

CHAPTER 20

In this chapter:

- Access Control Strategies
- Controlling Access with Apache
- Controlling Access with Microsoft IIS

Controlling Access to Your Web Content

Organizations run web servers because they are an easy way to distribute information to people on the Internet. But sometimes you don't want to distribute your information to *everybody*. For instance, you might have:

- Information on your web server intended only for employees of your organization
- An electronic publication that contains articles that are only available to customers who have paid a monthly subscription fee.
- Confidential technical information that is only for customers who have signed nondisclosure agreements
- A web-based interface to your order-entry system that is open to preauthorized users, but should not be open to the general public

These scenarios have different access control requirements. Fortunately, today's web servers have a variety of ways to restrict access to information.

Access Control Strategies

There are a number of techniques that can be used to control access to web-based information:

- Restricting access by using URLs that are "secret"—that is, URLs that are hidden and unpublished
- Restricting access to a particular group of computers based on those computers' hostnames or Internet addresses
- Restricting access to a particular group of users based on their identity

Most web servers can use these techniques to restrict access to HTML pages, CGI scripts, and API-invoking files. These techniques can be used alone or in combination. You can also add additional access control mechanisms to your own CGI and API programs.

Hidden URLs

The easiest way to restrict access to information and services is by storing the HTML files and CGI scripts in hidden locations on your web server.

For example, when Simson's daughter Sonia was born, he wanted to quickly put some photographs of her on the World Wide Web so that his friends and family could see them, but he didn't want to "publish" them so that anybody could look at them. Unfortunately, he didn't have the time to give usernames and passwords to the people he wanted to see the pictures. So Simson simply created a directory on his web server named *http://simson.vineyard.net/sonia/* and put the photographs inside. Then he sent the URL to his parents, his in-laws, and a few other networked friends.

Hidden URLs are about as secure as a key underneath your door mat. Nobody can access the data unless they know where to look; then they have access to all that they want. Furthermore, this information is transitive. You might tell John about the URL, and John might tell Eileen, and Eileen might post it to a mailing list of her thousand closest friends. Somebody might put a link to the URL on another web page—or even register the hidden URL with a web search engine.

Indeed, search engines such as Lycos, AltaVista, and Google pose a special problem for hidden URLs. Most search engines "spider" the Web by retrieving a page, indexing its content, analyzing the page for links, and then repeating the process with every page that is referenced by a link. If you have no links to a "secret" page, the search engines will generally not find it. However, if there is a single link to your page's hidden URL on any other page that is indexed by a search engine, it is likely that your hidden URL will be indexed as well. This can happen even if the page that linked to your hidden URL is later deleted; the links can still be active in the search engine's databanks. We've found lots of interesting and "hidden" pages by searching with keywords such as *secret*, *confidential*, *proprietary*, and so forth.

In general, avoid using secret URLs if you really care about maintaining the confidential nature of your page.



If you are a user on an Internet service provider, a hidden URL gives you a simple way to get limited access control for your information. However, if you want true password protection, you might try creating a *.htaccess* file (described later in this chapter in the section, "Controlling Access with Apache") and seeing what happens.

Host-Based Restrictions

Most web servers allow you to restrict access to particular directories from specific computers located on the Internet. You can specify these computers by their IP addresses or by their DNS hostnames.

Restricting access to IP-specific addresses or a range of IP addresses on a subnet is a relatively simple technique for limiting access to web-based information. This technique works well for an organization that has its own internal network and wishes to restrict access to people on that network. For example, you might have a network that has the IP addresses 204.17.195.1 through 204.17.195.254; by configuring your web server so that certain directories are accessible only to computers on network 204.17.195, you prevent outsiders from accessing information in those directories. This is a practical technique for many organizations that use Net 10 (10.0.0.0 through 10.255.255.255) behind their firewalls.

RFC 1918 reserves three blocks of IP address space for private addressing. These addresses are shown in Table 20-1.

Table 20-1. Private IP address space designated by RFC 1918

Range	Prefix notation	# of Hosts
10.0.0.0–10.255.255.255	10/8	16,777,214
172.16.0.0–172.31.255.255	172.16/12	10,48,574
192.168.0.0–192.168.255.255	192.168/16	65,534

According to RFC 1918:

An enterprise that decides to use IP addresses out of the address space defined in RFC 1918 can do so without any coordination with IANA or an Internet registry. The address space can thus be used by many enterprises. Addresses within this private address space will only be unique within the enterprise, or the set of enterprises which choose to cooperate over this space so they may communicate with each other in their own private internet.

