

FARONICS

**ANTI-EXECUTABLE™**

**ABSOLUTE Protection from  
Unauthorized Executables**



## **Faronics Anti-Executable Enterprise - Patch Management**

---

### **TECHNICAL WHITEPAPER**

Last modified: January, 2006

### **Faronics**

Toll Free Tel: 800-943-6422

Toll Free Fax: 800-943-6488

International Tel: +1 604-637-3333

International Fax: +1 604-637-8188

**[www.faronics.com](http://www.faronics.com)**

©1999-2006 Faronics Corporation. All rights reserved.  
Deep Freeze, Anti-Executable, and WINSelect are trademarks  
and/or registered trademarks of Faronics Corporation.  
All other company and product names are trademarks of their respective owners.

---

## Introduction

A major concern for all systems administrators is maintaining the security of their workstations. With new exploits and vulnerabilities being found all the time, a proper patch management strategy is critical to ensure the health and security of workstation deployment.

Anti-Executable keeps users on task and highly productive. It standardizes the environment by preventing all unauthorized and unwanted programs from being run. However, it introduces some interesting challenges within the process of applying patches because Anti-Executable does not discriminate — it prevents any programs not authorized from running.

There are several methods for integrating Anti-Executable with Patch Management, and when properly done, users can enjoy the bulletproof reliability of an Anti-Executable protected system, and system administrators can have the peace of mind that comes from knowing their systems are fully up to date.

This white paper discusses the different methods available to update software in an Anti-Executable environment.

## Scheduled Patch Maintenance

Scheduled patch maintenance allows the administrator to specify a period of time when the client machines restart with the Anti-Executable protection disabled. During this maintenance period, Windows Updates, antivirus definition updates, and other software updates can be scheduled. Scripts can be run and batch files can be executed. This is a very common method to keep client machines up to date with the most recent patches.

Scheduled patch maintenance is very popular in lab situations. During certain times on certain days of the week, labs are not in use. A maintenance period can be scheduled to run updates during these times.

The maintenance period is configured using the Anti-Executable Configuration Administrator. This program is used to configure workstation installation files, as well as configuration files. The configuration files are used to apply the changes to deployed workstations through the Anti-Executable Enterprise Console.

Depending on the policies in place, certain updates may need to be run. Windows and antivirus updates tend to be the most frequent. The following information explains some of the update scenarios encountered and the different methods available to handle these updates.

### Scheduling Windows Updates

The following information describes the steps required in order to set up Anti-Executable to work with Windows Updates.

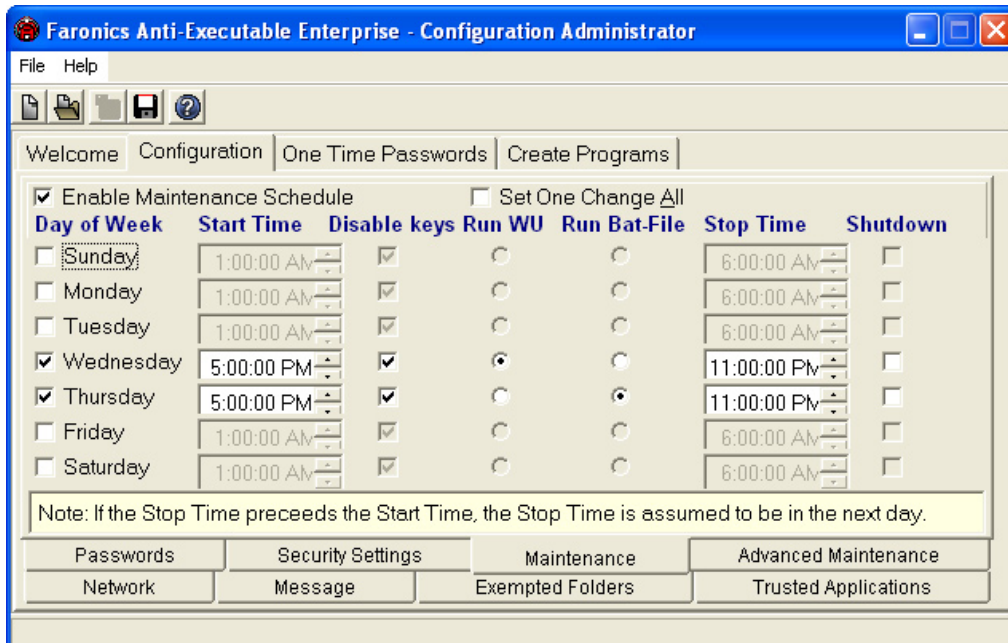
There are several different methods available to run a Windows Update in an Anti-Executable environment. Anti-Executable can be set up to start a Windows Update at a particular point in time during the maintenance period. It is also possible to configure the maintenance period to execute a batch file the maintenance period starts. Another solution would be to have another process start the updates during the maintenance period.

### Scheduling Windows Updates Through the Maintenance Option

The first method involves setting up a maintenance period using the Anti-Executable Configuration Administrator. An option is selected so Anti-Executable will run the Windows Updates after the machine goes into maintenance mode.

