

FARONICS

DEEPFREEZE™

ABSOLUTE Workstation Integrity

Deep Freeze Enterprise - Patch Management

TECHNICAL WHITEPAPER

Last modified: June 26, 2009

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

© 1999 - 2009 Faronics Corporation. All rights reserved. Faronics, Deep Freeze, Faronics Core Console, Faronics Anti-Executable, Faronics Device Filter, Faronics Power Save, Faronics Insight, Faronics System Profiler, and WINSelect are trademarks and/or registered trademarks of Faronics Corporation. All other company and product names are trademarks of their respective owners.

Contents

Introduction	3
Scheduled Patch Maintenance	3
Scheduling Windows Updates.....	3
Scheduling Windows Updates in a Deep Freeze Maintenance Period	4
Scheduling Windows Updates through Group Policy	5
Scheduling Antivirus Updates.....	7
Scheduling Additional Program Updates	7
Logon Patch Maintenance	8
Logon Patch Maintenance Theory	8
Logon Patch Maintenance Example	9
Creating the Update Script.....	9
Creating the Group Policy	15
Modifying the Group Policy	16
Enforcing the Group Policy	16
Real Time Patch Maintenance	17
Disabling Deep Freeze Locally	17
Disabling Deep Freeze Through the Enterprise Console	17
Disabling Deep Freeze Through the Command Line Control.....	17
Configuring Software to Update in a Thawed Location	17
Appendix A: Deep Freeze and SUS/WSUS FAQ	18
Appendix B: Deep Freeze Update Script	19
Appendix C: Common Update Scenarios	23
Scenario 1: Updating Clients in a Dynamic Update Environment	23
Scenario 2: Updating in a 24-Hour Lab Environment.....	23
Scenario 3: Updating in a Mobile Environment	23

Introduction

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

Deep Freeze allows systems administrators to ensure the integrity of their computers against exploits — even ones that have yet to be discovered. However, it introduces some challenges within the process of applying patches because Deep Freeze does not discriminate — it removes both the good and the bad changes and returns the computer to its original, pristine state on every restart.

There are several methods for integrating Deep Freeze with patch management. When properly done, users can enjoy the bulletproof reliability of a Deep Freeze 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 a Deep Freeze environment.

Scheduled Patch Maintenance

Scheduled patch maintenance allows the administrator to specify a period of time when the client computers restart with Deep Freeze in a Thawed state. During this Maintenance Period, software updates, Windows Updates, and antivirus definition updates can be scheduled. Scripts can be run and batch files can be executed.

Scheduled patch maintenance is an appropriate strategy for computer labs. 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.

Maintenance Periods are configured using the Deep Freeze Configuration Administrator. The Configuration Administrator is used to configure workstation installation files as well as configuration files. Configuration files can be used to apply the changes to deployed computers through the Deep Freeze 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

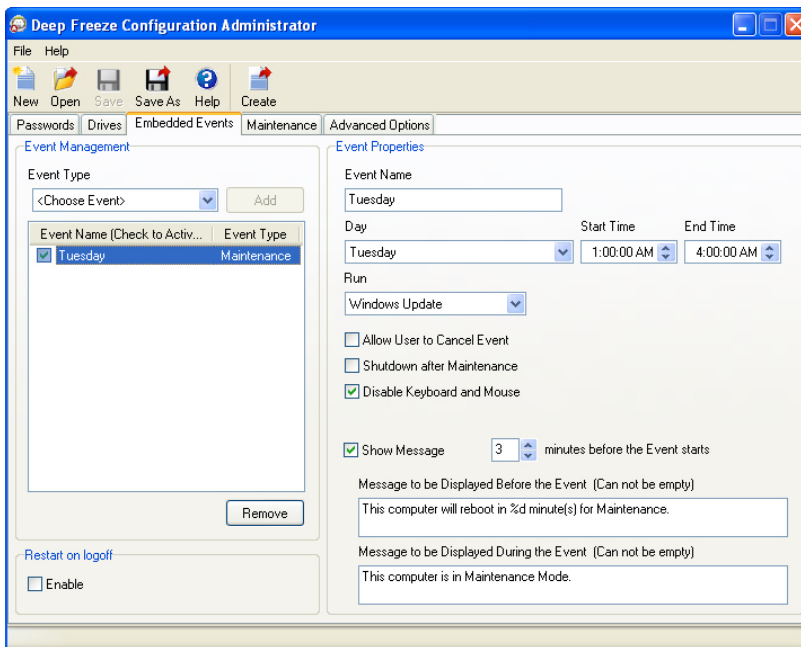
There are several different methods available to run a Windows update in a Frozen environment. Deep Freeze can be set up to start a Windows update during a Maintenance Period. Deep Freeze can also be set up to execute a batch file during the Maintenance Period. A batch file could be used to start the Windows update process. Finally, another program could be used to start Windows updates during the Deep Freeze Maintenance Period.