Because private addresses have no global meaning, routing information about private networks shall not be propagated on inter-enterprise links, and packets with private source or destination addresses should not be forwarded across such links. Routers in networks not using private address space, especially those of Internet service providers, are expected to be configured to reject (filter out) routing information about private networks. If such a router receives such information the rejection shall not be treated as a routing protocol error.

Instead of specifying computers by IP address, most web servers allow you to restrict access on the basis of DNS domains. For example, your company may have the domain *company.com* and you may configure your web server so any computer that has a name of the form **.company.com* can access your web server. Specifying client access based on DNS domain names has the advantage that you can change your IP addresses and you don't have to change your web server's configuration file as well. (Of course, you will have to change your DNS server's configuration files, but you would have to change those anyway.)

The advantage of host-based restrictions is that they are largely transparent to users. If a user is working from a host that is authorized and she clicks on a URL that

points to a restricted directory, she sees the directory. If the user is working from a host that is not authorized and she clicks on the URL that points to a restricted directory, the user sees a standard message that indicates that the information may not be viewed. A typical message is shown in Figure 20-1.

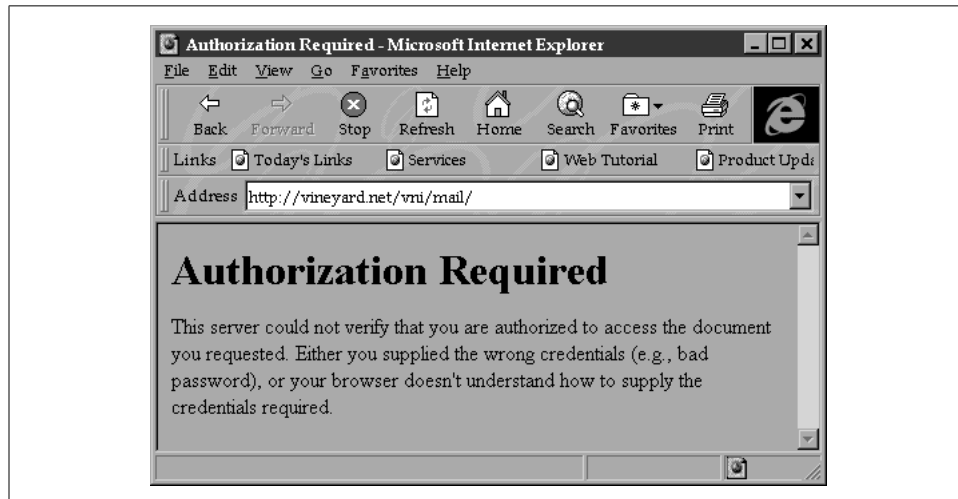


Figure 20-1. Access denied



Although the standard Domain Name Service protocol is subject to spoofing, security can be dramatically increased by the use of public key encryption as specified in the DNSSEC protocol (described in Chapter 4). Implementations of DNSSEC are now available from a variety of sources, including <ftp://ftp.isc.org/>. To improve the overall security of the Internet's Domain Name Service, DNSSEC should be deployed as rapidly as possible.

Using firewalls to implement host-based access control

You can also implement host-based restrictions using a firewall to block incoming HTTP connections to particular web servers that should only be used by people inside your organization. Such a network is illustrated in Figure 20-2.

Caveats with host-based access control

Host-based addressing is not foolproof:

- IP spoofing can be used to transmit IP packets that appear to come from a different computer from the one they actually do come from. This is not a risk for static content such as HTML files, since the server will be unable to send the response back to the attacker. However, spoofed IP packets are a concern for programs executed by web servers (e.g., CGI and ASP scripts).

