Lab 1.8

Setting up password protection for your website

Objectives

After completing this lab, you will be able to:

- Implement password protection for a website to enforce basic access control.

Synopsis

Protecting a website can be a laborious task since sites face countless threats from all over the Internet around the clock. For this reason, the administrators of the LabBook Support Server have already implemented appropriate safeguards to ensure the server's robustness against threats. However, users can certainly do more to heighten the security of their own websites. In this lab, you will learn how to apply password protection for a directory of your website so that only authorized users can access its contents. This technique is very practical since there are often occasions when we need to grant certain individuals access to part of our site content while barring the general Internet population.

Prerequisites

- You have a user account with remote login privileges on the LabBook Support Server, which has been configured to handle CGI scripts under your personal homepage directory.
- You are familiar with the use of an SSH client such as PuTTY.
- Your PC is connected to the Internet.

Background reading and preparation

Read the following online resources before doing this lab:

- 'Common Gateway Interface overview' — http://hoohoo.ncsa.uiuc.edu/cgi/intro.html

  You are expected to spend no more than five minutes reading this webpage.
- 'Common Gateway Interface' — http://en.wikipedia.org/wiki/Common_Gateway_Interface

  You are expected to spend no more than five minutes reading this article.

You are expected to spend no more than ten minutes on this preparation.

Expected duration

Approximately one hour (including background reading and preparation)

Procedure

Step 1 Setting up your homepage and CGI directories

The C Internet applications that you will develop in this lab will be accessed from your own personal website on the LabBook Support Server. The URL of this website will be:

http://labsupport.no-ip.org/~s1234567

(As usual, replace *s1234567* with your own username.) In order to let the LabBook Support Server serve the contents in your personal website, you need to do the following:

- Make your home directory accessible to other users.
- Create a publicly accessible and readable directory `public_html` under your home directory. This directory will be the root directory of your personal website.
- Create a publicly accessible and readable directory `cgi-bin` under `public_html`. Your C Internet applications will be deployed in `cgi-bin`.

Launch PuTTY (or another SSH client that you prefer) and connect to the LabBook Support Server as in Lab 1.1.

Enter the command `ls -l ~/.. | grep 'whoami'` to check the access permissions of your home directory. Remember that if you want to view the access permissions associated with a file or a directory, you need to use the `-l` option of `ls`. The argument `~/..` following the `ls` command indicates that you want to view all files and directories above your home directory. However, you are actually interested only in the file permissions associated with your own home directory. Hence the output of `ls` is piped to `grep` to filter out the unwanted information. The argument following `grep` , `'whoami'`, specifies that `grep` should output only those lines that contain your username. As a result, the output of the whole command will show the access permissions of your home directory only.

```
[s1234567@labsupport ~]$ ls -l ~/.. | grep 'whoami'
drwx------  8 s1234567 ct212-apr07 4096 May  3 18:03 s1234567
[s1234567@labsupport ~]$ █
```

To make your webpage directory accessible to other users, you should first turn on the execute permission of your home directory. To do so, enter the command `chmod +x ~`. Then verify that the execute permission is indeed turned on.

```
[s1234567@labsupport ~]$ chmod +x ~
[s1234567@labsupport ~]$ ls -l ~/.. | grep 'whoami'
drwx--x--x  8 s1234567 ct212-apr07 4096 May  3 18:03 s1234567
[s1234567@labsupport ~]$ █
```

Note: The execute permission of a directory determines whether the directory can be accessed. It does not mean that the directory can be 'executed' in any way. On the other hand, the read permission of a directory determines whether its contents can be listed using the `ls` command. Make sure you understand the differences between execute permission and read permission, and how they should be set according to your needs.

