CHAPTER 10

Securing Your Server

COMPUTER SECURITY IS A COMPLEX ISSUE. Examples of security breaks include unauthorized users accessing a system, base-level packet data interception, domain name hijacking, denial of service attacks, and so forth. This list grows longer as the technology evolves, and discussing every aspect of computer security could fill a book of its own.

This chapter covers only the most important issues in securing your IIS application. It first covers the security features in IIS that are fully integrated with the Windows NT and Windows 2000 Server operating systems, the types of security that are available, and how to utilize them. The discussion then proceeds to the Secure Sockets Layer (SSL) protocol, including how to obtain and install certificates to support secure transactions for your Web site.

Securing IIS on Windows NT 4.0 and Windows 2000 Servers

In Chapter 7, you can see how to password protect your Web application. This is done by creating a database table, which contains all the login names and passwords of authorized users. You will now find out how to utilize IIS security features.

Windows NT 4.0 supports New Technology File System (NTFS) 4 and FAT file system formats. Windows 2000 supports NTFS 4, NTFS 5, FAT, and FAT32. For you to fully utilize the security features in IIS, you should use NTFS 4 in Windows NT 4.0 and NTFS 5 in Windows 2000. NTFS, among other things, supports what FAT and FAT32 don't: Access Control Lists (ACLs) for directories and files. ACLs contain zero or more access control entries, each of which details a user's access rights to a specific file or directory. You will see how ACLs relate to IIS security features in the section "Anonymous Access," which is coming up.

If the operating system is currently installed on a server using the FAT or FAT32 file system, you can convert it to NTFS by running the `CONVERT` command. However, after you convert it to NTFS, you cannot convert it back to FAT or FAT32. For more information on the `CONVERT` command, look up "Convert" under Help.

With IIS 4.0 on Windows NT or IIS 5.0 on Windows 2000 Server installed on an NTFS volume, you can restrict access to any file or directory in your Web application to only those users who have been given permission. That's because NTFS always verifies that only authorized users can access a file or a directory. Different

authentication methods are available on the two operating systems. Windows NT 4.0 is examined here first, and the authentication methods in Windows 2000 Server are discussed later.

Authentication Methods in IIS 4.0 on Windows NT

IIS 4.0 has three authentication methods: Anonymous, Basic, and Windows NT Challenge/Response. When you first create a virtual directory for your Web application, Anonymous and Windows NT Challenge/Response are selected by default. However, you can change this default selection and apply one or more authentication methods to each virtual directory. To view the authentication method(s) in a virtual directory, perform the following steps:

1. Right-click your virtual directory icon in Internet Service Manager, and then click Properties.

2. Click the Directory Security tab (see Figure 10-1).

3. In the Anonymous Access and Authentication Control frame, click the Edit button.

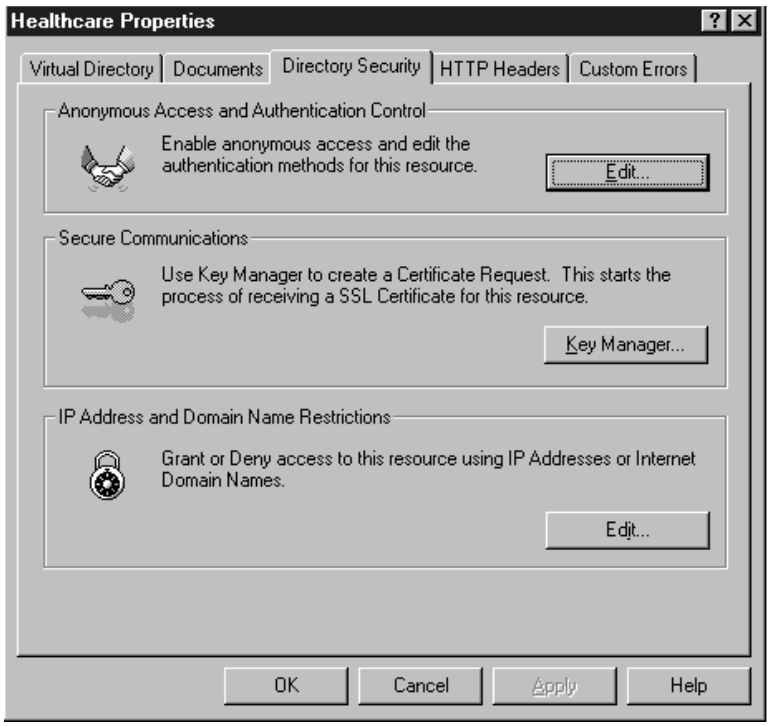

Figure 10-1. The Directory Security tab

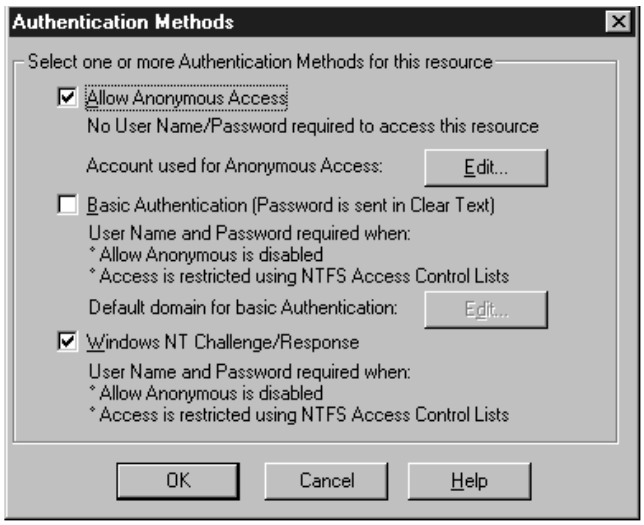

Figure 10-2. The three authentication methods

The Authentication Method dialog box appears (see Figure 10-2), showing all the authentication methods that are available. The three authentication methods are described in the following sections.

Anonymous Access

As mentioned previously, under NTFS every file and directory has a set of permissions that determines which users or groups have been granted access to specific files or directories, and which rights a user may exercise, such as read, write, or execute. You can view the ACL for a file or directory by right-clicking the file or directory in the Windows Explorer, choosing Properties, and then selecting the Permissions button on the Securities tab. The Directory or File Permissions dialog box appears.

Figure 10-3 shows the Directory Permission dialog box for viewing and editing the ACLs for a specific directory. For a file, the dialog box will read File Permissions as its title. The dialog box in Figure 10-3 displays all users and groups who have permissions to the Healthcare directory under the C:\Budi directory. It also shows the types of access each user or group has in the Type of Access list box. The access types are No Access, List, Read, Add, Add & Read, Change, Full Control, Special Directory Access, and Special File Access.

So then, how do your Internet users, whose identities are unknown to the server, have access? Internet users don't have user accounts, do they? When you install IIS, a special account is created for IIS to use. By default, all your Web applications in that server will include this special account in the users' ACLs. When

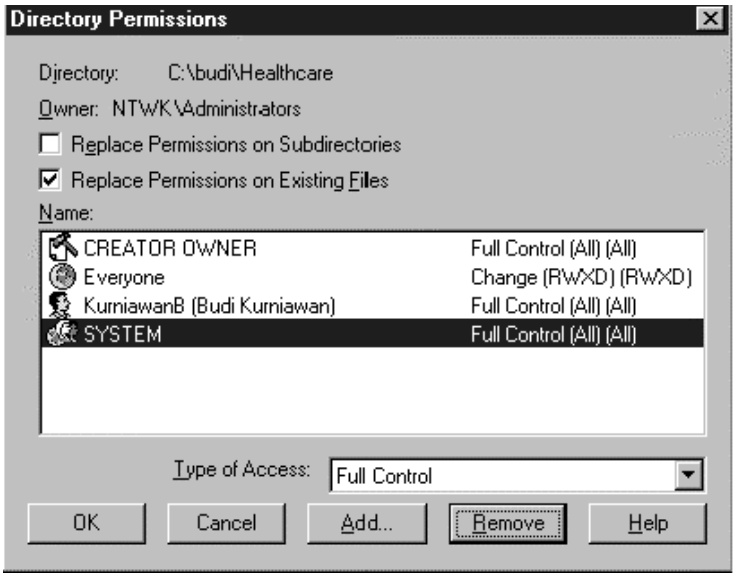

Figure 10-3. The Directory Permission dialog box for viewing and editing the ACLs for a specific directory. For a file, the dialog box will read File Permissions as its title.

an anonymous user needs access to a certain page, IIS employs this special user account to access that page on a user's behalf. You could say that IIS is impersonating the user.

The special user account for IIS is IUSR_*ComputerName*, where *ComputerName* is the name of the server. If your server name is Kingkong, then the name created for IIS is IUSR_Kingkong. Whenever someone unknown to the server requests a page through IIS, IIS will use this name.

By default, this user account has Read permission on the virtual directory. Unless the Allow Anonymous Access checkbox in the Authentication Method dialog box is cleared, IIS will first try to use this method when a user requests a page in the virtual directory.

IIS will try to use the password entered in the Internet Service Manager. If you change this special account's password without changing the one in the Internet Service Manager, the two passwords for the same user will be out of sync. As a result, user authentication using this method will fail. How do you prevent these passwords from being out of sync? By default, IIS keep these passwords in sync. However, you can check them from the Internet Information Manager by using the following steps:

1. From the Internet Information Manager, right-click your virtual directory, and then click Properties.

Figure 10-4. Keeping the passwords in sync

2. Click the Directory Security tab.

3. Click the Edit button. The Authentication Method dialog box appears.

4. Click the Edit button at the top. The Anonymous User Account dialog box appears.

5. Make sure that the Enable Automatic Password Synchronization check-box is selected, as in Figure 10-4.

If you remove the Read permission of the IUSR_*ComputerName* account for a file, an anonymous user will no longer be able to access that file. Removing the Read permission is normally done when you only want specific users to have access to your Web site. If an anonymous user attempts to view a page without the required permission, IIS will try to use other methods of authentication. If neither the Basic nor Challenge/Response method is selected, IIS will send a message to the Web browser telling the user that he or she doesn't have permission to view the requested page.

