
Dave Schneider:$apr1$3yu9pbP7$68jeDYMNDPCmtfqwSwZYt/

> htpasswd -D users student11
Deleting password for user student11

> type users
Dave Schneider:$apr1$3yu9pbP7$68jeDYMNDPCmtfqwSwZYt/

> htpasswd -b users student11a password
Adding password for user student11

> type users
Dave Schneider:$apr1$3yu9pbP7$68jeDYMNDPCmtfqwSwZYt/
student11a:$apr1$G2kj8PQM$IivnGLjtIPNttsTnvLPS6.
```

It is interesting to note that even though I entered "`password`" as the password for both `student11` and `student11a`, the hash for `student11a`'s password is slightly different from that for `student11`. Once again, this is because to create a secure hash, the hashing algorithm uses parameters other than just the text string ("`password`") that you entered.

Using Multiple Switches on the Command Line

One last thing before we move on to the next step: When you use multiple switches on a command line you may simply use one hyphen and put two or more switches immediately after each other. However, note that when you do this there must not be a space between the switches. For example, instead of:

```
> htpasswd -b -v users student11 password
Password for user student11 correct.
```

you could enter:

```
> htpasswd -bv users student11 password
Password for user student11 correct.
```

Step 3:
CREATING STUDENT FOLDERS

Password protection is a folder-level feature. That is, if we have a folder hierarchy that looks like this:

```
D:\Dropboxes
└──Website_Development_I
   └──student11
      ├──private
      │  ├──data
      │  └──scripts
      └──public
```

and we password-protect the `private` folder, the `data` and `scripts` folders will also be password-protected unless we *explicitly remove* password protection from those folders. It is very, very rare that we would want to do this, so I won't go into that here. In the vast majority of cases, we want to password-protect and entire subtree.

The folders to be password-protected are simply normal folders. They can be created using the `mkdir` DOS command or via the File Explorer program. The trick to password-protecting them is the next step.

Step 4:
RESTRICTING ACCESS

The key to password-protecting a folder hierarchy is to create a file named `.htaccess` in the hierarchy's top-level folder. With the changes that we made to the `httpd.conf` file, the Apache server will look for and read this file to see how to handle access control.

> *Note:* The name of the `.htaccess` file is rather strange. It looks like the file has no name, but has "`htaccess`" as its extension. This file naming convention is an artifact of the Unix system naming convention in which files whose names start with a period (`.`) are "hidden," meaning that they do not appear in normal directory listings. This convention is *not* followed in Windows, but the Apache server was originally developed on Unix, so the naming convention persists.

Creating the `.htaccess` File

The `.htaccess` file can be created with any text editor. At a minimum, it must contain three lines to control access to the folder in which it resides. As an example, here are the three lines that would need to be placed file:

```
D:\Dropboxes\Website_Development_I\student11\private\.htaccess
```

to restrict access to that folder — and all of its subfolders — to users `student11` and `"Dave Schneider"`:

```
AuthType Basic
AuthUserFile "D:\Dropboxes\Website_Development_I\student11\users"
Require user student11 "Dave Schneider"
```

Let's look at each of these lines in turn.

- The `AuthType` line must be exactly as shown. This line tells Apache that the type of the authorization to be uses is "`Basic`". There is one other type of authorization that Apache supports, but it is no longer recommended.
- The `AuthUserFile` line tells Apache which `users` file to look at for authorized usernames and passwords. Note that this directive must be followed by the full path to the `users` file, including the drive letter. It is also recommended that the full path be enclosed in quotes.
- The `Require user` line provides a list of usernames that are authorized to access the folder (and subfolders) in which the `.htaccess` file resides.

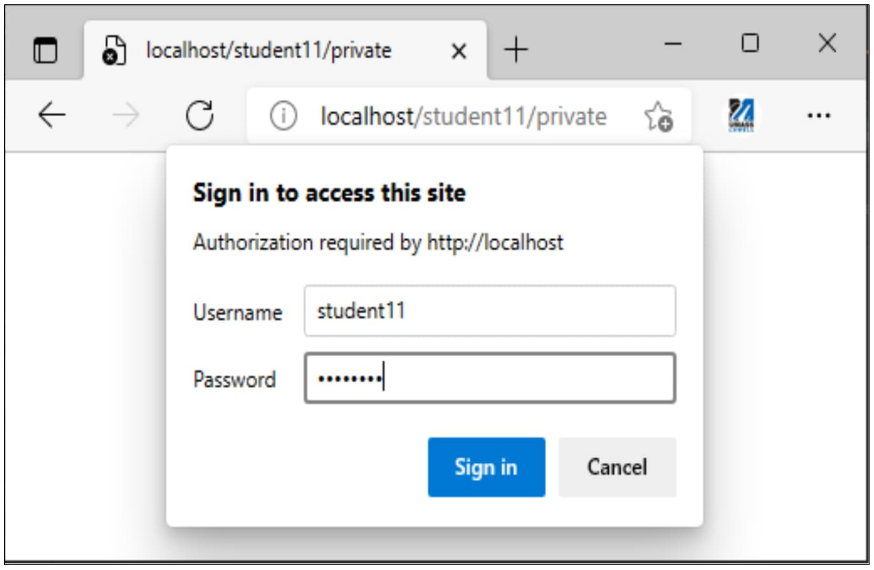
  Note: The word "`user`" ***must*** follow the `Require` directive, and "`user`" ***must*** be in all lowercase letters.

  As written in the example, there may be more than one authorized user specified on this line. And if a username contains a space, it must be enclosed in quote marks.

Accessing the Protected Folder

If everything is set up properly, when any user browses to `http://PI/student11/private`, he should be presented with a dialog box similar to the one on the next page. (Such dialog boxes vary slightly in different browsers.)

If the user enters a username and password that match an entry in the `users` file, the server will display the default page in `D:\Dropboxes\Website_Development_I\student11\private`. If an invalid username or password is entered, the browser will either display an error message or simply clear and redisplay the `Sign in` dialog box.


localhost/student11/private

Sign in to access this site

Authorization required by http://localhost

Username student11

Password ••••••••

Understanding Sessions

The user will remain signed in as long as the browser remains open. This is true even if the user switches to a new tab.

Internally, the browser creates a temporary session variable that indicates that the user is signed in. This variable remains active until the browser is closed. There is no simple way to "sign out" of a session, so students should be instructed to close their browser windows when they leave the room to ensure that others cannot simply sit down at their computers and access their private files.

Step 5: REDIRECTING FROM TOP LEVEL

If you have set up a student's main folder as suggested in the assumptions, the instructions in this document will protect a student's `private` folder. If the student places files in his `public` folder, those files will be visible to any user who browses to that folder. This setup is desirable so that a student can "publish" his pages for others to view, learn from, and comment on.

```
D:\Dropboxes
└──Website_Development_I
    └──student11
        ├──private
        └──public
```

The only remaining issue is that if a user browses to `http://PI/student11`, one of three things will happen.

- If a default (`index.html` or the like) file exists in the top-level folder, that file will be displayed.

- If a default file does not exist AND directory browsing is enabled in the `httpd.conf` file, the browser will display a listing showing the private and public folders that the user can click to enter those folders.
- If a default file does not exist and directory browsing is NOT enabled in the `httpd.conf` file, the user will see an error message.

A better approach to set up a student's site is to have `http://PI/student11` automatically redirect to `student11`'s `public` folder. To do this we need to create a small HTML file in the student's top-level folder that *redirects* to his `public` folder.

The redirection is done with a single line of JavaScript, but for that to work we need to create a basic HTML file to give the JavaScript context.

Create a file named `index.html` in the student's top-level folder (`D:\Dropboxes\Website_Development_I\student11` in the case of `student11`) and enter the following code:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title></title>
  <script>
    window.location.replace( "public/" ) ;
  </script>
</head>
<body>
</body>
</html>
```

The key line here is the one inside the `<script>...</script>` tags. This is the single line of JavaScript that replaces the current window with the default file in the student's `public` folder.

With this file in place, going to `http://PI/student11` automatically redirect to `http://PI/student11/public`.

CLOSING

That's it! There is a lot of material here, but once you get things set up for one student, setting up and password-protecting another student's folders should be straightforward and take much less time that doing so the first time.

As always, I would be happy to assist with this process and/or to answer any questions that you may have.