Next, create a new directory called `public_html` under your home directory. It will be the root directory for your personal website. Then create another new directory `cgi-bin` under `public_html`. Verify that these two new directories are readable and accessible by all users.

```
[s1234567@labsupport ~]$ mkdir public_html
[s1234567@labsupport ~]$ mkdir public_html/cgi-bin
[s1234567@labsupport ~]$ ls -l
drwxr-xr-x  2 s1234567 ct212-apr07  4096 May  3 20:02 public_html
[s1234567@labsupport ~]$ ls -l public_html
drwxr-xr-x  2 s1234567 ct212-apr07  4096 May  3 20:02 cgi-bin
[s1234567@labsupport ~]$ █
```

We have prepared an HTML document and a CGI script to let you test your personal website. These two files are compressed as a *tarball* (see Lab 1.6 Step1 for information about tarballs) which can be downloaded here:

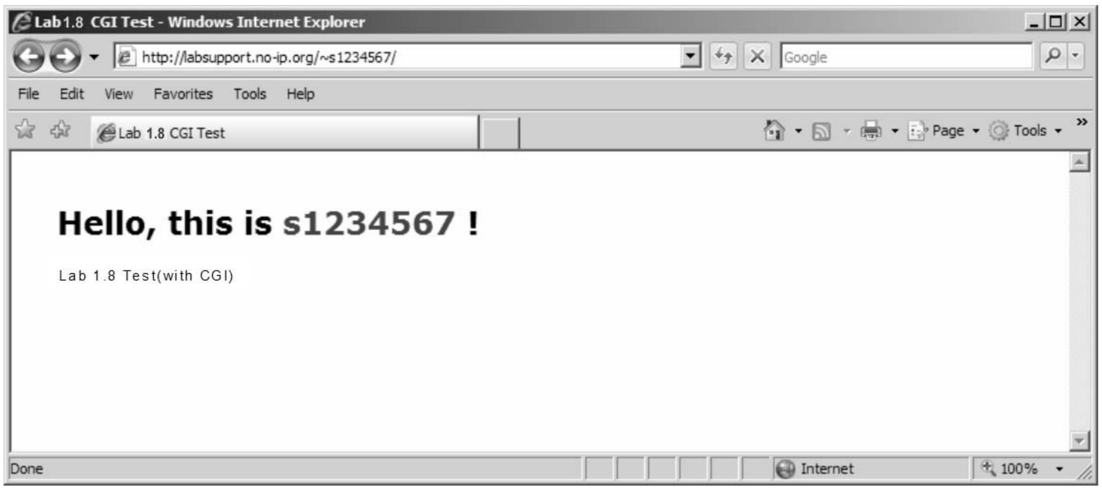
http://labsupport.no-ip.org/labs/2009/1.8/hello.tgz

Change to the `public_html` directory that you have just created. Use the `wget` command to download the tarball into `public_html`. Then use the `tar` command to extract the files from the tarball. After the extraction, you should find one new file, `index.html`, under `public_html`, and another new file, `hello.cgi`, under `cgi-bin`. Verify that `index.html` is readable by all users. In addition, verify that `index.html` has the execute permission enabled.[1] For `hello.cgi`, verify that it is readable and executable by all users. You can use the `chmod` command to change the permissions as necessary.

```
[s1234567@labsupport public_html]$ wget http://labsupport.no-ip.org/2009/1.8/hello.tgz
--23:51:32--  http://labsupport.no-ip.org/labs/2009/1.8/hello.tgz
Resolving labsupport.no-ip.org... 210.17.156.179
Connecting to labsupport.no-ip.org|210.17.156.179|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 452 [application/x-gzip]
Saving to: 'hello.tgz'

100%[=================================================>] 452         --.-K/s   in 0s

23:51:32 (49.5 MB/s) - 'hello.tgz' saved [452/452]

[s1234567@labsupport public_html]$ ls -l
total 4
-rw-r--r-- 1 s1234567 ct212-apr07 452 May 11 23:41 hello.tgz
[s1234567@labsupport public_html]$ tar xvzf hello.tgz
index.html
cgi-bin/
cgi-bin/hello.cgi
[s1234567@labsupport public_html]$ ls -l
total 12
drwxr-xr-x 2 s1234567 ct212-apr07 4096 May 11 23:39 cgi-bin
-rw-r--r-- 1 s1234567 ct212-apr07  452 May 11 23:41 hello.tgz
-rwxr-xr-x 1 s1234567 ct212-apr07  290 May 11 23:40 index.html
[s1234567@labsupport public_html]$ ls -l cgi-bin
total 4
-rwxr-xr-x 1 s1234567 ct212-apr07 86 May 11 23:37 hello.cgi
[s1234567@labsupport public_html]$ █
```