Basic Authentication

Enabling the Basic and/or Windows NT Response/Challenge authentication methods is only appropriate for a corporate or members-only Web site, where it is important to validate all users accessing the site. If your Web site falls into one of these categories, you must ensure that IIS can't use the IUSR_*ComputerName* account to access the restricted files anonymously. You should remove the Everyone group and the IUSR_*ComputerName* from the access list.

The Basic Authentication method is a widely used, industry-standard method. When you use Basic Authentication, the user name and password that

the user types in are sent to the Web server in Base64 encoded form. It is somewhat like sending the user name and password in plain text.

The Basic Authentication method transmits passwords over the network without data encryption. Someone attempting to attack your system security could use a protocol analyzer to examine user passwords during the authentication process. Therefore, Basic Authentication is *not* a very secure method for transmitting user names and passwords. The user name and password are Base64 encoded, which can be decoded easily even by novice hackers, although they still need to access your network and use a TCP/IP packet sniffer to intercept network packets.

The user name and password that the user types into the browser, which get sent to IIS, are used by IIS to execute a "Log on locally" command to the IIS machine. Because IIS then has the login name and password of that user, it can use them to access the requested resource.

Despite its lack of encryption, Basic Authentication should be used in one particular instance: when you need to authenticate users using a browser other than Internet Explorer. For example, Netscape Navigator understands only the Basic Authentication method because this method is the industry standard. If you have Challenge/Response and Basic Authentication selected, Internet Explorer always uses Windows NT Challenge/Response, and Navigator always chooses Basic Authentication. If you have sensitive data, this is a serious security concern.

Windows NT Challenge/Response

Windows NT Challenge/Response is a very secure way to determine who is making a request. The process flow of Challenge/Response is mandatory knowledge for anyone working on IIS.

Windows NT Challenge/Response does not send a password across the network because passwords can be intercepted and deciphered. Windows NT uses a nonreversible algorithm analogous to a meat grinder. You put something in, and out comes a hash. Windows NT uses the Internet standard MD4 hashing algorithm to produce a 16-byte (128-bit) hash. It is theoretically impossible to take both the hash and the algorithm and mathematically reverse the process to determine the password. In other words, the password serves as a "private key." Only someone who has the key can generate a particular hash.

A Windows NT domain controller has a database of user hashes generated from user passwords, but doesn't store the passwords themselves. (Note that this separation of hash and passwords does *not* necessarily make the domain controller less of a target for hackers because sometimes a hash can be used as a password equivalent.)

IIS 4.0 on a Windows NT 4.0 Server will try to use Challenge/Response authentication if all the following are true:

- Allow Anonymous in the Internet Service Manager under WWW Properties is not selected. The IUSR account does not then have sufficient permissions to access the requested resource.

- Windows NT Challenge/Response is selected in the Internet Service Manager.

- The browser making the request understands Challenge/Response (currently, Internet Explorer is the only browser that supports Challenge/Response).

If both Basic and Challenge/Response authentication methods are selected, how does IIS know which authentication method to use? It is fairly simple. IIS sends an "HTTP 401 Access Denied" message back to the browser with a list of authentication methods it accepts, saying in effect: "Hey, this is an exclusive club. Before you can get in, you must first identify yourself. These are the identification methods I accept."

Internet Explorer will always attempt to use Challenge/Response, and other browsers will use Basic.

Authentication Methods in IIS 5.0 on Windows 2000 Server

If you have installed IIS 5.0 on Windows 2000 Server, you have four authentication methods to use: Anonymous, Basic, Digest, and Integrated Windows. The Anonymous and Basic methods are the same as those in IIS 4.0, which were discussed previously in this chapter.

Digest Authentication

The Digest Authentication method is a new feature in IIS 5.0 and is available only on domains with a Windows 2000 domain controller. This method operates much like Basic Authentication except that passwords are sent as hash values. A hash value is a number derived from a text message, from which it is not feasible to decipher the original text.

Digest Authentication proceeds as follows:

1. IIS sends certain information that will be used in the authentication process to the browser.

2. The browser adds this information to its user name and password plus some other information and performs hashing on it.

3. The resulting hash is sent over the network to the server along with the additional information in clear text.

4. The server adds the additional information to a plain text copy it has of the client's password and hashes all of the information.

5. The server then compares the hash value it received with the one it just produced.

6. Access is granted only if the two numbers are identical.

Because the Digest Authentication method is a new feature in HTTP 1.1, not all browsers support it. If a noncompliant browser makes a request on a server that requires this method, the server will reject the request and send an error message to the client.

Integrated Windows Authentication

The Integrated Windows Authentication method is the enhanced version of the previous Windows NT Challenge/Response Authentication method. The Integrated Windows Authentication method can use both the Kerberos v5 authentication protocol and its own challenge/response protocol. If Directory Services is installed on the server and the requesting browser is compatible with the Kerberos v5 authentication protocol, both the Kerberos v5 and the challenge/response protocol are used; otherwise only the challenge/response protocol is used.