Complete the following steps to configure a maintenance period:

1. In the Anti-Executable Configuration Administrator, click the *Configuration* tab and click the *Maintenance* sub tab.
2. Check *Enable Maintenance Schedule*.
3. Specify days and times the maintenance mode will occur. The window should look similar to the following:



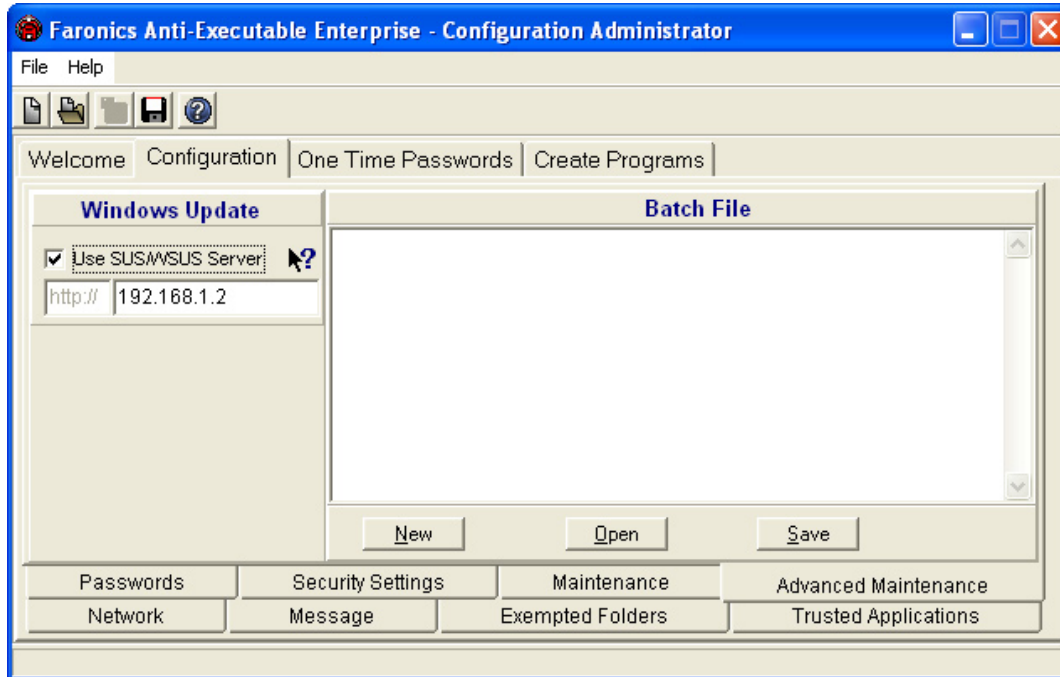
In the above screen, Wednesday has been selected for maintenance. At 5PM, Anti-Executable is disabled. At 11PM, the Anti-Executable is re-enabled. The *Disable keys* checkbox has been checked; this means the keyboard and mouse are locked while the machine is in maintenance mode. Some other options have also been selected. These options are discussed below.

Check the *Run WU* checkbox to allow Anti-Executable to start a Windows Update after it enters the maintenance period.

If a client machine is configured with this option, it would attempt to download and run updates from Microsoft's web site. If an SUS/WSUS server is available, this can be specified in the advanced settings.

If there is an SUS or WSUS server, this can be specified using the following steps:

1. Click the *Configuration* tab and the *Advanced Maintenance* sub tab.
2. Check *Use SUS/WSUS Server* and enter the IP address of the server. The screen should look similar to the following:



The client machine with these settings would attempt to download and run updates from the specified SUS/WSUS server rather than from Microsoft's web site.

### Scheduling Antivirus Updates

There are several different methods available to run antivirus updates depending on the antivirus solutions being used. The following are links to white papers for several of the most common solutions. These white papers explain several methods that can be used. Any of these white papers explain concepts that may be used with other solutions not listed here.

Computer Associates eTrust Anti-Virus

[http://www.faronics.com/whitepapers/FAEEnt\\_CAETrust.pdf](http://www.faronics.com/whitepapers/FAEEnt_CAETrust.pdf)

McAfee eTrust Orchestrator

[http://www.faronics.com/whitepapers/FAEEnt\\_McAfeeEPO.pdf](http://www.faronics.com/whitepapers/FAEEnt_McAfeeEPO.pdf)

Sophos Anti-Virus

[http://www.faronics.com/whitepapers/FAEEnt\\_SophosAntivirus.pdf](http://www.faronics.com/whitepapers/FAEEnt_SophosAntivirus.pdf)

Symantec Anti-Virus Corporate Edition

[http://www.faronics.com/whitepapers/FAEEnt\\_SymantecAntivirus.pdf](http://www.faronics.com/whitepapers/FAEEnt_SymantecAntivirus.pdf)

Trend Micro OfficeScan

[http://www.faronics.com/whitepapers/FAEEnt\\_TrendOfficeScan.pdf](http://www.faronics.com/whitepapers/FAEEnt_TrendOfficeScan.pdf)

Panda BusinessSecure Antivirus

[http://www.faronics.com/whitepapers/FAEEnt\\_PandaAntivirus.pdf](http://www.faronics.com/whitepapers/FAEEnt_PandaAntivirus.pdf)

## Scheduling Additional Program Updates

The concepts outlined for the antivirus definition updates can also be applied to updating other applications. However, not all methods described may work with a particular application. Refer to the above antivirus white papers for suggested methods.