Open the Web browser on your PC and visit your own personal website at http://labsupport.no-ip.org/~s1234567. You should see the following output.

[1] This is necessary because `index.html` utilizes the *server-side include* feature.


Lab 1.8 CGI Test - Windows Internet Explorer
http://labsupport.no-ip.org/~s1234567/
File Edit View Favorites Tools Help
Lab 1.8 CGI Test
Page Tools
Done
Internet
100%

Hello, this is s1234567 !

Lab 1.8 Test(with CGI)

Figure 1.8.1

The above webpage is displayed because you have created a default webpage, `index.html`, in your homepage directory. If your username is displayed correctly before the exclamation mark in the above webpage, then the CGI is working properly on your personal website. You can now proceed to the next step.

Note: `index.html` calls `hello.cgi`, which is written in Perl, to retrieve your username on the LabBook Support Server. You will meet more Perl CGI scripts in Lab 2.3.

Step 2 Downloading the program files

Some configuration files have been made ready at the LabBook Support Server to expedite the process. The files are compressed as a tarball, which is located here:

http://labsupport.no-ip.org/labs/2009/1.8/lab1.8.tar.gz

Use the `wget` command to download the tarball into your home directory `~/public_html`.

```
[s1234567@labsupport ~]$ cd ~/public_html
[s1234567@labsupport public_html]$ wget http://labsupport.no-
ip.org/labs/2009/1.8/lab1.8.tar.gz
--2009-04-16 15:28:10--  http://labsupport.no-ip.org/labs/2009/1.8/lab1.8.tar.gz
Resolving labsupport.no-ip.org... 203.80.251.50
Connecting to labsupport.no-ip.org|203.80.251.50|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 848 [application/x-gzip]
Saving to: 'lab1.8.tar.gz.1'

100%[======================================>] 848         --.-K/s   in 0s

2009-04-16 15:28:10 (70.3 MB/s) - 'lab1.8.tar.gz.1' saved [848/848]

[s1234567@labsupport public_html]$ █
```

Then use the command `tar -zxpvf lab1.8.tar.gz` to extract the source files from the tarball.

```
[s1234567@labsupport public_html]$ tar -zxpvf lab1.8.tar.gz
error_lab_1.8.html
lab1.8/
lab1.8/index.php
lab1.8/.htaccess
lab1.8/.myhtpasswd
error_lab_1_8.html
[s1234567@labsupport public_html]$ █
```

Use the commands `ls -l` and `ls -la lab1.8` to verify whether the directory `lab1.8` and other files have been extracted properly in your account. In particular, pay attention to their permission rights to see if they match the output shown below.

```
[s1234567@labsupport public_html]$ ls -l
total 5464
-rw-r--r-- 1 s1234567 s1234567     252 2009-04-16 15:20 error_lab_1_8.html
drwx--x--x 2 s1234567 s1234567    4096 2009-04-16 15:01 lab1.8
-rw------- 1 s1234567 s1234567     806 2009-04-16 15:45 lab1.8.tar.gz
[s1234567@labsupport public_html]$ ls -la lab1.8
total 20
drwx--x--x 2 s1234567 s1234567 4096 2009-04-16 15:01 .
drwxr-xr-x 6 s1234567 s1234567 4096 2009-04-16 15:46 ..
-rw-r--r-- 1 s1234567 s1234567  223 2009-04-16 15:01 .htaccess
-rw-r--r-- 1 s1234567 s1234567  344 2009-04-16 15:00 index.php
-rw-r--r-- 1 s1234567 s1234567   24 2009-04-16 14:57 .myhtpasswd
[s1234567@labsupport public_html]$ █
```

If they don't match, use the `chmod` command to rectify them, e.g. `chmod 711 <directory name>`, `chmod 744 <file name>`.