The Kerberos v5 authentication protocol is a feature of the Windows 2000 Distributed Services architecture. In order for Kerberos v5 authentication to be successful, both the client and the server must have a trusted connection to a Key Distribution Center (KDC) and the Directory Services must be compatible. More information on this protocol can be found in Windows 2000 Help.

The Integrated Windows Authentication method works like this:

1. Unlike Basic Authentication, it does not initially prompt users for a user name and password. The current Windows user information on the client computer is used instead. You can configure Internet Explorer version 4.0 or later to initially prompt for user information if you wish.

2. If the authentication exchange initially fails to identify the user, the browser will prompt the user for a Windows user account user name and password.

3. Internet Explorer will continue to prompt the user until the user enters a valid user name and password, or it will close the prompt dialog box.

Even though this method is secure, it is only supported by Internet Explorer version 2.0 or later. Integrated Windows Authentication does not work over an HTTP Proxy connection.

Restricting Access from a Specific IP Address

With IIS 4.0 and 5.0, you can prevent certain computers, groups of computers, or entire networks from accessing your Web site. For example, if your intranet server is connected to the Internet and you don't want a particular site to be accessed by Internet users, you can configure IIS so it only grants access to users from your network. Or, if you know that a malicious person is trying to attack your Web site, you can configure IIS to deny that person access.

When this feature is enabled in IIS, IIS checks the IP address of the user's computer against the server's IP address restriction settings.

To deny access to computers, groups of computers, or domains, perform the following steps:

1. In the Internet Service Manager in IIS 4.0 or IIS snap-in in IIS 5.0, right-click a Web site, directory, or file, and select Properties to open its property sheets.

2. Select the appropriate Directory Security or File Security property sheet. Under IP Address and Domain Name Restrictions, click Edit.

3. In the IP Address and Domain Name Restrictions dialog box, select the Granted Access option. When this option is selected, you grant access to all computers and domains except those that you specifically deny access to.

4. Click Add.

5. In the Deny Access On dialog box, select Single Computer, Group of Computers, or Domain Name options.

6. You can click the DNS Lookup button to search for computers or domains by name, rather than by IP address. IIS will search on the current domain for the computer, and if found, will enter its IP address in the IP address text box.

7. Click OK to close both dialog boxes.

Note that a user accessing your Web server through a Proxy server will appear to have the IP address of the Proxy server.

To grant access to computers, groups of computers, or domains, perform the following steps:

1. In the Internet Service Manager in IIS 4.0 or IIS snap-in in IIS 5.0, right-click a Web site, directory, or file, and select Properties to open its property sheets.

2. Select the appropriate Directory Security or File Security property sheet. Under IP Address and Domain Name Restrictions, click Edit.

3. In the IP Address and Domain Name Restrictions dialog box, select the Denied Access option. When this option is selected, you deny access to all computers and domains except those that you specifically grant access to.

4. Click Add.

5. In the Grant Access On dialog box, select Single Computer, Group of Computers, or Domain Name options.

6. You can click the DNS Lookup button to search for computers or domains by name, rather than by IP address. Type in a name.

7. Click OK to close both dialog boxes.

In addition to configuring IIS, you can also prevent certain computers from accessing your Web site through your code. The `REMOTE_ADDR` server environment variable of the Request object's ServerVariables collection returns the IP address of the client host making the request. Using this server environment variable, you can restrict access by any computer you don't intend to allow into your system.

```
If Request.ServerVariables("REMOTE_ADDR") = "1.2.3.4" Then
   ( Reject access here
  .
  .
  .

End If
```

Using SSL for Encryption, Authentication, and Data Integrity

Now that you know how to ensure that your Web site is accessed only by authorized users, you need to be aware of another type of attack. Like telephone lines

that can be tapped, information transferred from a Web browser to a server, and vice versa, can be stolen. When the information is sensitive, you will want to take measures to protect your data.

For encrypting purposes, you don't have much choice other than the Secure Sockets Layer (SSL) protocol. Originally developed by Netscape, SSL is a protocol for transmitting information securely. SSL is the only method that works with the majority of current browsers. SSL uses encryption to guarantee confidentiality and also solves authentication and data integrity issues.

Encryption

SSL encrypts information as it passes back and forth between the Web server and the browser. SSL comes with 40-bit key and 128-bit key encryptions. Of course, the more bits you use, the stronger the encryption is. However, more bits also means more work for the server and browser to encrypt and decrypt the message being passed back and forth.

To prevent the server performance from deteriorating significantly when using SSL, you should apply encryption only to files that will contain and receive sensitive data. Also, keep the files free of elements, such as images, that can be resource hungry. To ease maintenance, you should also create a special directory for these files.

Authentication

SSL also provides something very important to Internet users: authentication. Authentication protects users against any impostor who claims to be someone he or she isn't.

For example, if you go to a Web site to buy books, you might feel confident that it is safe to type in your credit card details. However, it is possible for a clever thief to create a Web site that is indistinguishable from the online bookstore's in order to steal a user's credit card information. To prevent this, you can use SSL for authentication.