Scheduling Windows Updates in a Deep Freeze Maintenance Period

The first method involves setting up a Maintenance Period using the Deep Freeze Configuration Administrator. An option is selected so Deep Freeze will run Windows updates after the computer goes into Maintenance Mode.

Complete the following steps to configure a Maintenance Period:

1. In the Deep Freeze Configuration Administrator, click the *Embedded Events* tab.
2. Specify day and time the Maintenance Period will occur. The window should look similar to the following:

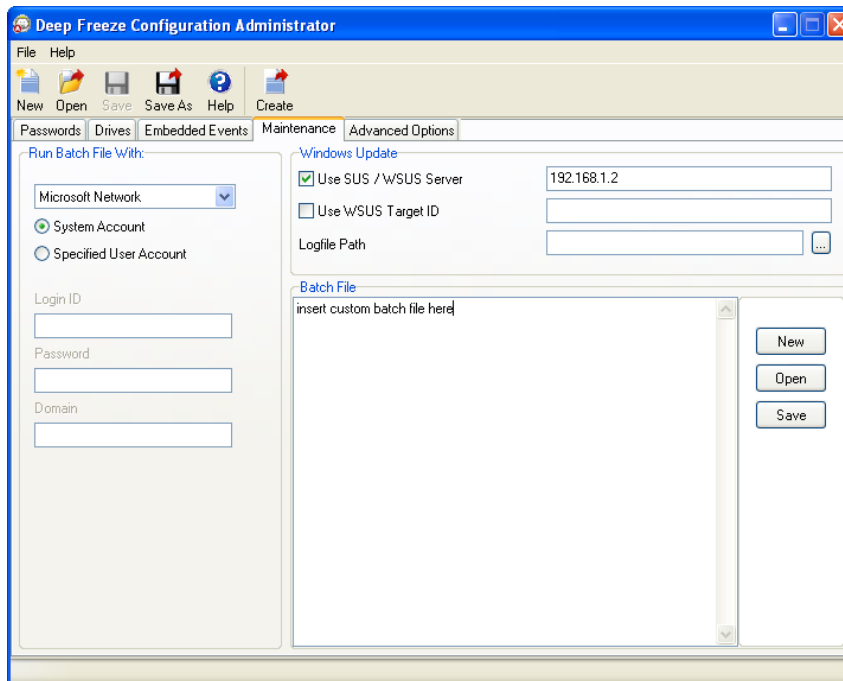


In the above screen, Tuesday has been selected for Maintenance with the *Run: Windows Update* option. At 1:00 AM, the computer restarts in a Thawed state. At 4:00 AM, the computer restarts in a Frozen state. The *Disable Keyboard and Mouse* checkbox has been checked; this means that the keyboard and mouse are locked while the computer is in Maintenance Mode.

If a client computer is configured with this option, it would attempt to download and run updates from Microsoft's web site during the Maintenance Period.

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

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



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

Scheduling Windows Updates through Group Policy

Although settings for Windows update can be administered through Deep Freeze, many administrators prefer to control settings for automatic updates through the use of Group Policy in a domain environment.