Step 3 Creating a user database

We are going to control access to the newly created directory `~/public_html/lab1.8`. Only those users who can present a valid username and password will be given access. Setting up such user authentication takes two steps: you create a file containing the usernames and passwords (we will do this here in Step 3); and you tell the server what resources are to be protected and which users are allowed (after entering a valid username and password) to access them (we will do this in Step 4).

Among the files extracted in Step 2, the file `.myhtpasswd` under the directory `lab1.8` is for storing usernames and passwords. You can inspect its contents using `cat .myhtpasswd`.

```
[s1234567@labsupport public_html]$ cd lab1.8
[s1234567@labsupport lab1.8]$ cat .myhtpasswd
labbook09:z0DfvZixLqGLY
```

For the last line of the above listing, the first field (the characters before the colon) represents the username while the second field (those after the colon) represents the user's encrypted password. So, the username is `labbook09` and the password is `z0DfvZixLqGLY`. To add more user information to the file, you can use the following command:

```
htpasswd -b .myhtpasswd <username> <password>
```

For example, to add the user s1234567 with the password apr09, we can do as follows.

```
[s1234567@labsupport lab1.8]$ htpasswd -b .myhtpasswd s1234567 apr09
Adding password for user s1234567
[s1234567@labsupport lab1.8]$ cat .myhtpasswd
labbook09:z0DfvZixLqGLY
s1234567:UfaUHrEsDBZTk
[s1234567@labsupport lab1.8]$ █
```

You may add some user information to your copy of the database file `.myhtpasswd` using the above method.

Step 4 Configuring the `.htaccess` file

Now you need to tell the server what resources are to be protected and which users are allowed to access them. We should create an `.htaccess` file and write into it the directives that notify the Web server how the protection should be carried out. A sample `.htaccess` file has is included among the files extracted in Step 2 and placed under the directory `~/public_html/lab1.8`. Use the `cat` command to inspect its content as follows.

```
[s1234567@labsupport lab1.8]$ cat .htaccess
AuthType Basic
AuthName "CT212 Lab 1.8 - Protecting a web directory"
AuthUserFile /home/s1234567/public_html/lab1.8/.myhtpasswd

<LIMIT GET POST>
require valid-user
</LIMIT>
ErrorDocument 401 /~s1234567/error_lab_1_8.html
[s1234567@labsupport lab1.8]$ █
```

Let's explain the directives in the file. You have to modify some of them using vi or any other text editor in order to make them work for you:

a The `AuthType` directive tells the server what protocol is to be used for authentication. At the moment, `Basic` is the only method available.

b The `AuthName` directive specifies a realm name for this protection. Once a user has entered a valid username and password, any other resources within the same realm name can be accessed with the same username and password. This can be used to create two areas which share the same username and password.

c The `AuthUserFile` directive tells the server the location of the user database file. So, you have to modify this directory by replacing `s1234567` with your user account login ID.

d The `require` directive tells the server which usernames from the user database file are valid for access. The argument `valid-user` after the `require` directive specifies that any username in the file can be used. But it could be configured to allow only certain users in (e.g. `require user martin jane`).

e The `ErrorDocument` directive tells the server what to do if a user can't go through the authentication successfully. If the 401 status (i.e. unauthorized access) is detected, the Web server will redirect the user to the HTML page `/~s1234567/error_lab_1_8.html`, which should contain some information telling the user what has happened. Again, you have to replace `s1234567` with your user account login ID. The file `error_lab_1_8.html` is among the files extracted in Step 2 and should exist in your `~/public_html` directory.