## Logon Patch Maintenance

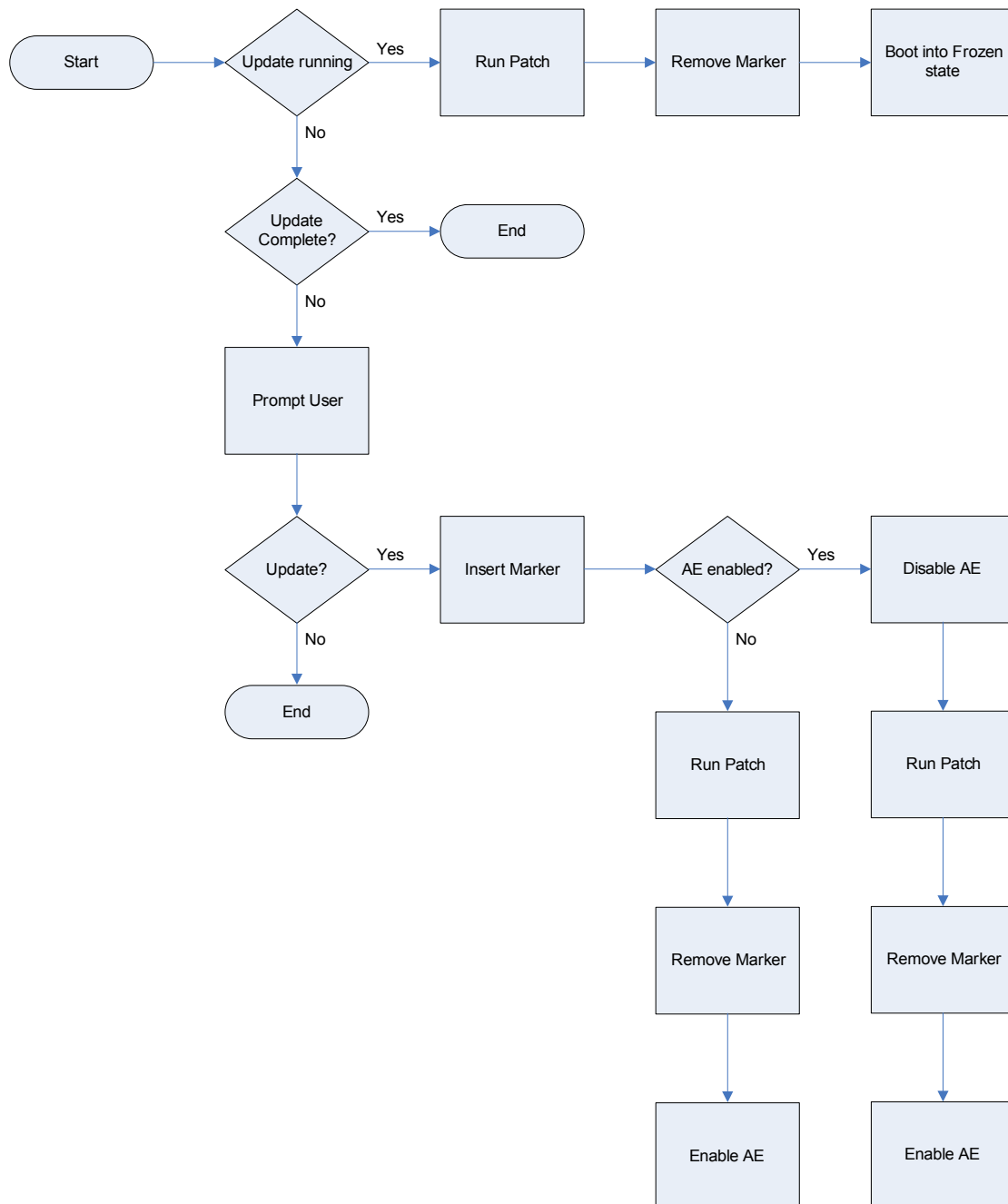
This option allows the administrator to install updates to the client machine when a certain user logs on. In an Active Directory environment, a logon script can be executed to update the client machine. Using the Anti-Executable Command Line Control, Anti-Executable can be disabled before the updates are run and enabled afterwards.

Logon patch maintenance is quite popular in a mobile environment where users are working with laptops and are often on the road. When they get back to the office and login, a script is executed to determine if the users require any updates. If they do, the users are prompted as to whether they would like to run the updates. If they agree, Anti-Executable is disabled, the updates are run, and Anti-Executable is enabled.

The following information is quite complex. Theory will be explained first to provide enough information to get the big picture. Following the theory is an actual example that can be modified and implemented. The example uses a VB script in conjunction with Active Directory Group Policy and text files.

## Logon Patch Maintenance Theory

This concept deals with updating a machine that has Anti-Executable installed when the user logs on. With some slight modifications, the same theory can be applied to an environment where patches are scheduled or performed in real time. The following flowchart outlines the required steps, depending on the state of the update process:



It is important to understand that the above flowchart is a very simple model. In a real-world example, the flowchart would most likely have additional steps to disable the keyboard and mouse and check for the current version of the patch to run. Those steps are beyond the scope of this white paper.