It is recommended that the following configuration be used to configure the Automatic Update client on a computer running Deep Freeze:

Computer Configuration → *Administrative Templates* → *Windows Update*:

- *Configure Automatic Updates*: Enabled.
- *Configure Automatic Updating*: Enabled.
- *Scheduled install day*: 0 if Maintenance will happen every day; the specific day if Maintenance will only occur once per week.
- *Scheduled install time*: Set to 30 minutes after the start of the Deep Freeze Maintenance Period.
- *Do not display Install Updates and Shut Down option in Shutdown Windows dialog box*: Enabled
- *Do not adjust default option to Install Updates and Shut Down in Shut Down Windows dialog box*: Enabled

- *Reschedule Automatic Updates scheduled installations*: Disabled
- *No auto-restart for scheduled Automatic Updates installations*: Disabled

This policy will ensure that Windows updates are installed during the Maintenance Period and that any updates that are downloaded (but not installed) will not attempt to reinstall on the client computer while Frozen.

Administrators need to ensure that the Maintenance Period configured in Deep Freeze is long enough to complete the download and installation of the updates from whatever source is configured and that computers are either left on to enter the Maintenance Period or are woken up prior to the start of the Maintenance Period.

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/DFEnt_CAETrust.pdf

Kaspersky Antivirus

http://www.faronics.com/doc/wp/DFEnt_Kaspersky.pdf

McAfee eTrust Orchestrator

http://www.faronics.com/whitepapers/DFEnt_McAfeeEPO.pdf

Panda BusinessSecure Antivirus

http://www.faronics.com/whitepapers/DFEnt_PandaAntivirus.pdf

Sophos Anti-Virus

http://www.faronics.com/whitepapers/DFEnt_SophosAntivirus.pdf

Symantec Anti-Virus Corporate Edition

http://www.faronics.com/whitepapers/DFEnt_SymantecAntivirus.pdf

Trend Micro OfficeScan

http://www.faronics.com/whitepapers/DFEnt_TrendOfficeScan.pdf

For additional white papers describing antivirus products that may have been added to the Faronics Content Library after publication of this white paper, refer to:

<http://www.faronics.com/library>

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, or the white paper entitled *Retaining User Data* at the following location: http://www.faronics.com/whitepapers/DF_RetainUserData.pdf

Logon Patch Maintenance

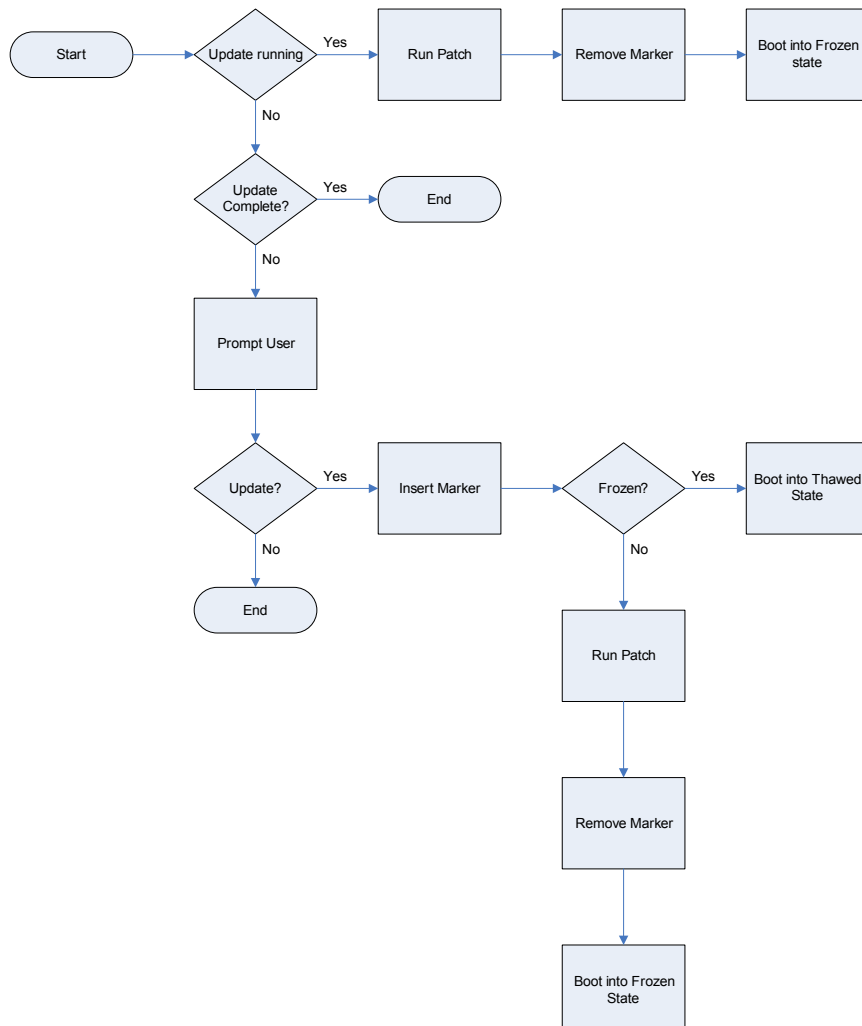
This option allows the administrator to install updates to the client computer when a certain user logs on. In an Active Directory environment, a logon script can be executed to update the client computer. Using Deep Freeze command line control (DFC), Deep Freeze can be disabled before the updates are run and re-enabled afterwards.