Data integrity

When using SSL, whenever a message is sent, the sending and receiving computers each generate code based on the message content. If even a single character in the message content is altered en route, the receiving computer will generate different code, and then alert the recipient that the message is not legitimate. With data integrity, both parties involved in the transaction know that what they're seeing is exactly what the other party sent.

Using Asymmetric Technology for Encryption and Decryption

When encoding and decoding messages, you need an encryption algorithm and a key. The key is used in conjunction with the algorithm to convert the message into scrambled cipher text. Some techniques, such as DES, RC2, and RC5, are known as symmetric key technology because the algorithms use the same key for encrypting and decrypting the message.

Other methods, such as the one implemented in SSL, use two keys—the public key and the private key. The two keys are mathematically related, but it is not feasible to deduce one without knowing the other. The public key is made available to anyone who requires it, but the private key is kept private, known only to the party generating the key pair. If the public key is used for encryption, the private key is used for decryption, and vice versa. Encryption algorithms that use different keys for encryption and decryption are known as *asymmetric* or *public key* technology.

When an Internet user wants to purchase a product from a Web site, the communication between the Web browser and the Web server should happen over a secure channel. This process should happen automatically behind the scenes, making it transparent to the user.

The following example should give you a good idea of the process with a secure site utilizing SSL. Consider the scenario in which Ken, an Internet user, is ready to type in his credit card details to purchase a book from Amazon.com (or another bookseller on the Web). Note that this is not *exactly* what happens when using SSL, however, this should give you a general idea of how the public and private key pair works.

1. The browser Ken is using asks the Web server at Amazon for the server's public key.

2. The server sends its public key to the user. Public keys are freely available to anyone who requests them.

3. The browser uses the server's public key to encrypt the data it is about to send.

4. The data is sent across the network. Even though someone can tap the network and copy the data, that person can't read it because he does not have the server's private key.

5. The server uses its own private key to decrypt the data. The server also knows if the data received is valid because if the data is corrupt, the private key will fail to decrypt it.

Using Digital Certificates for Authentication

Note that the process I described in the previous section resolves only the confidentiality and data integrity issues. What if a malicious person hijacks Amazon's domain name, so that every browser request for `http://www.amazon.com` does not go to the real site, but instead goes somewhere else? Ken's sensitive data can then be obtained without Ken's knowledge or permission. This is a problem involving authentication. Ken's browser should also authenticate that the server is the one it claims to be.

The authentication issue is resolved by using *digital certificates*. Digital certificates, or just *certificates,* are documents that provide authentication of persons and entities on a network or on the Internet. Proper use of certificates makes it impossible for malicious users to intercept a message or falsify their identities. A certificate contains the following information:

- The certificate issuer's name
- The entity for whom the certificate is being issued
- The public key of the entity for whom the certificate is being issued
- Some time stamps specifying the validation date of the certificate

Certificates can only be issued by a trusted party. This trusted party, whose job is to verify the identity of any person or organization who applies for a certificate, is called a certificate authority (CA). Examples of CAs are VeriSign, Inc. (`http://www.verisign.com`), Thawte Consulting (`http://www.thawte.com`, which has now has been acquired by VeriSign), and GTE CyberTrust Solutions (`http://www.cybertrust.gte.com`).

An Example of Enabling Your Web Site to Use SSL

The entire process of enabling your Web site to use SSL, from generating the public key and the private key pair to using certificates in an e-commerce Web site, is explained in the following step-by-step process in systems utilizing SSL. Consider the scenario of a fictitious company called SoapOnline, which sells soaps on the Internet.

1. SoapOnline generates its own public/private key pair.
2. The public key is sent to a trusted CA.

3. The CA verifies that it is indeed SoapOnline who made the request.

4. Once satisfied that the requester is legitimate, the certificate authority uses its own private key to encrypt SoapOnline's public key, along with some other data such as an expiration date, serial number, name, and so on. This is used as a certificate. The current industry standard is a X.509 v3 certificate.

5. The CA sends the certificate back to SoapOnline.

6. SoapOnline installs the certificate on their Web server. This makes SoapOnline's encrypted public key available to all Internet users who want to purchase soaps from SoapOnline. SoapOnline is now ready for secure e-commerce.

7. When a user is ready to send the sensitive data, such as credit card details, the user's browser will request SoapOnline's public key. This process happens behind the scene, without the user noticing it.

8. SoapOnline's Web server sends its public key to the user's browser. Remember that the public key has been encrypted by the CA using the CA's private key (in Step 4). The browser can use SoapOnline's public key to encrypt the data it's about to send *only if* the browser has the CA's public key.

   Indeed, most browsers include the public keys of a number of common trusted CA's. This situation is similar to someone having a key (in this case the CA's public key) and given a keyhole (an encrypted public key claimed to be SoapOnline's). If the key and the keyhole match (if the public key can be used to decrypt SoapOnline's encrypted public key), then the keyhole is a genuine one (it is really SoapOnline). This step is important to prove the identity of SoapOnline. If the browser manages to obtain a public key after using the CA's public key, then it is indeed SoapOnline's Web site it's been communicating with, and not that of an impostor who has hijacked SoapOnline's domain name.