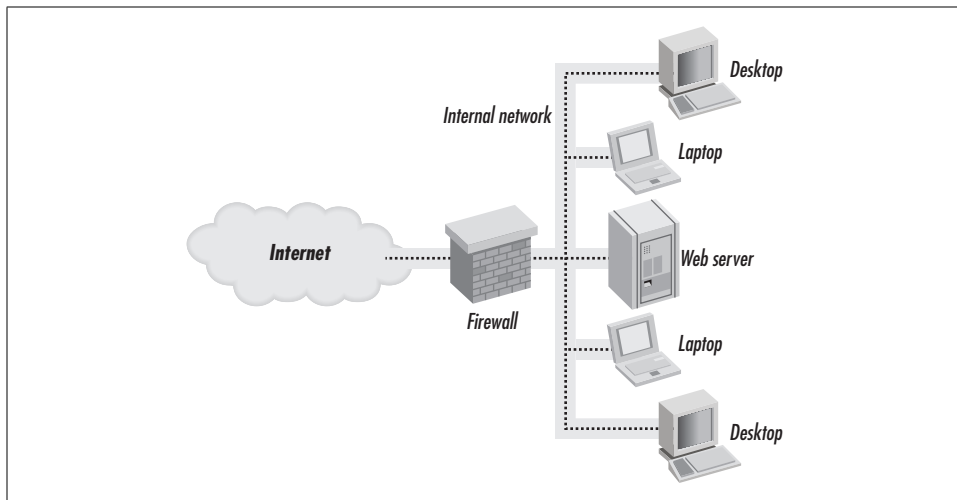


Figure 20-2. Using a firewall to implement host-based restrictions; access to the internal web server is blocked by the firewall.

- Host-based access control will not protect information from attackers who enter an organization’s network using a remote-access system such as Carbon Copy, Windows NT RAS, VPNs, or IP tunneling. In these cases, the attacker’s computer will appear to be behind your firewall, even though the attacker’s computer may actually be located elsewhere.
- Host-based addressing that is based on DNS names requires that you have a secure DNS server. Otherwise, an attacker could simply add his own computer to your DNS domain, and thereby gain access to the confidential files on your web server.

Identity-Based Access Controls

Restricting access to your web server based on usernames is one of the most effective ways of controlling access. Each user is given a username and a password. The username identifies the person who wishes to access the web server, and the password authenticates the person.

When a user attempts to reference an access-controlled part of a web site, the web server requires the web browser to provide a username and password. The web browser recognizes this request and displays a request, such as the one shown in Figure 20-3.

Because passwords are easily shared or forgotten, many organizations are looking for alternatives to them. One approach is to use public key technology. Another approach is to give authorized users a physical token, such as a smart card, which they must have to gain access. Most of these systems merely require that the users

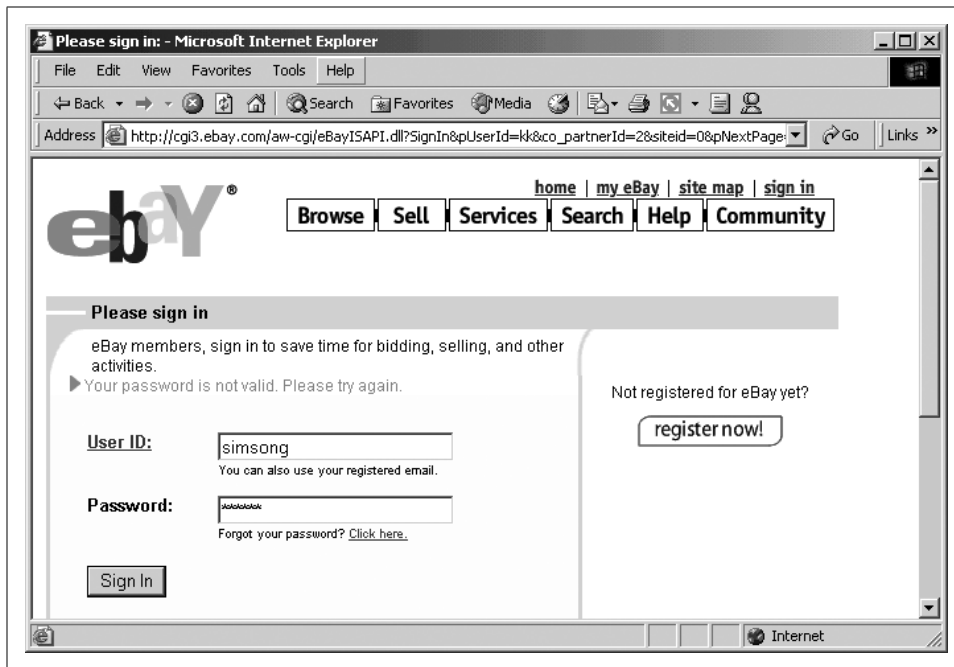


Figure 20-3. Prompt for user's password

enter their normal username and a different form of password. For example, users of the RSA Security SecurID card enter a password that is displayed on their smart cards; the password changes every minute.