Logon patch maintenance is appropriate in mobile environments where users are working with laptops and are often off the local network. When they get back to the office or school 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, Deep Freeze is disabled, the updates are run, and Deep Freeze is re-enabled.

The example below assumes that the person implementing the script is familiar with Group Policy, Active Directory, and Visual Basic Scripting.

Logon Patch Maintenance Theory

This concept deals with updating a Frozen machine 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:



Because the computer boots several times, the script needs to check a value to see what phase of the script is currently running. Because the computer will be Frozen at times, a value cannot be stored in the Frozen partition. This means the value must be stored either on the network or in a Thawed partition on the computer.

It is also 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:

http://www.faronics.com/exe/DFEnt_ADUpdateScript.zip

Creating the Update Script

This script checks to see if the computer requires updates. If the computer requires an update, it prompts the user. If the user selects *Yes*, the computer is put into a Thawed state. At this point, the patch is applied and the computer is returned to a Frozen state.

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
    RunPatch
    RemoveMarker
    BootFrozen
Else
    If UpdateComplete = False Then
        If UserPatchPrompt = True Then
            InsertMarker
            If Frozen = True Then
                BootThawed
            Else
                RunPatch
                RemoveMarker
                BootFrozen
            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("Anupdatehasbeendetected.Wouldyouliketoruntheupdatenow?"&vbLF&_
"The update process will require several reboots!", 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 *Frozen* function:

```
' ***** DEEP FREEZE FROZEN? *****
' Checks to see if Deep Freeze is Frozen and returns True or False.
Function Frozen
  Set objShell = CreateObject("Wscript.Shell")
  intStatus = objShell.Run("DFC password /ISFROZEN", 1, True)
  If intStatus = 0 Then 'DF is Thawed
    Frozen = False
  Else
    If intStatus = 1 Then 'DF is Frozen
      Frozen = True
    Else
      'A number of other reasons.
    End If
  End If
End Function
```

The *Frozen* function checks to see if Deep Freeze is Frozen. If it is Frozen, the function returns a value of `True`. If Deep Freeze is Thawed, the function returns a value of `False`.

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

```
' ***** BOOT FROZEN *****
Sub BootFrozen
  Set objShell = CreateObject("Wscript.Shell")
  objShell.Run("DFC password /BOOTFROZEN")
End Sub
```

The *BootFrozen* routine is used to put computers into a Frozen State. The password in the DFC command line must be modified to match password created for the command line control.

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

```
' ***** BOOT THAWED *****  
Sub BootThawed  
    Set objShell = CreateObject("Wscript.Shell")  
    objShell.Run("DFC password /BOOTTHAWED")  
End Sub
```

The *BootThawed* routine is used to set computers in a Thawed state. The password in the DFC 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 computer. 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 *DF Update.vbs*. Make sure the file is saved as a *.vbs* and not a *.txt*. The icon should look like the following: 

The script is now ready to be implemented through a logon script in Group Policy.