## Logon Patch Maintenance Example

The following example uses an Active Directory environment to call a script file when a user logs on. The following section describes how to create a script based on the earlier flowchart and implement Group Policy to call this script when a user logs on. A full version of the script can be downloaded from the following location:

### Creating the Update Script

This script checks to see if the machine requires updates. If the machine requires an update, it prompts the user. If the user selects *Yes*, Anti-Executable is disabled and the patch is applied. After the patch is applied, Anti-Executable is again enabled.

Use the following steps to create the script file one section at a time:

The script file can be created using many different editors. In this case, Notepad is used.

1. Open Notepad and enter the following text to create the global assemblies:

```
' ***** GLOBAL ASSEMBLIES *****  
Set objNet = CreateObject("WScript.NetWork")
```

This code segment creates an object called *objNet* used throughout the script.

2. Enter the following text to create the global variables:

```
' ***** GLOBAL VARIABLES *****  
strUNCPath = "\\FarDemo.local\NETLOGON\  
strMarkerFile = objNet.ComputerName & ".mar"  
strMarkerCompleteFile = "COMPLETED-" & objNet.ComputerName & ".fin"
```

*strUNCPath* is a variable that maps to a server. Modify the path to match that of the server being used. This is where the marker files are created. The Marker files are used to determine whether the machine requires an update and whether the update is completed.

*strMarkerFile* is a variable holding the name of the marker file used to indicate whether an update is running. Each marker file has the unique name equal to the machine the update is running on.

*strMarkerCompleteFile* is a variable holding the name of the file to indicate if the patch has been run. If this file exists, the update has been run and is not required to run again.

3. Enter the following text to create the main routine:

```

' ***** MAIN *****
' Calls all of the other routines...
If UpdateRunning = True Then
    DisableAE
    RunPatch
    RemoveMarker
    EnableAE
Else
    If UpdateComplete = False Then
        If UserPatchPrompt = True Then
            InsertMarker
            If AEnabled = True Then
                DisableAE
                RunPatch
                RemoveMarker
                EnableAE
            Else
                RunPatch
                RemoveMarker
                EnableAE
            End If
        Else
            ' Exit Script
        End If
    Else
        ' Exit Script
    End If
End If

```

The main routine follows the structure of the flowchart. It calls the other routines as required.

4. Enter the following text to create the *UpdateRunning* function:

```
' ***** UPDATE RUNNING? *****
' Check for marker file. If exists, the update is running. Return True.
Function UpdateRunning
  Set objFS = CreateObject("Scripting.FileSystemObject")
  Set objFolder = objFS.GetFolder(strUNCPath)
  Set objRE = new RegExp
  objRE.Pattern = strMarkerFile
  objRE.IgnoreCase = True

  For Each objFile In objFolder.Files
    If objRE.Test(objFile.Name) Then
      UpdateRunning = True
      Exit Function
    End If
  Next
  UpdateRunning = False
End Function
```

The *UpdateRunning* function checks to see if the marker file exists on the server. If it does, the updates must be running and the function returns the value of *True*.

5. Enter the following text to create the *UpdateComplete* function:

```
' ***** UPDATE COMPLETE? *****
' Checks for completed marker file. If it exists, the update has already
run.
Function UpdateComplete
  Set objFS = CreateObject("Scripting.FileSystemObject")
  Set objFolder = objFS.GetFolder(strUNCPath)
  Set objRE = new RegExp
  objRE.Pattern = strMarkerCompleteFile
  objRE.IgnoreCase = True

  For Each objFile In objFolder.Files
    If objRE.Test(objFile.Name) Then
      UpdateComplete = True
      Exit Function
    End If
  Next
  UpdateComplete = False
End Function
```

The *UpdateComplete* function checks to see if a marker file has been created which signifies the completion of the update. If this file exists, the function returns a value of *True*.

6. Enter the following text to create the *UserPatchPrompt* function:

```
' ***** USER PATCH PROMPT *****  
' Prompt the user whether they would like to run the updates at this time.  
Function UserPatchPrompt  
    intAnswer = MsgBox("An update has been detected. Would you like to run the  
update now?", vbYesNo, "Update Detected")  
    If intAnswer = vbYes Then  
        UserPatchPrompt = True  
        InsertMarker  
    Else  
        UserPatchPrompt = False  
    End If  
End Function
```

The *UserPatchPrompt* function prompts the user with a *Yes/No* dialog. If the user selects *Yes*, the patch runs and the function returns a value of `True`. If the user selects *No*, the function return a value of `False` and the patch will not run.

7. Enter the following text to create the *RunPatch* routine:

```
' ***** RUN PATCH *****  
' The code to run the patches would occur here.  
Sub RunPatch  
    ' Enter code to execute the patch(es)  
    MsgBox "Patch has been applied"  
    InsertCompleteMarker  
End Sub
```

The *RunPatch* routine is used to run the patch. Any code to start a patch can be placed into this routine. After the patch is run, a message is sent to the user indicating the patch has been completed. Another routine, called *InsertCompleteMarker* is run to create a marker file to indicate the patch has been run.