9. Once satisfied with the authenticity of SoapOnline, the browser uses SoapOnline's public key to encrypt the data it is about to send.

10. The data is sent across the Internet.

11. SoapOnline uses its own private key to decrypt the data.

Configuring Your Server to Use SSL

Configuring your server to use SSL involves three steps. First, you must generate a Certificate Signing Request (CSR) file and an encryption key pair file using Microsoft Key Manager. Second, apply for a server certificate at a third-party certificate authority by sending the CA the certificate request file you generate. Third, after you receive your server certificate, you must install it, again using Microsoft Key Manager.

Generating a Certificate Signing Request (CSR) File

To generate a CSR file, follow these steps:

1. Open the Internet Service Manager or IIS snap-in.
2. Right-click the Web site for which you want to request the certificate, and select Properties. The Properties dialog box appears.
3. Click the Directory Security tab.
4. Click the Key Manager icon. The Key Manager dialog box appears.
5. Click Create New Key from the Key menu. A series of dialog boxes appears. Fill in the text boxes in this series of dialog boxes. The first in the series is shown in Figure 10-5.

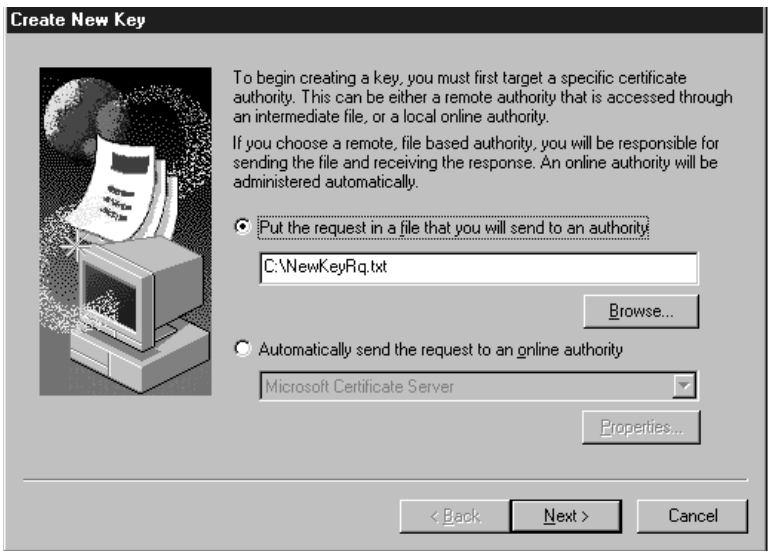

Figure 10-5. Starting the process of generating a key pair

6. Select Put The Request In A File That You Will Send To An Authority.

Subsequent dialog boxes prompt you for the following information:

- Key Name
- Password
- Bit Length: By default, it is 512.
- Organization
- Organizational Unit
- Common Name
- Country
- State/Province
- City/Locality
- Your Name
- Email Address
- Phone Number

Fill in the Key Name field with the name for the key and the Password field with the password that you will use later when installing the certificate. The remainder of the fields should be self-explanatory.

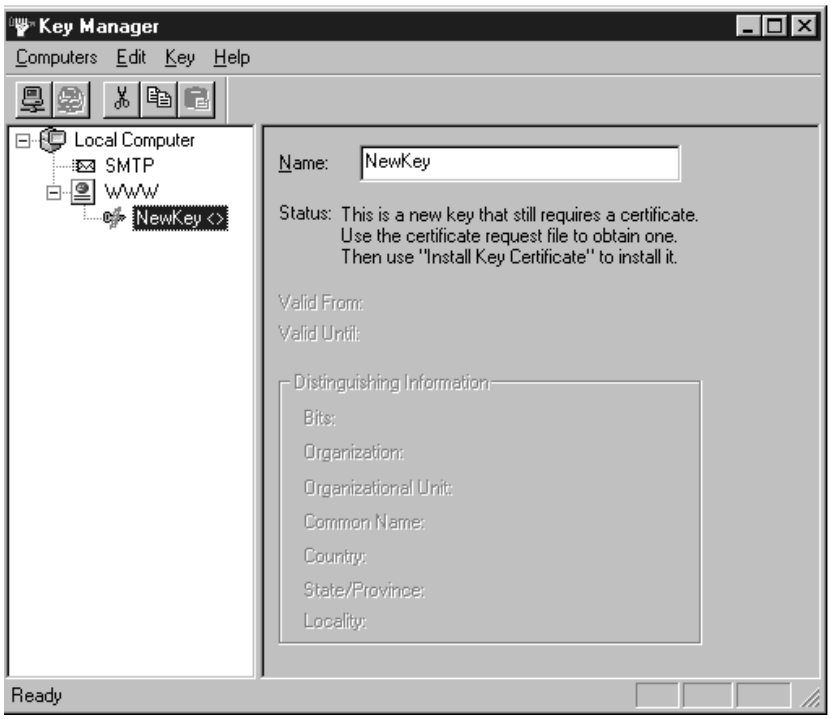

Figure 10-6. The key pair has been generated but still requires a certificate

After you supply this information, a certificate request file will be saved to your hard drive. A broken key will appear in the Key Manager dialog box, stating that a certificate request file has been generated but has not been installed, as shown in the Figure 10-6.