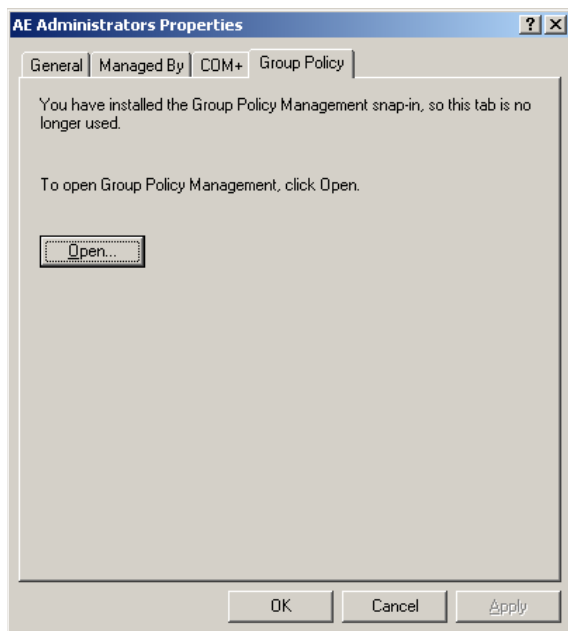
NOTE: The script does not contain any error handling.

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. The 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 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 *DfLogonPatchManagement* and click OK.
A GPO with the name of *DfLogonPatchManagement* 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:

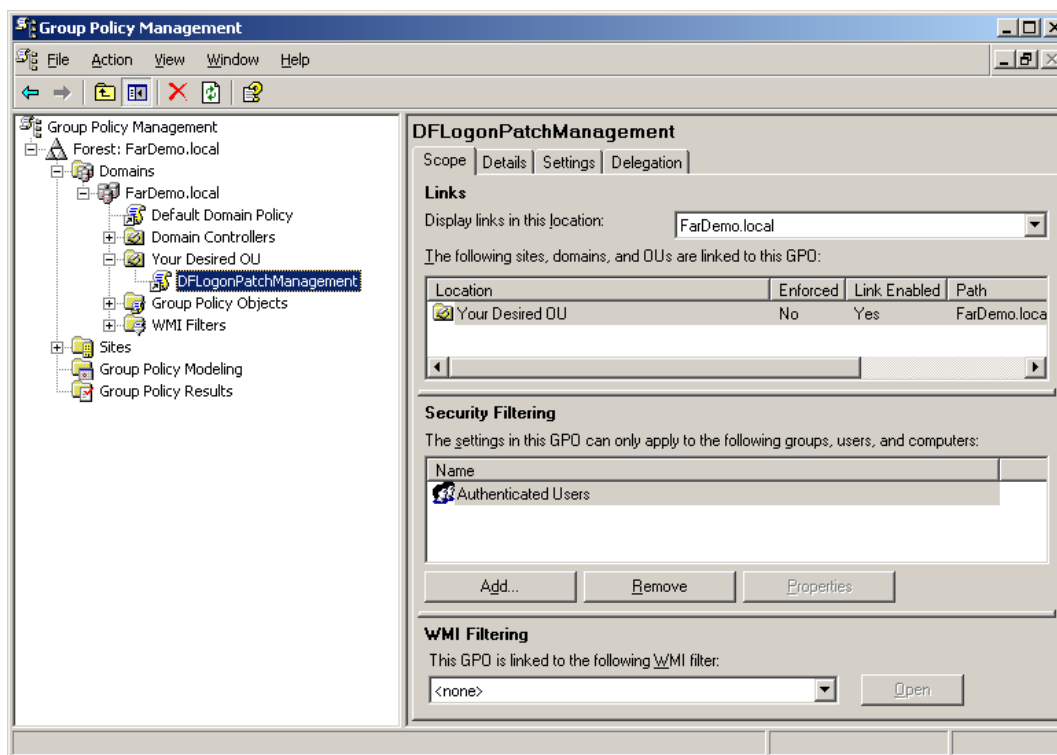
1. Right-click on *DfLogonPatchManagement* and select *Edit*. The *Group Policy Object Editor* opens.
2. Browse to the Logon/Logoff scripts for the user through *User Configuration>Windows Settings>Script (Logon/Logoff)*.
3. Double-click *Logon* to open the *Logon Properties* dialog.
4. Click *Show Files...* 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 *DfLogonPatchManagement* 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 computer in real time. It is best used when the computers are not in use. Sometimes a patch needs to be manually applied to a group of computers and scheduling the task may not be an option. This method involves disabling Deep Freeze locally at the client through the Enterprise Console or with the command line control (DFC). The update can then be applied and Deep Freeze can be re-enabled.