8. Enter the following text to create the *AEEEnabled* function:

```
' ***** ANTI-EXECUTABLE ENABLED? *****  
' Checks to see if Anti-Execututable is on and return True or False.  
Function AEEEnabled  
    Set objShell = CreateObject("Wscript.Shell")  
    intStatus = objShell.Run("AEC password /ISON", 1, True)  
    If intStatus = 0 Then 'AE is disabled  
        AEEEnabled = False  
    Else  
        If intStatus = 1 Then 'AE is enabled  
            AEEEnabled = True  
        Else  
            'A number of other reasons.  
        End If  
    End If  
End Function
```

The *AEEEnabled* function checks to see if Anti-Executable is enabled. If it is enabled, the function returns a value of True. If Anti-Executable is disabled the function returns a value of False.

9. Enter the following text to create the *EnableAE* routine:

```
' ***** ENABLE AE *****  
Sub EnableAE  
    Set objShell = CreateObject("Wscript.Shell")  
    objShell.Run("AEC password /ON")  
End Sub.
```

The *EnableAE* routine is used to enable Anti-Executable on the workstation. The password in the AEC command line must be modified to match the password created for the command line control.

10. Enter the following text to create the *DisableAE* routine:

```
' ***** DISABLE AE *****  
Sub DisableAE  
    Set objShell = CreateObject("Wscript.Shell")  
    objShell.Run("AEC password /OFF")  
End Sub
```

The *DisableAE* routine is used to disable Anti-Executable on the workstation. The password in the AEC command line must be modified to match the password created for the command line control.

11. Enter the following text to create the *InsertMarker* routine:

```
' ***** INSERT MARKER *****
' Insert the marker file to indicate the patch is in progress.
Sub InsertMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    Set objFile = objFSO.CreateTextFile(strUNCPath & strMARKERFILE)
End Sub
```

The *InsertMarker* routine creates a marker file on the server to indicate the patch is currently being run. This marker file remains on the server until it is removed by the *DeleteMarker* routine.

12. Enter the following text to create the *RemoveMarker* routine:

```
' ***** REMOVE MARKER *****
' Remove the marker file to indicate the patch is complete
Sub RemoveMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    objFSO.DeleteFile(strUNCPath & strMarkerFile)
End Sub
```

The *RemoveMarker* routine removes the marker file on the server to indicate the patch is no longer being run.

13. Enter the following text to create the *InsertMarkerComplete* routine:


```
' ***** INSERT UPDATE COMPLETE MARKER *****
' This inserts an update completed file to prevent update looping
Sub InsertCompleteMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    Set objFile = objFSO.CreateTextFile(strUNCPath &
strMarkerCompleteFile)
End Sub
```

The *InsertMarkerComplete* routine creates a file to indicate if the patch has been run on a machine. As long as this file exists on the server, the user is never prompted and the patch is never run.

14. Enter the following text to cleanup the script objects:

```
' ***** CLEANUP *****
Set objNet = Nothing
Set objFile = Nothing
Set objRE = Nothing
Set objFolder = Nothing
Set objTS = Nothing
Set objFS = Nothing
Set objTextFile = Nothing
Set objFSO = Nothing
```

This code cleans up all the objects that have been created throughout the script.

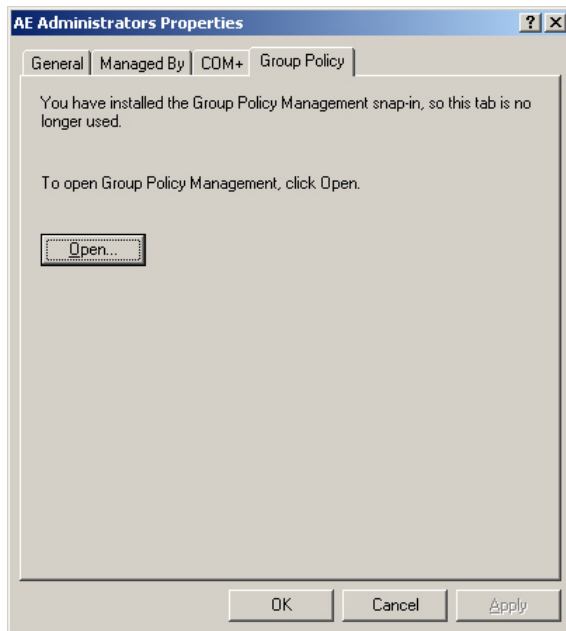
15. Save the file as *AE Update.vbs*. Make sure the file is saved as a *.vbs* and not a *.txt*. The icon should look like the following: 

## Creating the Group Policy

Before the policies are created, ensure the server has been updated to use the Group Policy Management Console. The following documentation assumes this patch has been downloaded and installed on the server. This utility can be found by searching Microsoft's Web site for *Group Policy Management Console*.

It is assumed there is an Organizational Unit (OU) for those users whom will be logging on to the network with a laptop machine requiring updates. Use the following steps to create the group policy:

1. Right-click on the desired User OU and select *Properties*. The properties dialog appears.
2. Select the *Group Policy* tab. If the Group Policy Management console is successfully installed, the following screen appears:



3. Click *Open*.  
The *Group Policy Management* window opens, displaying all the OUs that have been created.
4. Right-click on the desired OU and select *Create and Link GPO Here*. The *New GPO* dialog appears.
5. Type *AeLogonPatchManagement* and click OK.  
A GPO with the name of *AeLogonPatchManagement* appears under the desired OU.