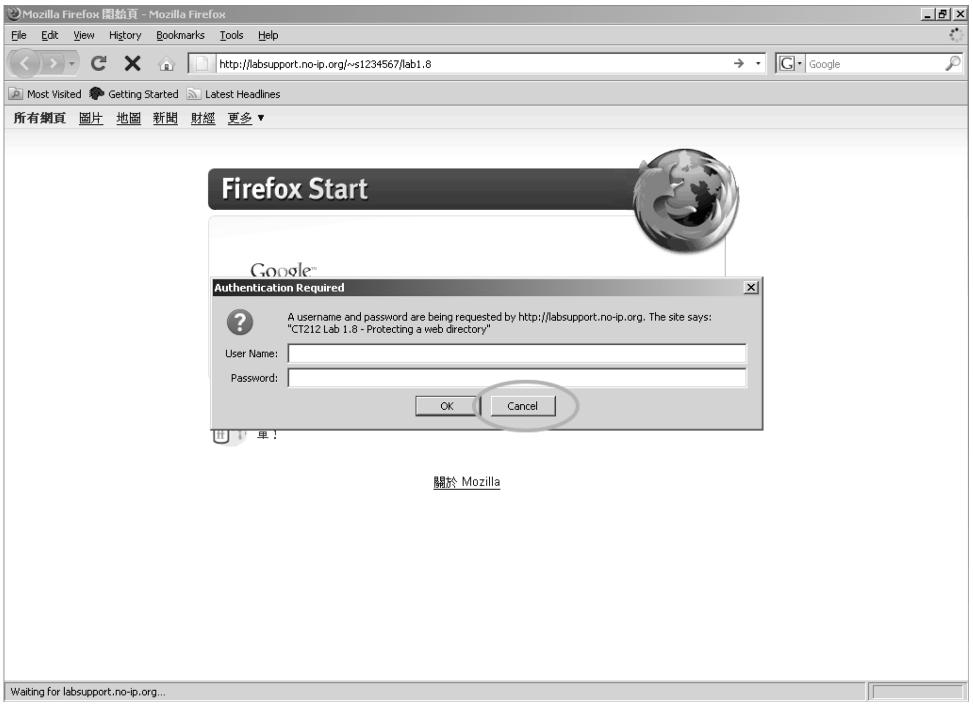
Before going further, double-check if the changes specified in points (c) and (e) above have been made.

Step 5 Verifying user authentication

Now, you can verify user authentication by browsing the following URL:

http://labsupport.no-ip.org/~<<your login id>>/lab1.8

A dialog box will appear asking you for the username and password.


Mozilla Firefox 開始頁 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://labsupport.no-ip.org/~s1234567/lab1.8
Google
Most Visited Getting Started Latest Headlines
所有網頁 圖片 地圖 新聞 財經 更多 ▾
Firefox Start
Google
關於 Mozilla
Waiting for labsupport.no-ip.org...

Authentication Required

A username and password are being requested by http://labsupport.no-ip.org. The site says: "CT212 Lab 1.8 - Protecting a web directory"

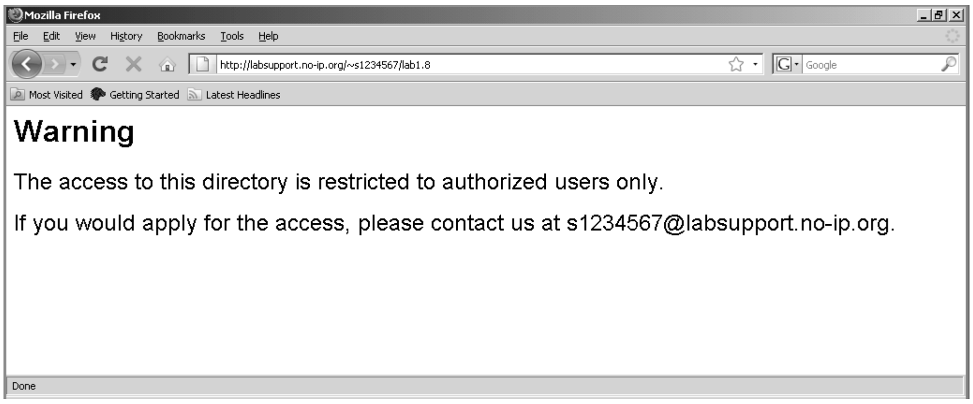
User Name:

Password:

OK Cancel

Figure 1.8.2

If you cancel the authentication process by pressing the **Cancel** button, error status 401 (i.e. unauthorized access) will arise and the Web server will direct you to the error page `error_lab_1_8.html`, according to the directive in the `.htaccess` file we made previously.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://labsupport.no-ip.org/~s1234567/lab1.8
Google
Most Visited Getting Started Latest Headlines
Done

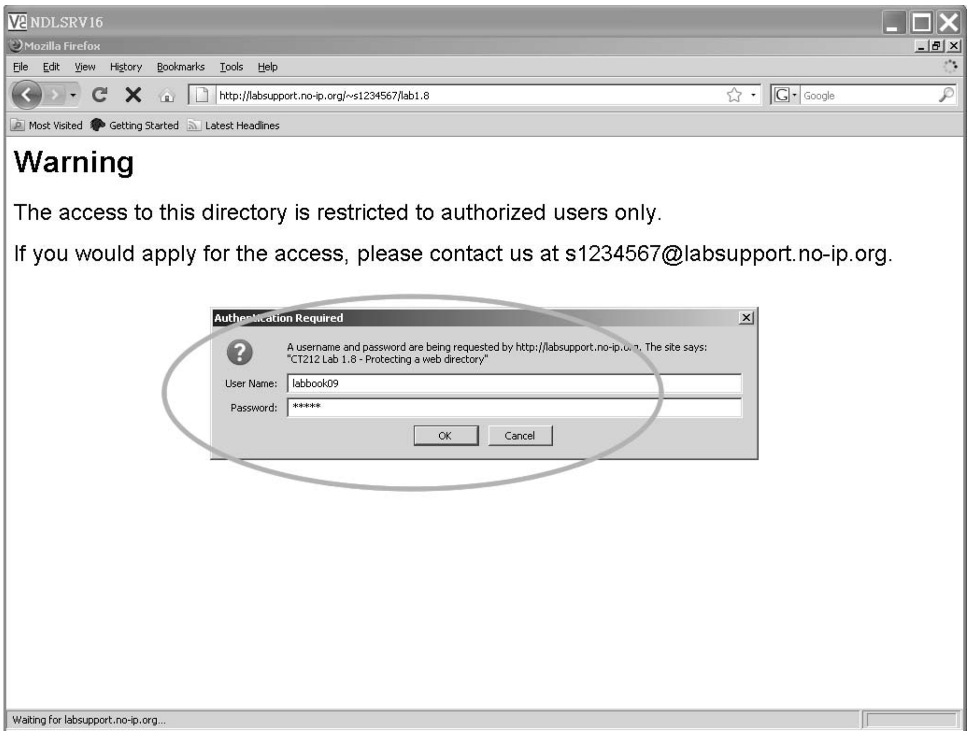
Warning

The access to this directory is restricted to authorized users only.

If you would apply for the access, please contact us at s1234567@labsupport.no-ip.org.

Figure 1.8.3

Now, press the refresh button of your browser to reload the URL. This time, input *labbook09* and *lab1.8* as the username and password respectively, or any other username and password you have added to the user database in Step 3, to get through the authentication.


NDLSRV16
Mozilla Firefox
File Edit View History Bookmarks Tools Help
Most Visited Getting Started Latest Headlines
Google

http://labsupport.no-ip.org/~s1234567/lab1.8

Warning

The access to this directory is restricted to authorized users only.