If you open the text file, you will find the following lines:

```
Webmaster: ceo@labsale.com
Phone: 02 90000000
Server: Microsoft Key Manager for IIS Version 4.0

Common-name: www.labsale.com
Organization Unit: Sales
Organization: Labsale
Locality: Sydney
State: NSW
Country: AU

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBCjCBtQIBADBQMQswCQYDVQQGEwJBVTEMMAoGA1UECBMDTlNXMQ8wDQYDVQQH
EwZTeWRuZXkxCjAIBgNVBAoTAWExCjAIBgNVBAsTAWExCjAIBgNVBAMTAWEwXDAN
BgkqhkiG9w0BAQEFAANLADBIAkEAhIcX93TeyS2wcioV4+WCrmD7CZ+7N9kws52t
ujXgnw+f8X9CKgp4J1SEC17t+gruMMnepNDy45I29n3UuRIH3QIDAQABoAAwDQYJ
KoZIhvcNAQEEBQADQQAZXl7ciVoa8CKHpH39XqMdGtMix/DPlFG9IkUBJdVcdPv4
7Hu1WspwCfOkhDYi93YnS3K2pPbl6lFsIEdDmhQu
-----END NEW CERTIFICATE REQUEST-----
```

Applying for a Certificate

The easiest way to obtain a digital certificate is go to a Web site for one of the certificate authorities and apply online (see "Using Digital Certificates for Authentication" earlier in this chapter). After your information is verified and the application is approved, you will receive an e-mail that tells you how to obtain your new server certificate.

For more secure systems, you may want to use the 128-bit SSL, rather than the 40-bit version. However, not everyone can obtain the 128-bit SSL. The United States government used to classify the 128-bit SSL as munitions, therefore products such as software utilizing the 128-bit SSL were not permitted to be exported outside the U.S., even though the restriction has been relaxed as of January 2000.

Organizations obtaining a special license from the U.S. government can now export products that use 128-bit SSL to any nongovernment entity and to any commercial government owned entity (except those that produce munitions) in

any country except Afghanistan (Taliban-controlled areas), Cuba, Iran, Iraq, Libya, North Korea, Serbia (except Kosovo), Sudan, and Syria. Organizations in those countries can only use the 40-bit SSL.

If you want to get some experience with certificates before making a purchase, CAs will normally give you a free trial for a limited time. Head for one of their Web sites, fill in your identity details, and copy and paste your Certificate Signing Request (CSR) into the page provided. Usually, once you submit your CSR, the CA's Web site will generate your trial certificate straight away. The certificate appears as a string of text. Copy and paste the certificate generated by the CA as a text file. Keep in mind that this is only for testing purposes. You can install the certificate but it will expire within a given short period of time.

You are then ready to install your server certificate.

Installing Your Server Certificate

After you obtain your server certificate from the certification authority, you can install it in your server. To do so, perform the following steps:

1. Open Microsoft Key Manager.

2. Right-click the broken key icon, and choose Install Key Certificate.

3. Browse to the certificate file you received from the Certificate Authority, and double-click the file.

4. Type in the password. This is the same password you used to generate the key.

5. Then, specify the IP address and port to use with SSL. The default port for SSL is 443. It is advisable that you use the default port. When you are finished, an icon of a completed key will appear in the Key Manager.

The server certificate is valid for only a certain period of time. To continue using SSL, you must request a new server certificate before the expiration date.

Enabling SSL in Your Server

After the server certificate is installed in the server, users can request any page from your Web site using the secure protocol.

Users should then be able to access your page using the HTTPS protocol if your application is accessible through the following link:

```
http://www.yourdomain.com/MetaDetect.asp
```

then in order to use the secure channel, the link changes to

```
https://www.yourdomain.com/MetaDetect.asp
```

If you are using a port other than 443 (the default for SSL), the port number must be present in the link, such as

```
https://www.yourdomain.com:445/MetaDetect.asp
```

The application can be accessed by using the HTTP and the HTTPS protocol. To detect whether a user request is handled on the secure port, use the `SERVER_PORT_SECURE` server environment variable of the `Request` object's `ServerVariables` collection. The value of this variable will be the string `"1"` if the secure channel is used; otherwise, it will be the string `"0"`. An example is given in Listing 10-1.

Listing 10-1

```
If Request.ServerVariables("SERVER_PORT_SECURE") = "1" Then
  ' Secure port
  .
  .

Else
  ' Insecure port
  .
  .

End If
```

In addition, if the page is requested using SSL, some server environment variables whose names are prefixed with `CERT_` will contain values related to the server and client certificates.

