Lab 10.3: Building an ISAPI Filter (EmpOnly)

Lab Overview

Objectives

After completing this lab, you will be able to:

- Use the ISAPI Extension Wizard to create an ISAPI filter project with Microsoft Visual C++.
- Implement the project, including:
  - Implementing handler member functions corresponding to the server notification events.
  - Implementing the handlers and supporting functions and classes.
- Install and test the ISAPI filter.

Scenario

One way to extend a Web site powered by a Microsoft Web server is to develop ISAPI filters. These components can flexibly extend the Web server by:

1. Intercepting notification events.
2. Performing various actions according to their purpose.
3. Altering the HTTP transaction or returning a dynamically created HTML page before relinquishing control.

Lab Setup

To develop the ISAPI component, this lab requires that you have Microsoft Visual C++ version 4.1 or later. To test this component, you must have administrative privileges on a computer running Windows NT and a Microsoft Web service, such as Internet Information Server or Peer Web Services.

This demonstration shows the solution to this lab.

Estimated time to complete this lab: **90 minutes**.

Exercises

The following exercises provide practice working with the concepts and techniques covered in this chapter.

Exercise 1: Creating the Project

In this exercise, you will use Microsoft Visual C++ to create the EmpOnly ISAPI filter project. You will select and edit various options in the ISAPI Extension Wizard dialogs, as appropriate for an employee-authentication filter.

Exercise 2: Implementing EmpOnly

In this exercise, you will implement the EmpOnly ISAPI filter. To accomplish this, you will implement event-handler member functions, and add and implement helper member functions.

Exercise 3: Testing EmpOnly

In this exercise, you will install the EmpOnly ISAPI filter onto your ISAPI-compliant Web server, and test it with Internet Explorer.

Exercise 1: Creating the Project

In this exercise, you will use Microsoft Visual C++ to create the EmpOnly ISAPI filter project. You will select and edit various options in the ISAPI Extension Wizard dialogs, as appropriate for an employee-authentication filter.

The EmpOnly filter will intercede in the Web server processing at the point where the URL is mapped to a physical resource (a file). This filter will authenticate users trying to get access to the Employees Only Web page of the Main Street Market site. If they are not on a list of authorized employees, then EmpOnly will redirect the user to a "consolation" page.

➢ **Create a new project for EmpOnly**

1. Start Microsoft Developer Studio. From the File menu, choose New.
2. In the New dialog box, select Project Workspace, and choose OK. The New Project Workspace dialog displays.
3. Supply the following information in the New Project Workspace dialog:
   - Type – Select ISAPI Extension Wizard
   - Name – Enter **EmpOnly**
   - Platform – Check Win32 (default)
   - Location – Enter or browse for \MID\Labs\Lab10.3\EmpOnly

   Then choose the Create button. The ISAPI Extension Wizard dialog, Step 1 of 1, will display.
4. In the Wizard dialog, make the following corrections and verifications:
   - The Generate A Filter Object check box should be selected, and Generate A Server Extension Object should be cleared.

   Note When the Generate A Filter Object option is selected, the ISAPI Extension Wizard adds a second dialog to its interface. This is reflected by the Wizard title, which changes to Step 1 of 2.

   - Change the Filter Description to EmpOnly ISAPI Filter.
   - Use the MFC libraries As A Shared DLL.

   When you are done, choose Next.
5. In Step 2 of 2, set the following options:
   - Medium priority level
   - Non-Secured Port Session connection type only
   - URL Mapping Requests notification type only

   Here, it is assumed that if the user is trusted enough to make a secured connection to the Web server, then there is no need for further authentication.

   When you are done, choose Finish.
6. A Summary dialog box appears. After you review the information in it, choose OK to generate the new project.