Disabling Deep Freeze Locally

Use the following steps to put Deep Freeze into a Thawed state from the local computer:

1. To access the Deep Freeze login dialog, use one of the following methods to log on:
 - Press **SHIFT** and double-click the Deep Freeze icon in the System Tray
 - Use the keyboard shortcut **CTRL+SHIFT+ALT+F6**
2. The Deep Freeze login dialog appears. Enter your Deep Freeze 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 *Boot Control* tab, select *Boot Thawed* and click *OK*. When the computer restarts, it is in a Thawed state. At this point any changes made to the computer are permanent.

Disabling Deep Freeze Through the Enterprise Console

Use the following steps to put a computer into a Thawed state using the Deep Freeze Enterprise Console:

1. Launch the Deep Freeze Enterprise Console.
2. Select the computers that need to be put into a Thawed state.
3. Click the *Reboot Thawed* icon in the toolbar or right-click and select *Reboot Thawed* from the menu. The selected computers will reboot in the Thawed state.

Disabling Deep Freeze Through the Command Line Control

The Deep Freeze command line control (DFC) can be used to disable Deep Freeze. It can be used in scripts, batch files, and in conjunction with any third party management utility capable of pushing scripts to systems. For more information about the different switches offered by the command line control, refer to the following document:

http://www.faronics.com/whitepapers/DF_RemoteAdministration.pdf

Configuring Software to Update in a Thawed Location

It is possible to update software that resides in a Thawed location. In these cases, the software would have to exist entirely on the Thawed partition. Remember the following rules when configuring software to run from a Thawed location:

1. If updates have to make changes to the registry, Deep Freeze needs to be in a Thawed state. The reason for this is that the registry is stored on the Frozen location.
2. Many programs store data to the user folders. The user folders can be redirected to a Thawed location. However, if the user folders are not redirected, ensure that the updates are not making changes to settings stored there.

Appendix A: Deep Freeze and SUS/WSUS FAQ

Does the Run Windows Updates feature require an administrator to be logged into the computer?

The feature works while any type of user is logged in, or if the computer isn't logged in at all. It uses the Windows Update service running under the local system account.

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

If you are using Deep Freeze to install the updates, you can view the status of the updates in the *DFWuLogfile.log* in the *Faronics* folder. The location and name of this file can be changed on the *Maintenance* tab of the Deep Freeze Configuration Administrator.

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

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 computer restart during the update process if the update being installed requires it to do so?

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

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

Deep Freeze automatically coordinates the update. Deep Freeze does not actually perform the update but calls the Microsoft update service during the Maintenance Period. The Microsoft update service then performs the update either via the Internet or a designated SUS/WSUS server.