One of the advantages of user-based access controls over host-based controls is that authorized users can access your web server from anywhere on the Internet. A sales force that is based around the country or around the world can use Internet service providers to access the corporate web site, rather than having to place long distance calls to the home office. Or you might have a sales person click into your company's web site from a high-speed network connection while visiting a client.

User-based access can also be implemented through the use of cookies (see "Understanding Cookies" in Chapter 8).

Controlling Access with Apache

One of the most common ways to restrict access to web-based information is to protect it using usernames and passwords. Although different servers support many different ways of password-protecting web information, one of the most common techniques is with the <Limit> server configuration directive present in the Apache web server.

Using `<Limit>`, you can control which files on your web server can be accessed and by whom. The Apache server gives you two locations where you can place your access control information:

- You can place the restrictions for any given directory (and all of its subdirectories) in a special file located in that directory. Traditionally the name of this file is `.htaccess`, although you can change the name in the server's configuration file.
- Alternatively, you can place all of the access control restrictions in a single configuration file. The Apache server allows you to place access control information in the server's single `httpd.conf` file.

Whether you choose to use many access files or a single file is up to you. It is certainly more convenient to have a file in each directory. It also makes it easier to move directories within your web server, as you do not need to update the master access control file. Furthermore, you do not need to restart your server whenever you make a change to the access control list—the server will notice that there is a new `.htaccess` file and behave appropriately.

On the other hand, having an access file in each directory means there are more files that you need to check to see whether the directories are protected. There is also the possibility that an attacker will be able to convince your web server (or another server running on the same computer) to fetch this file directly. Although the attacker's possession of the `.htaccess` file doesn't ruin your system's security, it gives an attacker additional information that might be used to find other holes.

Enforcing Access Control Restrictions with the `.htaccess` File

Here is a simple file that restricts access to registered users whose usernames appear in the file `/ws/adm/users`:

```
% cat .htaccess
AuthType Basic
AuthName Web Solutions
AuthUserFile /ws/adm/users

<Limit GET POST>
require valid-user
</Limit>
%
```

The `.htaccess` file consists of two parts. At the beginning of the file is a set of commands that allow you to specify the authorization parameters for the given directory. The second half of the file contains a `<Limit . . . > . . . </Limit>` block containing security parameters that are enforced for the HTTP GET and POST commands.

The `.htaccess` file needs to be placed in the directory on the web server that you wish to protect. For example, if your web server is named `www.ex.com` and has a document

root of `/usr/local/etc/httpd/htdocs`, naming this file in the directory `/usr/local/etc/httpd/htdocs/internal/.htaccess` would restrict all information prefixed by the URL `http://www.ex.com/internal/` so that it could only be accessed by authorized users.

Enforcing Access Control Restrictions with the Web Server's Configuration File

The access restrictions described in the `.htaccess` file can also be placed in the configuration file of the Apache web server. In this case, the commands would be enclosed within a pair of `<Directory directoryname>` and `</Directory>` tags. The `directoryname` parameter should be the directory's full pathname and not the directory within the web server's document root or the directory name as referenced by a symbolic link. For example:

```
...
<Directory /usr/local/etc/httpd/htdocs/internal>
AuthType Basic
AuthName Web Solutions
AuthUserFile /ws/adm/users

<Limit GET POST>
require valid-user
</Limit>
</Directory>
...
```

The format of the user account files (`/ws/adm/users` in this example) is similar to the Unix password file, but only contains usernames and encrypted passwords. It is described in detail later in this chapter in the section “<Limit> Examples.”

Commands Before the <Limit>... </Limit> Directive

The following commands can be placed before the `<Limit>... </Limit>` block of most web servers:

AllowOverride what

Specifies which directives can be overridden with directory-based access files. This command is only used for access information placed in system-wide configuration files such as `conf/access.conf` or `conf/httpd.conf`.

AuthName name

Sets the name of the Authorization Realm for the directory. The name of the realm is displayed by the web browser when it asks for a username and password. It is also used by the web browser to cache usernames and passwords.

AuthRealm realm

Sets the name of the Authorization Realm for the directory; this command is used instead of `AuthName` by older web servers.

AuthType type

Specifies the type of authentication used by the server. Most web servers only support “basic,” which is standard usernames and passwords.

AuthUserFile absolute_pathname

Specifies the pathname of the *httpd* password file. This password file is created and maintained with a special password program; in the case of the Apache web server, use the *htpasswd* program. Note that the web server’s password file is not stored in the same format as */etc/passwd*. The format is described in “Manually Setting Up Web Users and Passwords” later in this chapter.