➢ **Use the Project Workspace to investigate EmpOnly**

1. Select the ClassView pane of the Project Workspace, if it is not already displayed. Expand all of the branches.

2. Double-click on the following entries to view the associated source code:
   - CEmpOnlyFilter – to view the CHttpFilter-derived class declaration. Note the declaration of the overridden notification event handler, CHttpFilter::OnUrlMap.
   - GetFilterVersion – to view the initialization member function that the Wizard generated for the ISAPI filter.
   - OnUrlMap – to view the (mostly empty) event handler generated by the Wizard.
   - theFilter – to view the single instantiation of the CHttpFilter-derived class.
3. Select the ResourceView pane. Expand all of the branches.
4. Double-click on the following entries to view the associated Windows resource.
   - String Table – has a single string resource, IDS_FILTER, the string description used by CEmpOnlyFilter::GetFilterVersion to register an extension description with the Web server. You supplied this string when you ran the ISAPI Extension Wizard in the previous section.
   - VS_VERSION_INFO – a standard editable program version resource.
5. In the FileView pane, open EmpOnly.def. Note that it only exports the two entry points that every ISAPI filter must have: the global C functions HttpFilterProc and GetFilterVersion.

➢ **Test the EmpOnly project**

1. Build the project, targeting Win32 Debug. EmpOnly should compile and link cleanly.
2. Close all the source windows, except the ones for files EmpOnly.h and EmpOnly.cpp. You will be using these files in subsequent exercises of this lab.

Exercise 2: Implementing EmpOnly

In this exercise, you will implement the EmpOnly ISAPI filter. To accomplish this, you will implement the filter's constructor and the override of the CHttpFilter::OnUrlMap event-handler member function, and add and implement helper member functions.

You will also examine the two supplemental files that the EmpOnly filter will use: Employ.dat and Consoltn.htm.

➢ **Examine the supplemental files Employ.dat and Consoltn.htm**

1. Use the Windows Explorer to locate Employ.dat and Consoltn.htm in the *Mastering Internet Development* CD-ROM directory \Labs\Lab10.3.
2. Open and examine the file Employ.dat in a text editor.

   Employ.dat is a text file that represents the database of authorized Main Street Market employees. This file contains a sample list of employee names and client machine IP addresses.
3. In the Windows Explorer, double-click on Consoltn.htm to open it in Internet Explorer.

   Consoltn.htm is the consolation page that unauthorized users will see when they try to access the Employees Only page.

➢ **Edit the CEmpOnlyFilter class declaration**

To complete the EmpOnly ISAPI filter, you will create two new helper functions and a data member. Both will be declared in a new, protected implementation section.

1. Open EmpOnly.h, if it is not already open.
2. Immediately before the end of the class declaration, add a comment denoting a section for implementation members.
3. Afterward, add a Protected access specifier.
4. In this new section, add declarations for two new helper member functions named **IsAuthUserName** and **IsAuthUserIP**. Both should have the same signature: a return value of BOOL and a single argument of type LPCSTR.
5. Add a CString data member named **strAuthUsers**.

➢ Implement the constructor for CEmpOnlyFilter

A filter's constructor is called only once. This occurs when the Web publishing service is started and it loads all ISAPI filters into memory. Therefore, the constructor is a good place to perform one-time, filter-initialization processing. EmpOnly will use it to read the employees database file into the class member strAuthUsers.

1. Locate CEmpOnlyFilter::CEmpOnlyFilter in the file EmpOnly.cpp for editing.
2. Define the following two local variables:
   - A CString named **buf**.
   - A CStdioFile object named **infile**. Create this object by opening the file Employ.dat in read-only mode. You must supply the absolute directory path to your server's \Scripts subdirectory. For example:

     ```
     c:\\Winnt\\System32\\InetSrv\\Scripts\Employ.dat
     ```

   Note that the CStdioFile constructor can cause a file exception (CFileException).