If you would apply for the access, please contact us at s1234567@labsupport.no-ip.org.

Authentication Required

A username and password are being requested by http://labsupport.no-ip.org. The site says: "CT212 Lab 1.8 - Protecting a web directory"

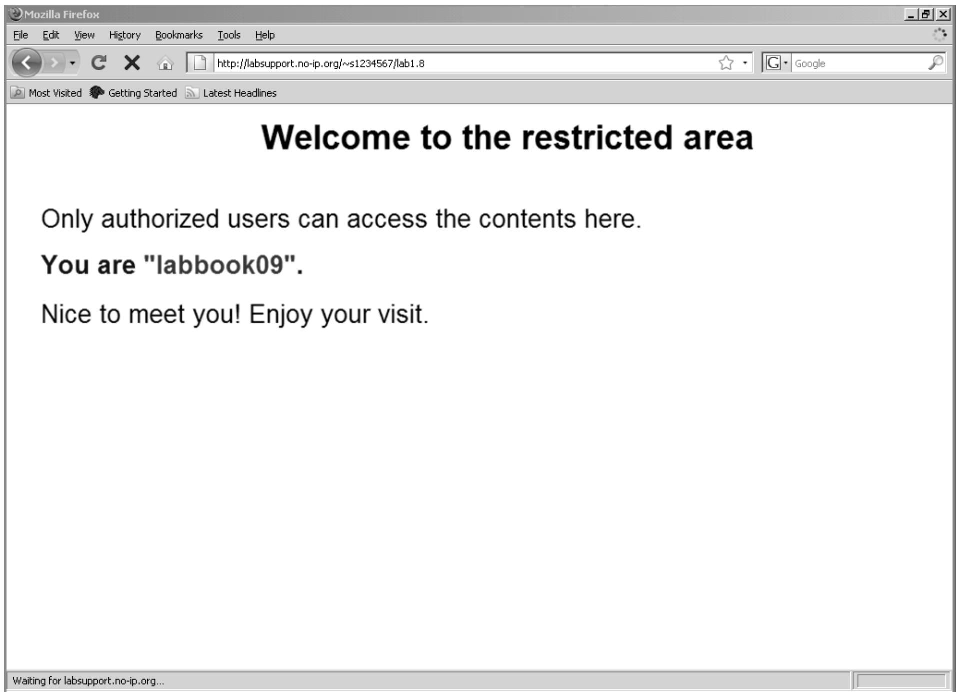
User Name: labbook09

Password: *****

Waiting for labsupport.no-ip.org...

Figure 1.8.4

If authentication is successful, you will see the following greeting page.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
Most Visited Getting Started Latest Headlines
Google

http://labsupport.no-ip.org/~s1234567/lab1.8

Welcome to the restricted area

Only authorized users can access the contents here.

You are "labbook09".

Nice to meet you! Enjoy your visit.

Waiting for labsupport.no-ip.org...

Figure 1.8.5

You have successfully implemented password protection for your website.

Summary

In this lab, you learned how to implement password protection for a directory under a website. The approach implemented involves the use of the `.htaccess` file of the Apache server. The `.htaccess` file contains a number of directives specifying how the Web server should protect the directory. We have gone through the key directives in this lab. The implementation is practical since it is straightforward and effective. In fact, there are quite a few websites on the Internet that use this approach to protect their content.

Further reading and useful resources

- 'Getting started with CGI programming in C' — http://www.cs.tut.fi/~jkorpela/forms/cgic.html (for more about CGI programming in C)

Questions and exercises

1. In the `.htaccess` file, what do the directives `AuthUserFile` and `require` mean?
2. Suggest one way to further improve the password protection mechanism implemented in this lab.
3. Edit the error page in Figure 1.8.3 (`error_lab_1_8.html`), which is composed of simple HTML, to replace the contact email s1234567@labsupport.no-ip.org with your own email.