AuthGroupFile absolute_pathname

Specifies the pathname of the *httpd* group file. This group file is a regular text file. Note that this file is not in the format of the Unix */etc/group* file. Instead, each line begins with a group name and a colon and then lists the members, separating the member names with spaces. For example:

```
stooges: larry moe curley
staff: sascha wendy ian
```

Limit methods to limit

Begins a section that lists the limitations on the directory. For more information on the *Limit* section, see the next section of this chapter.

Options opt1 opt2 opt3 . . .

Turns on or off individual options within a particular directory. Options available are listed in the following table.

Option	Meaning
ExecCGI	Allows CGI scripts to be executed within this directory.
FollowSymLinks	Allows the web server to follow symbolic links within this directory.
Includes	Allows server-side include files.
Indexes	Allows automatic indexing of the directory if an index file (such as <i>index.html</i>) is not present.
IncludesNoExec	Allows server-side includes, but disables CGI scripts in the includes.
SymLinksIfOwnerMatch	Allows symbolic links to be followed only if the target of the file or the directory containing the target file matches the owner of the link.
All	Turns on all options.
None	Turns off all options.

Commands Within the <Limit> . . . </Limit> Block

The <Limit> directive is the heart of the Apache access control system. It is used to specify the actual hosts and/or the users that are to be allowed or to be denied access to the directory.

The format of the <Limit> directive is straightforward:

```
<Limit HTTP commands>  
directives  
</Limit>
```

Normally, you will want to limit both GET and POST commands.

The following directives may be present within a <Limit> block:

order options

Specifies the order in which allow and deny statements are evaluated. Specify “order deny,allow” to cause the deny entries to be evaluated first; servers that match both the “deny” and “allow” lists are allowed.

Specify “allow,deny” to check the allow entries first; servers that match both are denied.

Specify “mutual-failure” to cause hosts on the allow list to be allowed, those on the deny list to be denied, and all others to be denied.

allow from host1 host2 ...

Specifies hosts that are allowed access.

deny from host1 host2 ...

Specifies hosts that are denied access.

require user user1 user2 user

Only the specified users “user1, user2, and user3 . . .” are granted access.

require group group1 group2 ...

Any user who is in one of the specified groups may be granted access.

require valid-user

Any user that is listed in the *AuthUserFile* will be granted access.

Hosts in the *allow* and *deny* statements may be any of the following:

- A domain name, such as *.vineyard.net* (note the leading “.” character)
- A fully qualified hostname such as *nc.vineyard.net*
- An IP address such as *204.17.195.100*
- A partial IP address such as *204.17.195*, which matches any host on that subnet
- The keyword “all”, which matches all hosts

<Limit> Examples

If you wish to restrict access to a directory’s files to everyone on the 204.17.195 subnet, you could add the following lines to your *access.conf* file:

```
<Directory /usr/local/etc/httpd/htdocs/special>  
<Limit GET POST>  
order deny,allow
```

```
deny from all
allow from 204.17.195
</Limit>
</Directory>
```

If you then wish to allow only the authenticated users *wendy* and *sascha* to access the files, and only when they are on subnet 204.17.195, you could add these lines:

```
AuthType Basic
AuthName The-T-Directory
AuthUserFile /etc/web/auth
<Limit GET POST>
order deny,allow
deny from all
allow from 204.17.195
require user sascha wendy
</Limit>
```

If you wish to allow the users *wendy* and *sascha* to access the files from anywhere on the Internet, provided they type the correct username and password, try this:

```
AuthType Basic
AuthName The-T-Directory
AuthUserFile /etc/web/auth
<Limit GET POST>
require user sascha wendy
</Limit>
```

If you wish to allow any registered user to access files on your system in a given directory, place this *.htaccess* file in that directory:

```
AuthType Basic
AuthName The-T-Group
AuthUserFile /etc/web/auth
<Limit GET POST>
require valid-user
</Limit>
```



After modifying your *.htaccess* file, test it by attempting to access the information in the protected directory with both a valid account and an invalid account.