Tip If an exception is not handled by the ISAPI extension, Microsoft Web servers will handle the exception. They do so by terminating the current ISAPI process in a silent and abrupt manner. A better strategy is for you to explicitly handle all possible exceptions that your ISAPI extension can throw. For example, here the following scheme would be preferable:

1. Catch the file exception in the constructor.
2. Take appropriate remedial or diagnostic actions. For example, open and write an entry to the server's log file indicating that a file exception occurred when attempting to open Employ.dat.
3. Set the value of a dedicated class member flag (that was created for this purpose). Other member functions of the filter class can then check this flag.
4. In **CHttpFilter::GetFilterVersion**, consider returning False. This instructs the Web server to unload the filter from memory.(The filter will not receive any subsequent notifications.)

---

3. Create a **while** loop whose condition is based on the result of a read operation. Use CStdioFile::ReadString to fill the local buffer (buf) by reading a line from Employ.dat. In the body of the loop, concatenate strAuthUsers with buf and a semicolon (as a record separator). Then, empty buf to prepare it for the next read.

➢ Implement the body of the event handler CEmpOnlyFilter::OnUrlMap

This member function is responsible for the main logic of the filter. In OnUrlMap, you will determine if the request is for the Employees Only page. If it is, you will check the client name and IP address for a match in the employee database, and take the appropriate action.

1. In the body of CEmpOnlyFilter::OnUrlMap, define the following new local variables:
   - Two character arrays, **pstrName[100]** and **pstrIP[20]**, and a character pointer **pstrSearch**.
   - A DWORD named **dwSize**.
   - A Boolean variable named **bIsAuth**, initialized to FALSE.
2. Check to see if the client is requesting the Employees Only page:
   a. Prepare the pszPhysicalPath member of the HTTP_FILTER_URL_MAP structure by invoking _strlwr on it. This is necessary because the filenames in Windows are case-insensitive.
   b. Use the ANSI function strstr to determine if the file emponly.htm can be located in the pszPhysicalPath member. Use the variable pstrSearch to store the name match location.
   c. If this search failed, return SF_STATUS_REQ_NEXT_NOTIFICATION.

3. Obtain the employee's name, and check it against the employee database. Assume that all employee names are required to be at least six characters.
   a. Set dwSize to the size of the pstrName array.
   b. Call CHttpFilter::GetServerVariable to place the name of the remote user into pstrName.
   c. If the size of the employee name is greater than five, call the CEmpOnlyFilter::IsAuthUserName helper function to determine if the user is an authorized employee. Store the result in bIsAuth.
4. If the user's name was not found, check the IP address for authorization. (Assume an IP address must be at least seven characters.) If bIsAuth is FALSE, then:
   a. Set dwSize to the size of the pstrIP array.
   b. Call CHttpFilter::GetServerVariable to place the address of the remote user into pstrIP.
   c. If the size of the address is greater than six characters, call the CEmpOnlyFilter::IsAuthUserIP helper function to determine if the user is an authorized employee. Store the result in the variable bIsAuth.
5. Finally, check the value of bIsAuth to determine which resource the user will see. If this value is TRUE, return SF_STATUS_REQ_NEXT_NOTIFICATION to allow the user access to the Employees Only page (pending other filter processing).

   If bIsAuth is FALSE, then redirect the user to the consolation page:
   a. Call strcpy to copy the literal Consoltn.htm file over the value EmpOnly.htm, which is stored in pstrSearch.
   b. Return SF_STATUS_REQ_HANDLED_NOTIFICATION to indicate that no further processing of this notification event should occur.

➢ **Implement the helper functions**

For this example, the implementations of CEmpOnlyFilter::IsAuthUserName and CEmpOnlyFilter::IsAuthUserIP will be almost identical. Because interaction with the employee database has been encapsulated into the filter's constructor and the two helper functions, replacing it with a more sophisticated scheme should be straightforward.

1. Below CEmpOnlyFilter::OnUrlMap, add a comment indicating a section for implementation member functions.
2. Add code to the function header and body for these two member functions.
3. In the body of IsAuthUserName, define an integer named **hit**.
4. Invoke the member function CString::Find on CEmpOnlyFilter::strAuthUsers to see if the employee database contains the argument string. Store the returned value in hit.
5. If hit has a value of -1, return FALSE; otherwise, return TRUE.
6. Copy this implementation and paste it into CEmpOnlyFilter::IsAuthUserIP body. Ensure that the CString::Find function's argument name is correct.