## Modifying the Group Policy

Now that the GPO has been created, it needs to be modified. In this case, the user Logon script is modified using the following steps:

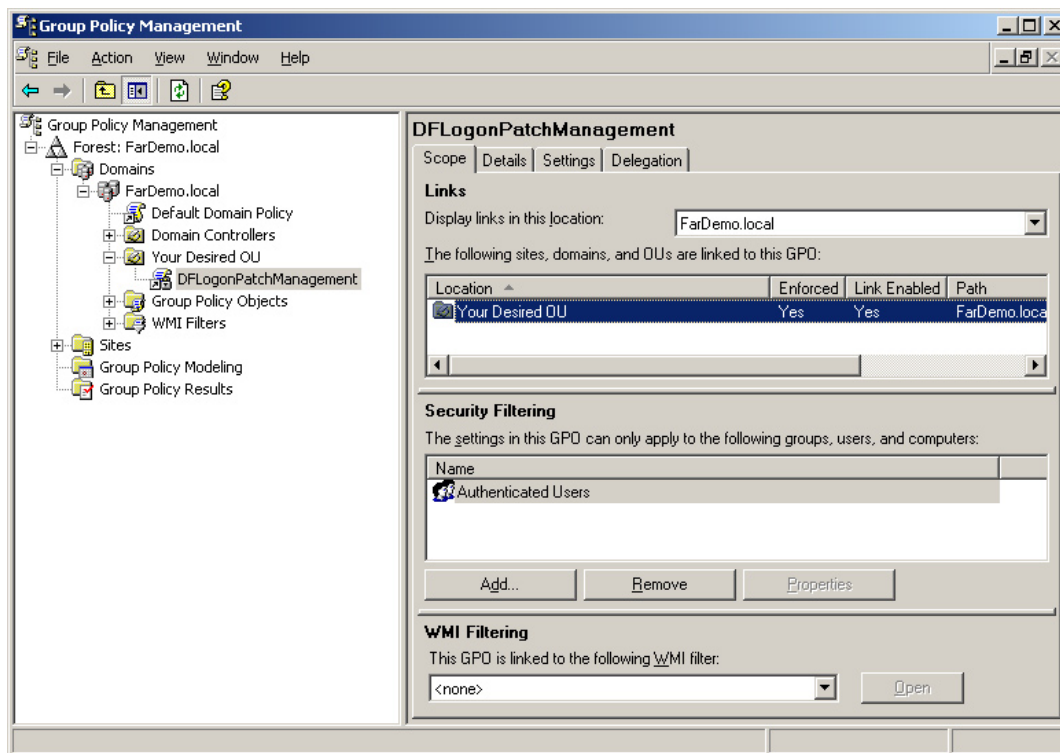
1. Right-click on *DfLogonPatchManagement* and select *Edit*. The *Group Policy Object Editor* opens.
2. Browse to the Logon/Logoff scripts for the user through *User Configuration>Windows Settings>Script (Logon/Logoff)*.
3. Double-click *Logon* to open the *Logon Properties* dialog.
4. Click *Show Files...* to open *Windows Explorer*. Place the script file created earlier in this folder.
5. Close *Windows Explorer*.
6. Click *Add* in the *Logon Properties* dialog. The *Open* dialog should appear and point to the folder where the script was just placed.
7. Select *DF Update.vbs* and click *OK*.
8. Click *OK* on the *Logon Properties* dialog to save the settings.

## Enforcing the Group Policy

The logon script has been configured to execute when the user logs on. However, the GPO is not yet enforced. Enforcing a GPO indicates to the Active Directory server that it needs to run.

To enforce the newly created GPO, right-click on *AeLogonPatchManagement* and select *Enforced* to ensure the logon/logoff scripts are applied to the selected OU.

The policy now indicates that it is enforced. This can be verified by checking to see if the *Enforced* column in the *Group Policy Management* window displays a *Yes*.



## Real Time Patch Maintenance

This method involves patching a machine in real time. This method is best used when the machines are not in use. Sometimes a patch needs to be manually applied to a group of machines. Scheduling the task may not be an option. This method involves disabling Anti-Executable locally at the client through the Enterprise Console or with the Command Line Control. The update can then be applied and Anti-Executable can be re-enabled.

### Disabling Anti-Executable Locally

Use the following steps to disable Anti-Executable from the local machine:

1. To access the Anti-Executable control dialog, use one of the following methods to log on:
  - Press SHIFT and double-click the Anti-Executable icon in the System Tray
  - Use the keyboard shortcut CTRL+SHIFT+ALT+F10
2. The Anti-Executable password dialog appears. Enter your password. This password would have been configured in the Configuration Administrator prior to creating the workstation installation file, or applied through a configuration update.
3. Under the *General* tab, select *Disabled* and click OK. The Anti-Executable icon will change. At this point any program can be executed on this machine.

### Disabling Anti-Executable Through the Enterprise Console

Use the following steps to put a workstation into a Thawed state using the Anti-Executable Enterprise console:

1. Open the Anti-Executable Enterprise console. This program is created under the *Create Programs* tab in the Configuration Administrator.
2. Select the workstations that need to have Anti-Executable disabled.
3. Click the button with the option *Anti-Executable will be switched OFF* in the toolbar. This will disable Anti-Executable on the selected workstations.