Manually Setting Up Web Users and Passwords

To use authenticated users, you need to create a password file. You can do this with the *htpasswd* program, using the “-c” option to create the file. For example:

```
# ./htpasswd -c /usr/local/etc/httpd/pw/auth sascha
Adding password for sascha.
New password: deus333
Re-type new password: deus333
#
```

You can add additional users and passwords with the *htpasswd* program. When you add additional users, do *not* use the “-c” option, or you will erase all of the users who are currently in the file:

```
# ./htpasswd /usr/local/etc/httpd/pw/auth wendy
Adding password for wendy.
New password:excom22
Re-type new password:excom22
#
```

The password file is similar, but not identical, to the standard */etc/passwd* file:

```
# cat /usr/local/etc/httpd/pw/auth
sascha:ZdZ2f8M0eVcNY
wendy:ukJTIFYWHKwtA
#
```

Because the web server uses *crypt*-style passwords, it is important that the password file be inaccessible to normal users on the server (and to users over the Web) to prevent an ambitious attacker from trying to guess passwords using a program like *Crack*.

Advanced User Management

If you need to manage more than a few users, you will want to implement a more sophisticated user management system.

Use a database

Instead of storing users, passwords, and groups in a single file, you can store them in a database such as MySQL. The Apache web server can then be programmed to refer to this database to determine whether users are valid. The user authentication database can be on the same computer as the web server, or it can be located on a central database server.

Use RADIUS or LDAP

RADIUS is a remote authentication protocol originally developed by Livingston and now used widely throughout the Internet industry. Radius provides for centralized username/password management. To use Radius, your web server is configured to validate username/password authentication requests with a remote RADIUS server.

LDAP is a general-purpose directory system that is increasingly being used for remote authentication. As with RADIUS, if you configure your web server to use LDAP for authentication, username/password pairs are sent to the remote LDAP server for validation. LDAP offers considerably more flexibility than RADIUS, such as the ability to manage groups and more easily handle authentication tokens. However, unlike RADIUS, LDAP sends usernames and passwords without encryption. If you wish to use an LDAP server, you need to protect usernames and passwords that are sent by either having them sent on a segregated network or having the transmissions encrypted by using LDAP over SSL.

Use PKI and digital certificates

Instead of using a username and password to authenticate a user, you can use a digital certificate that is stored on the user's hard disk.

To make use of digital certificates, a web site user must first create a public key and a secret key. The public key is then signed by a certification authority, which returns to the user a certificate that consists of the user's public key, a distinguished name (DN), and the certification authority's signature. You then configure your web server so that it will allow access to any user who has a valid certificate.

The advantage of using PKI and digital certificates is that you do not need to distribute valid user accounts to the web server—you only need to distribute the public key for the certification authority and a list of revoked certificates. This isn't a great advantage when you are running a single web server, but it can be tremendously advantageous when you are running hundreds or thousands of web servers.

For further information on digital certificates, see Chapter 21.

Controlling Access with Microsoft IIS

Microsoft's Internet Information Services (IIS) is a web service that is shipped as part of the Windows NT 4, 2000, and XP operating systems. It is a full-featured web server that does just about anything that you could possibly want (other than run on Unix, that is).

Installing IIS

To install IIS, follow these steps:

1. Open the "Add/Remove Programs" control panel.
2. Select "Add/Remove Windows Components."
3. Check "Internet Information Services."
4. Click "Next."

IIS installs the following directories on your system:

\inetpub

Root directory for your web server

\inetpub\wwwroot

Root document directory for the web server

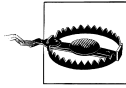
\systemroot\Help\iisHelp

Help files

\systemroot\system32\inetsrv

Program files

The directories containing user content will remain on your system after you completely uninstall IIS.



As soon as you install IIS, be sure that you go to the Microsoft Windows Update web site and download all relevant patches for IIS before you start the server. If possible, download the patches from behind a firewall. If you fail to install the IIS patches, your Windows server will almost certainly be broken into and compromised. This is true even if you are behind a corporate firewall or are otherwise “protected.”

Downloading and Installing the IIS Patches

To install the patches, follow these steps:

1. Log into your Windows system using an account that has Administrator access.
2. Using Microsoft’s Internet Explorer, open the URL *http://windowsupdate.microsoft.com/*. On most installations, you can easily open this URL by picking “Windows Update” from the Start menu.
3. You may be prompted as to whether or not you wish to run an ActiveX component that is signed by Microsoft Corporation. You must run this ActiveX applet in order to run the Windows Update feature.
4. Windows Update will search your system and identify which patches need to be updated. In all likelihood, you will need to install the Critical Updates Package. If a new service pack has come out, you will be advised to install that as well.
5. Select the updates you wish to download and click “Download” to download the software.
6. You will be prompted to accept the Supplemental Microsoft End User License Agreement (“Supplemental EULA”). Read the license agreement. You will note that “The entire risk arising out of use or performance of the OS components and any support services remains with you.” You will also note that Microsoft disclaims all liability arising from its software for whatever damage that the software may cause, “even if Microsoft . . . has been advised of the possibility of such damages.”
7. Click “Yes” to accept the Supplemental EULA.
8. Windows Update will download and install the necessary updates. In all likelihood, your computer will need to be rebooted when the updates are installed.