Note This implementation of the EmpOnly filter is not secure. With some knowledge of employee accounts, simple IP address and user name spoofing would allow access to the Employees Only page.

Exercise 3: Testing EmpOnly

In this exercise, you will install the EmpOnly ISAPI filter onto your ISAPI-compliant Web server. To accomplish this, you will copy the filter DLL and employee database file to the \Scripts subdirectory of the Web server. Then, you will copy the consolation page to the \WWRoot subdirectory. Finally, you will edit the registry to add an entry for this filter.

You will also test the EmpOnly filter with Internet Explorer. You will try to access the Employees Only page, EmpOnly.htm, before and after you add a client entry for yourself in the employees database.

➢ Install the EmpOnly ISAPI filter on a Microsoft Web server

Installing an ISAPI filter is slightly more complicated than installing an ISAPI application. You copy the filter DLL and supporting program files into the \Scripts subdirectory of your Web server. Then, you copy the supporting Web pages to the \WWWRoot subdirectory.

In addition to these steps, you register ISAPI filters in the Windows system registration database. When a Microsoft Web server starts, it loads all of the registered ISAPI filters.

1. To stop the Web service, forcing it to unload an ISAPI application, use one of the following techniques:
   - Reboot Windows to initialize the Web server.
   - Use the Internet Service Manager to stop the Web service.
   - Use the Web-based Service Administrator to stop the Web service.
   - Use the Services applet of the Control Panel to stop the Web service.
   - From the command line, issue the command **net stop W3Svc** (Use **net start W3Svc** to restart the Web publishing service.)
2. Copy the file EmpOnly.dll to the \Scripts subdirectory of your Microsoft Web server.
3. Copy the employee database, Employ.dat, to the \Scripts subdirectory. This file can be found on the *Mastering Intenet Development* CD-ROM in the \Labs\Lab10.3 directory.
4. Copy the Consolation page, Consoltn.htm, to the server's \WWWRoot directory. This file can also be found in the \Labs\Lab10.3 directory.
5. Add an entry for EmpOnly to the Windows registration database:
   a. Start the Windows system registration database editor, RegEdit.exe.
   b. Locate the entry Filter DLLs under the key HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\W3Svc\Parameters
   c. Modify this entry's value data by appending the absolute path of the EmpOnly filter. For example:

      ```
      c:\WinNT\System32\InetSrv\Scripts\EmpOnly.dll
      ```

   If there are already filters registered, precede the new entry with a comma separator.
6. Restart the Web service, using one of the techniques listed in Step 1.

➢ Test EmpOnly with the Internet Explorer

1. Test the action of the EmpOnly ISAPI filter for an unauthorized user by entering the following URL in Internet Explorer:

   `http://<server-name>/EmpOnly.htm`

   Because your name and IP address are not in the employee database file, EmpOnly.dll redirects your request to the Consolation page, Consoltn.htm.
2. Edit the text file Employ.dat, and add a line with your full name, user name, and the IP address of your client machine.

   If you are using anonymous access, then your user name is not important. To obtain the IP address of your client machine, run IPConfig.exe or WinIPCfg.exe from the command line.
3. Stop the WWW publishing service by using one of the techniques listed in Step 1, and then restart it.

   This will force the EmpOnly filter to reread the employee database.
4. Choose Refresh in Internet Explorer to update the page. This time, you should be an authorized employee, so the Employees Only page should appear.

As an alternate method, you can use the Windows NT challenge-response logon mechanism, which is supported by Microsoft Internet Information Server and Internet Explorer. You can restrict access by setting the security permissions in the file EmpOnly.htm, however, redirection to another page would not occur for unauthorized personnel, but would only display a server error message.