### Disabling Anti-Executable Through the Command Line Control

The Anti-Executable Command Line Control can be used to disable Anti-Executable. This control can be used in scripts, batch files, and any other method that allows an *.exe* file to be run with switches. For more information about the different switches offered by the command line control, refer to the following document:

[http://www.faronics.com/whitepapers/FAEEnt\\_RemoteAdministration.pdf](http://www.faronics.com/whitepapers/FAEEnt_RemoteAdministration.pdf)

## Appendix A - Faronics Anti-Executable and SUS/WSUS FAQ

Which Windows operating systems are able to use the Run Windows Updates feature in Faronics Anti-Executable Enterprise?

The *Run Windows Updates* feature only supports the following versions of Windows:

- Windows 2000 Service Pack 2 or 3 with SUS Client
- Windows 2000 Service Pack 4 which includes the SUS Client
- Windows XP with SUS Client
- Windows XP Service Pack 1 which includes the SUS Client

Does the *Run Windows Updates* feature require an administrator to be logged onto the workstation?

The feature will work while logged on as any type of user, or even if the workstation isn't logged on at all. This feature uses the Windows Update service running under the local system account.

How can I be sure that the updates have been installed correctly?

Faronics Anti-Executable Enterprise does not actually perform the updates or track which updates have been installed. The normal Microsoft method of installing updates is used instead. To check if the update took place, consult the workstation's update log at `C:\WINDOWS\Windows Update log`.

What happens if an update is interrupted during download or installation because the Maintenance period ended or the workstation was restarted or powered off?

A minimum Maintenance period of five hours is required to use the *Run Windows Updates* feature in order to minimize the likelihood of this occurring. If an update is incomplete for any reason, the mechanism that Microsoft uses will correct and reinstall the update the next time the service is called.

Will the workstation restart during the update process if the update being installed requires it to?

Yes, the workstation will restart as many times as are required, until the updates are completed.

I would like my Windows 95/98/Me workstations to have a Maintenance period of less than five hours, but I need to assign at least five hours to my Windows 2000/XP workstations. How can I do this?

You will need to create a second installation program for the Windows 95/98/Me workstations and uncheck *Run Windows Updates* in that configuration. Update only the Windows 95/98/Me workstations with this second configuration.

Why does Faronics Anti-Executable require a Maintenance period of at least five hours in order to run Windows updates?

The five hour period is enforced to prevent problems with workstations being rebooted in the middle of installing a patch, due to a slow Internet connection, a very large critical update, or overwhelmed servers at Microsoft due to a spike in demand. Especially due to the availability of Windows XP Service Pack 2, we needed to make sure that adequate time was forced to allow this patch to install correctly.

What do I have to configure on each workstation to ensure that the updates are downloaded during the Maintenance period?

Faronics Anti-Executable Enterprise will automatically coordinate the update. Faronics Anti-Executable Enterprise will not actually perform the update but will call the Microsoft update service during the Maintenance period. The Microsoft update service will then perform the

update either via the Internet or a designated SUS server.

Can the IP address of the SUS server be updated with a configuration update?

Yes, all of the Maintenance options can be changed with a configuration update.

Where can I download the files that I need to get this feature working?

Microsoft SUS client can be downloaded at:

<http://www.microsoft.com/windows2000/downloads/recommended/susclient/default.asp>

Microsoft SUS server can be downloaded by searching for SUS server at:

<http://www.microsoft.com>

Where can I find more information about deploying SUS?

Microsoft has a comprehensive Software Update Services Deployment White Paper available at: <http://www.microsoft.com/windowsserversystem/sus/susdeployment.mspx>

I already have a GPO (Group Policy Objects) set that forces down updates. Will this override the Run Windows Updates settings in Faronics Anti-Executable or cause any problems?

Faronics Anti-Executable will fully coordinate the Windows Update using the Microsoft mechanisms. If any third-party patch management products are deployed, then using Faronics Anti-Executable's *Run Windows Updates* feature is redundant and not recommended.

I want to use the Run Windows Updates feature, but the *Automatic Updates* tab of my System Properties Control Panel does not allow me to enable Windows updates. How can I change this setting?

Faronics Anti-Executable Enterprise disables the Automatic Updates settings in the System Properties Control Panel so they won't interfere with the normal operation of Anti-Executable. No settings need to be configured if you have checked *Run Windows Updates* in the Faronics Anti-Executable configuration that is installed. Faronics Anti-Executable will automatically make the call to update the software independent of the settings in this panel.

Why can't I select *Run Windows Updates* and *Run Batch File* on the same day? I am unable to check both boxes on the Maintenance tab of the Configuration Administrator.

Faronics Anti-Executable does not allow *Run Windows Updates* and *Run Batch File* to both be checked for the same day of the week. This is done to eliminate the possibility of a conflict occurring between the two options. It is possible to use a batch file to start Windows Updates and other updates that are required.