Controlling Access to IIS Web Pages

After your computer reboots, IIS will be running. If you go to the URL *http://localhost/*, you will be prompted to enter a valid username and password. Enter a username and password for a local administrative user and you will be presented with the IIS

localstart.asp page. This page will give you information on how to start up the IIS console, how to view the online documentation, and how to create a document root.

Using the IIS snap-in component to the Windows Computer Management application, you can control many aspects of the IIS web and FTP server. To run the component, select “Administrative Tools” from the “Control Panel” window. Then double-click on “Computer Management.” Expand the “Services and Applications” item in the tree (see Figure 20-4).

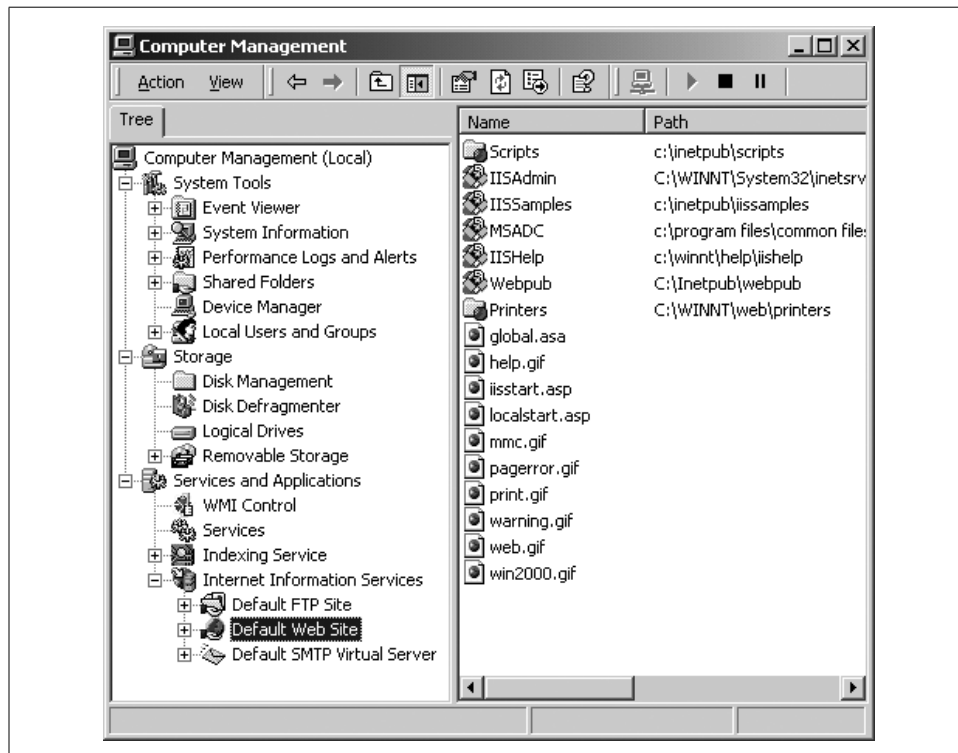


Figure 20-4. The Internet Information Services snap-in for the Windows Computer Management control panel allows you to control many aspects of IIS security.

To modify the properties of the web site, select “Default Web Site” and choose the “Properties” item from the Action menu.

Using this interface, you can enable or disable a variety of features, including:

- Whether or not accesses to the web site are logged, and if they are logged, where they are logged.
- The “home directory,” or document root, of the web server.
- Whether the directory should be indexed.
- Whether anonymous access is allowed to the directory. (Default is yes.)

- If anonymous access is allowed, the local user that is used for granting anonymous access.
- Which authentication system is used for granting access.
- Whether visitors are allowed read and/or write access to the web server. (Default is read only.)
- Whether directory browsing is allowed. (Default is no.)
- Whether access is allowed to the source code of your scripts. (Default is no.)
- Whether additional headers should be sent with the web pages.
- What the PICS rating is for this web site.

If there are subdirectories in your *wwwroot* directory, you can select them and control their access on a directory-by-directory basis.