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

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

Appendix B: Deep Freeze Update Script

The entire script explained in the Logon Patch Maintenance section has been included here. It can be downloaded from the following address: http://www.faronics.com/exe/DFEnt_ADUpdateScript.zip

```
' *****
' ***          DF SIMPLE UPDATE SCRIPT SAMPLE          ***
' ***
' *** Author:   Faronics Corporation                    ***
' *** Date:    12/29/2005                              ***
' ***
' *** Associated Files:                                ***
' *** <ComputerName>.mar - Used to indicate patch is running ***
' *** COMPLETE-<ComputerName>.fin - Indicates patch complete ***
' *** DFC.exe - Deep Freeze Command Line Control      ***
' *****

' NOTES:
' The following script will turn off Deep Freeze, run updates and turn on Deep Freeze.

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

' ***** GLOBAL VARIABLES *****
' Modify the UNC path to match that of your server environment.
strUNCPath = "\\FarDemo.local\NETLOGON\"
strMarkerFile = objNet.ComputerName & ".mar"
strMarkerCompleteFile = "COMPLETED-" & objNet.ComputerName & ".fin"

' ***** MAIN *****
' Calls all of the other routines...
If UpdateRunning = True Then
    RunPatch
    RemoveMarker
    BootFrozen
Else
    If UpdateComplete = False Then
        If UserPatchPrompt = True Then
            InsertMarker
            If Frozen = True Then
                BootThawed
            Else
                RunPatch
                RemoveMarker
                BootFrozen
            End If
        Else
            ' Exit Script
        End If
    Else
        ' Exit Script
    End If
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("Anupdatehasbeendetected.Wouldyouliketoruntheupdatenow?"&vbLF&_
        "The update process will require several reboots!", vbYesNo, "Update Detected")
    If intAnswer = vbYes Then
        UserPatchPrompt = True
        InsertMarker
    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)
  ' The next two lines would run a program by the name of update.exe
  ' Set objShell = CreateObject("Wscript.Shell")
  ' objShell.Run("update.exe")
  MsgBox "Patch has been applied"
  InsertCompleteMarker
End Sub

' ***** DEEP FREEZE FROZEN? *****
' Checks to see if Deep Freeze is Frozen and returns True or False.
Function Frozen
  Set objShell = CreateObject("Wscript.Shell")
  intStatus = objShell.Run("DFC password /ISFROZEN", 1, True)
  If intStatus = 0 Then 'DF is Thawed
    Frozen = False
  Else
    If intStatus = 1 Then 'DF is Frozen
      Frozen = True
    Else
      'A number of other reasons.
    End If
  End If
End Function

' ***** BOOT FROZEN *****
Sub BootFrozen
  Set objShell = CreateObject("Wscript.Shell")
  objShell.Run("DFC password /BOOTFROZEN")
End Sub

' ***** BOOT THAWED *****
Sub BootThawed
  Set objShell = CreateObject("Wscript.Shell")
  objShell.Run("DFC password /BOOTTHAWED")
End Sub

' ***** 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

' ***** 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
```

Appendix C: Common Update Scenarios

The following section presents some update scenarios and possible solutions to these scenarios.

Scenario 1: Updating Clients in a Dynamic Update Environment

Requirement:

The policy in the organization is to update the computers as soon as possible with the latest critical updates and antivirus definitions. Using management software, the updates are delivered to the client computers as soon as the updates are available. Because the computers have Deep Freeze installed, these updates are removed with a restart. How can patches be deployed in this type of environment?

Solution:

A scheduled Maintenance Period can be used to permanently update the client machines with the newest updates. The Maintenance Period could be scheduled to occur as often as desired. Any changes made during the Maintenance Period are permanent. Any new updates made to the computer while it is Frozen are only present until the machine is restarted. This has the same exact effect of a machine that is not Frozen with all the benefits of a Frozen machine. For more information, refer to the section entitled [Scheduled Patch Maintenance](#).

Scenario 2: Updating in a 24-Hour Lab Environment

Requirement:

Some cases exist where computers are in use for 24 hours. In these environments, it can be difficult to take computers offline to apply changes. Most patches do not require a restart. In order to disable Deep Freeze, a restart is required. How can patches be deployed in this type of environment?

Solution:

In these types of environments, the computers should be kept in a consistent state. Deep Freeze ensures these computers are reliable and available at all times. The solution for patching is based on a rotation. As one computer is rotated in, another is taken offline to be updated.

An alternate solution is to have the whole lab taken offline to perform a real-time update. For more information, refer to the section entitled [Real-Time Patch Maintenance](#).

Scenario 3: Updating in a Mobile Environment

Requirement:

All users work with portable laptops. These laptops are almost never connected to the network. When they do connect to the network, updates are run. It is not possible to configure a Maintenance Period. What methods are available here?

Solution:

This environment will most likely use a logon script when the laptop is attached to the network. This script initializes the update procedure. Deep Freeze can easily be disabled at this time using its command line control (DFC). This control can be called within the script. For more information, refer to the section entitled [Logon Patch Maintenance](#).