## Appendix B - Anti-Executable Update Script

The entire script explained in the Logon Patch Maintenance section has been included here. This makes it easy to see the entire script rather than one segment at a time. The script can be downloaded from the following address: [http://www.faronics.com/exe/AEEnt\\_ADUpdateScript.zip](http://www.faronics.com/exe/AEEnt_ADUpdateScript.zip)

```
' *****
' ***          AE SIMPLE UPDATE SCRIPT SAMPLE          ***
' ***                                                              ***
' *** Author:   Faronics Coproration                    ***
' *** Date:     January 2006                            ***
' ***                                                              ***
' *** Associated Files:                                  ***
' *** <ComputerName>.mar - Used to indicate patch is running ***
' *** COMPLETE-<ComputerName>.fin - Indicates patch complete ***
' *** AEC.exe - Anti-Executable Command Line Control      ***
' *****

' NOTES:
' The following script will turn off Anti-Execuatble, run updates and turn on Anti-
' Executable.

' ***** GLOBAL ASSEMBLIES *****
Set objNet = CreateObject("WScript.NetWork")

' ***** GLOBAL VARIABLES *****
strUNCPath = "\\FarDemo.local\NETLOGON\"
strMarkerFile = objNet.ComputerName & ".mar"
strMarkerCompleteFile = "COMPLETED-" & objNet.ComputerName & ".fin"

' ***** MAIN *****
' Calls all of the other routines...
If UpdateRunning = True Then
    DisableAE
    RunPatch
    RemoveMarker
    EnableAE
Else
    If UpdateComplete = False Then
        If UserPatchPrompt = True Then
            InsertMarker
            If AEEEnabled = True Then
                DisableAE
                RunPatch
                RemoveMarker
                EnableAE
            Else
                RunPatch
                RemoveMarker
                EnableAE
            End If
        End If
    Else

```

```
        ' Exit Script
    End If
Else
    ' Exit Script
    End If
End If

' ***** UPDATE RUNNING? *****
' Check for marker file. If exists, the update is running. Return True.
Function UpdateRunning
    Set objFS = CreateObject("Scripting.FileSystemObject")
    Set objFolder = objFS.GetFolder(strUNCPath)
    Set objRE = new RegExp
    objRE.Pattern = strMarkerFile
    objRE.IgnoreCase = True

    For Each objFile In objFolder.Files
        If objRE.Test(objFile.Name) Then
            UpdateRunning = True
            Exit Function
        End If
    Next
    UpdateRunning = False
End Function

' ***** UPDATE COMPLETE? *****
' Checks for completed marker file. If it exists, the update has already run.
Function UpdateComplete
    Set objFS = CreateObject("Scripting.FileSystemObject")
    Set objFolder = objFS.GetFolder(strUNCPath)
    Set objRE = new RegExp
    objRE.Pattern = strMarkerCompleteFile
    objRE.IgnoreCase = True

    For Each objFile In objFolder.Files
        If objRE.Test(objFile.Name) Then
            UpdateComplete = True
            Exit Function
        End If
    Next
    UpdateComplete = False
End Function

' ***** USER PATCH PROMPT *****
' Prompt the user whether they would like to run the updates at this time.
Function UserPatchPrompt
    intAnswer = MsgBox("An update has been detected. Would you like to run the update now?", vbYesNo, "Update Detected")
    If intAnswer = vbYes Then
        UserPatchPrompt = True
        InsertMarker
    End If
End Function
```

```

Else
    UserPatchPrompt = False
End If
End Function

' ***** RUN PATCH *****
' The code to run the patches would occur here.
Sub RunPatch
    ' Enter code to execute the patch(es)
    MsgBox "Patch has been applied"
    InsertCompleteMarker
End Sub

' ***** ANTI-EXECUTABLE ENABLED? *****
' Checks to see if Anti-Executable is on and return True or False.
Function AEEEnabled
    Set objShell = CreateObject("Wscript.Shell")
    intStatus = objShell.Run("AEC password /ISON", 1, True)
    If intStatus = 0 Then 'AE is disabled
        AEEEnabled = False
    Else
        If intStatus = 1 Then 'AE is enabled
            AEEEnabled = True
        Else
            'A number of other reasons.
        End If
    End If
End Function

' ***** INSERT MARKER *****
' Insert the marker file to indicate the patch is in progress.
Sub InsertMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    Set objFile = objFSO.CreateTextFile(strUNCPath & strMARKERFILE)
End Sub

' ***** REMOVE MARKER *****
' Remove the marker file to indicate the patch is complete
Sub RemoveMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    objFSO.DeleteFile(strUNCPath & strMarkerFile)
End Sub

' ***** INSERT UPDATE COMPLETE MARKER *****
' This inserts an update completed file to prevent update looping
Sub InsertCompleteMarker
    Set objFSO = CreateObject("Scripting.FileSystemObject")
    Set objFile = objFSO.CreateTextFile(strUNCPath & strMarkerCompleteFile)
End Sub

```

```
' ***** DISABLE AE *****  
Sub DisableAE  
  Set objShell = CreateObject("Wscript.Shell")  
  objShell.Run("AEC password /OFF")  
End Sub
```

```
' ***** ENABLE AE *****  
Sub EnableAE  
  Set objShell = CreateObject("Wscript.Shell")  
  objShell.Run("AEC password /ON")  
End Sub
```

```
' ***** CLEANUP *****  
Set objNet = Nothing  
Set objFile = Nothing  
Set objRE = Nothing  
Set objFolder = Nothing  
Set objTS = Nothing  
Set objFS = Nothing  
Set objTextFile = Nothing  
Set objFSO = Nothing
```