Restricting Access to IIS Directories

It is surprisingly easy to create a directory in your IIS web server that will be restricted to a specific set of users over the Internet. Windows IIS integrates with the Windows username directory and Windows directory permissions. Web users are generally users who have local accounts on your computer. For anonymous access, the default IIS installation creates an account named *IUSR_computername* (where *computername* is the name of your computer). If the *IUSR_computername* user has access to read a directory, then the directory can be read over the Internet.

In this example, we will create a directory called *private* and allow access only to a user named *blue* with a password *blueboy*.

1. Create a directory called *private* in the directory *c:\inetpub\wwwroot*.
2. Find the directory in the Computer Management application, select it, and examine its properties. This will display a window titled “private Properties.”
3. Select the “Directory Security” tab of the “private Properties” window.
4. In the box titled “Anonymous access and authentication control,” click the button labeled “Edit . . .”.
5. Uncheck the box that says “Anonymous access.”
6. Check the box that says “Basic Authentication” (the password is sent in clear-text). If you do not check this box, IIS will only use Microsoft’s proprietary extensions to the HTTP protocol, which make Windows’ challenge-response authentication system run over HTTP. (If you do not wish to have usernames and passwords sent in the clear over the Internet, simply use URLs that begin with *https:* instead of *http:*. See Figure 20-5.)
7. Click “Yes” to indicate that you wish to use Basic Authentication.
8. Click “OK” to close the Authentication Methods window.

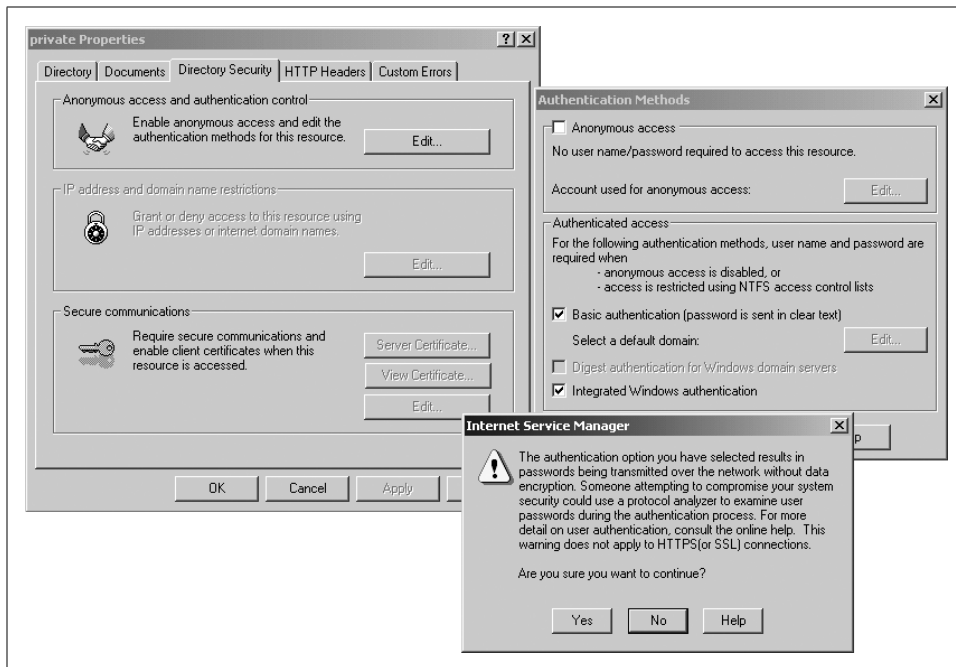


Figure 20-5. Using the Computer Management application to restrict access to a directory

9. Click “Apply” to apply your changes to the directory.
 10. In the Computer Management application, select on the “Local Users and Groups” element of the tree.
 11. Choose “New User” from the Action menu.
 12. Create a new user that has the username *blue* and the password *blueboy*.
- The user that you have created will now have access to the directory.



Be sure to configure your system so that the users you create will not have undue access to your Windows computer. Specifically, you may wish to remove these users from the Users group and place them in a specific group for World Wide Web access. You should also firewall your Windows server so that users on the Internet only have access to ports 80 and 443.

If your *wwwroot* directory is on an NTFS partition, you can use the NTFS directory permissions to control which users have access to the private directory. If your *wwwroot* directory is on a FAT32 partition, all valid users will have access to all files.

For more information on IIS security, refer to the IIS Documentation under the section “Administration → Server Administration → Security.”