You may force users to use SSL. However, because requesting a page through SSL takes longer to process, it is not advisable to apply SSL to all files. Secure communication should only be used when transferring sensitive data, such as credit card details.

To enable SSL in a file or a directory, you can do the following:

1. Open the Property page for a directory or a page within the Internet Service Manager, and choose the Directory Security or the File Security tab.
2. Click the Edit button under Secure Communications, and choose Require Secure Channel When Accessing This Resource. See Figure 10-7.

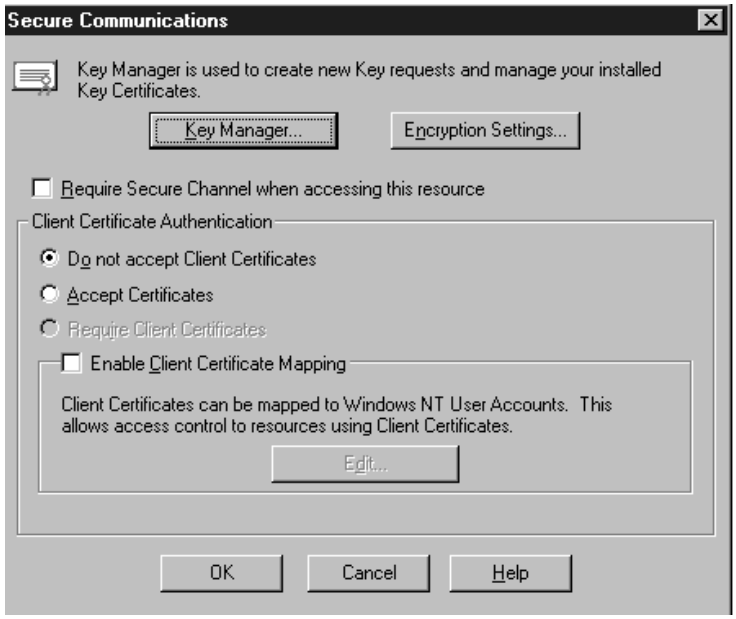

Figure 10-7. Forcing users to use SSL when requesting a page

3. If you want to require 128-bit SSL, click the button labeled Encryption Settings, and choose Require 128-Bit Encryption.

Please note that the Secure Communications dialog box will not appear unless you have a server certificate installed.

When a page is requested using SSL, in Netscape browsers the padlock in the lower left corner of the window will be closed instead of open. Netscape users can also follow these steps to see what level of encryption is protecting their transactions with your site:

1. Go to the Web site you want to check.

2. Click the Security button in the Navigator's toolbar. The Security Info dialog box indicates whether the Web site uses encryption.

3. If it does, click the Open Page Info button to display more information about the site's security features including the type of encryption used.

In Internet Explorer, a padlock icon appears in the bar at the bottom of Internet Explorer's window. Internet Explorer users can find out a Web site's encryption level by following these steps:

1. Go to the Web site you want to check.

2. Right-click on the Web site's page, and select Properties.

3. Click the Certificates button.

4. In the Fields box, select Encryption Type. The Details box shows you the level of encryption (40-bit or 128-bit).

Redirecting to a Secure Channel in the WebClass

Unlike with ASP applications, in which you normally have one page for every Web page in your site, with IIS applications you have only one ASP page. Therefore, switching to a secure channel requires a different strategy.

Where insecure communication is used, passing another hyperlink is done by using the `URLFor` function or, in an HTML Template WebItem, by connecting an element to another WebItem. When you are using the HTTPS protocol, however, you need to modify your code as in the following examples.

If you are using the `URLFor` function, pass the URL by using the following code, where the `SERVER_NAME` server environment variable is the server's host name, DNS alias, or IP address as it would appear in self-referencing URLs, and the URL server environment variable contains the base portion of the URL.

```
"https://" & Request.ServerVariables("SERVER_NAME") & _
  Left$(Request.ServerVariables("URL"), _
    InStrRev(Request.ServerVariables("URL"), "/")) & _
    URLFor(MyWebItem)
```

Because the URL server variable in the previous code also returns the ASP page name, which is also returned by the `URLFor` function, you must truncate the URL server environment variable value using the `Left$` and `InStrRev` functions to avoid double ASP page name.

Likewise, in an HTML Template WebItem, you can use the same server environment variables. The following example demonstrates how you can pass the hyperlink in the form's `ACTION` attribute:

```
<form method=post action=<WC@FormAction></WC@FormAction>
```

In order for the form's `ACTION` attribute to receive the hyperlink, you must add the code in Listing 10-2 in the `ProcessTag` event procedure of the same HTML Template WebItem, as shown next:

Listing 10-2

```
Private Sub PayForm_ProcessTag(ByVal TagName As String, TagContents As String,
SendTags As Boolean)
  If TagName = "WC@FormAction" Then
    TagContents = "https://" & Request.ServerVariables("SERVER_NAME") & _
      Left$(Request.ServerVariables("URL"), _
      InStrRev(Request.ServerVariables("URL"), "/")) & _
      URLFor(MyWebItem)
  End If
End Sub
```