Chapter No.3
"Solving Least Privilege Problems with the Application Compatibility Toolkit"
In this package, you will find:

A Biography of the author of the book

A preview chapter from the book, Chapter NO.3 "Solving Least Privilege Problems with the Application Compatibility Toolkit"

A synopsis of the book’s content

Information on where to buy this book

About the Author

Russell Smith specializes in management and security of Microsoft-based IT systems and is a Contributing Editor for CDW’s Biztech magazine and writes regularly for industry journal Windows IT Professional. Russell is also contributing author to Supporting and Troubleshooting Applications on a Microsoft Windows Vista Client for Enterprise Support Technicians from Microsoft’s Official Academic Course (MOAC) series of books published by Wiley and Sons.

An independent IT consultant and MCSE with more than ten years of experience, Russell's recent projects include Active Directory Security Consultant for the UK Health Service National Programme for Information Technology (NPfIT) and Exchange Architect for Wipro Technologies. Russell also has extensive experience as an IT trainer.

For More Information:
Least Privilege Security for Windows 7, Vista and XP

In this, the first book to be entirely dedicated to the subject of running Least Privilege Security (or standard user accounts) on Windows operating systems in the enterprise, you will learn about the benefits Least Privilege brings organizations in terms of not only security, but regulatory compliance, improved manageability, and operational simplicity. The book provides a complete guide to implementing Least Privilege Security on the desktop, with step-by-step instructions and advice about how to overcome the most common technical and political challenges.

What This Book Covers

Chapter 1, An Overview of Least Privilege Security in Microsoft Windows, explores the principle of Least Privilege Security and shows how to implement it in different versions of Microsoft Windows. It also explains how to control and change system privileges, benefit from implementing Least Privilege Security on the desktop, and overcome the most common technical and political problems and challenges when implementing Least Privilege Security.

Chapter 2, Political and Cultural Challenges for Least Privilege Security, covers the reasons why users may not accept Least Privilege Security on the desktop. It also clearly explains and justifies the benefits of Least Privilege Security for your organization. The chapter also covers how to apply Least Privilege Security to different categories of users and get buy-in from management.

Chapter 3, Solving Least Privilege Problems with the Application Compatibility Toolkit, covers how to modify incompatible applications on the fly and achieve the best balance between compatibility and security by using Application Compatibility shims. It explains how to create shims using the Application Compatibility Toolkit 5.5 and distribute compatibility databases to devices across the enterprise.

Chapter 4, User Account Control, covers how to achieve a seamless user experience by using the different components and compatibility features of User Account Control. It also explains how to configure User Account Control on multiple computers using Group Policy and the inner workings of User Account Control's core components.

Chapter 5, Tools and Techniques for Solving Least Privilege Security Problems, covers how to set up a system for temporarily granting administrative privileges to standard users for support purposes. It also covers how to use Task Scheduler to run common processes without the need to elevate privileges and how to install third-party solutions to configure administrative privileges for applications and Windows processes on-the-fly.

For More Information:
Chapter 6, *Software Distribution using Group Policy*, explains how to prepare applications for Group Policy Software Installation (GPSI) and Windows Installer deployment. It also explains how to repackage legacy setup programs in Windows Installer .msi format and how to make GPSI more scalable and flexible using the Distributed File System (DFS). It covers how to target client computers using Windows Management Instrumentation (WMI) filters and Group Policy Scope of Management.

Chapter 7, *Managing Internet Explorer Add-ons*, covers how to support per-user and per-machine ActiveX controls and manage Internet Explorer add-ons via Group Policy. It also explains how to install per-machine ActiveX controls using the ActiveX Installer Service (AxIS) and how to implement best practices for working with ActiveX controls in a managed environment.

Chapter 8, *Supporting Users Running with Least-Privilege*, explains how to support Least-Privilege user accounts using reliable remote access solutions, how to connect to remote systems with administrative privileges using different techniques and enable remote access using Group Policy and Windows Firewall.

Chapter 9, *Deploying Software Restriction Policies and AppLocker*, explains how to deploy default Software Restriction Policy (SRP) or AppLocker rules to ensure only programs installed in protected locations can run. It discusses how to force an application to launch with standard user privileges even if the user is an administrator and how to blacklist an application using SRP or AppLocker.

Chapter 10, *Least Privilege in Windows XP*, covers how to redeploys Windows XP with Least Privilege Security configured and identify problems with applications caused by Least Privilege Security using the Microsoft Deployment Toolkit. It also explains how to mitigate the problems and limitations users may face when running with a Least Privilege Security account and how to handle ActiveX controls in Windows XP.

For More Information:
Chapter 11, Preparing Vista and Windows 7 for Least Privilege Security, explains how to collect and analyze data to identify any potential compatibility problems with Least Privilege Security and software installed on networked PCs using Microsoft's Application Compatibility Toolkit (ACT). The reader will learn how to analyze logon scripts for Least Privilege compatibility, how to prepare a desktop image with Least Privilege Security enabled from the start and deploy the new image while preserving users' files and settings.

Chapter 12, Provisioning Applications on Secure Desktops with Remote Desktop Services, explains how to install the core server roles for Remote Desktop Services in Windows Server 2008 R2 using Windows PowerShell. It also explains how to set up and understand Remote Desktop Licensing and configure Remote Desktop Gateway for secure remote access to applications over HTTPS. This chapter also discusses how to advertise published Remote Applications on Windows 7’s Start menu using Remote Desktop Web Access.

Chapter 13, Balancing Flexibility and Security with Application Virtualization, covers how to sequence an application for streaming and virtualization, and how to set up the App-V Client to work with a server-less deployment model.

Chapter 14, Deploying XP Mode VMs with MED-V, explains how to deploy legacy applications that are not compatible with newer versions of Windows and how to set up Windows XP Mode for Windows 7. It also explains how to configure the different components of MED-V for managing and deploying VMs in a large corporate environment and how to prepare VMs for use with MED-V.

For More Information:
Solving Least Privilege Problems with the Application Compatibility Toolkit

The launch of Windows XP in 2001 heralded the long-awaited transition of consumer-orientated editions of Windows to the NT codebase. Microsoft designed the Windows Application Compatibility Infrastructure as part of Windows XP to help system administrators and home users solve compatibility problems with applications that were designed to run in Windows 98 or earlier versions of the 9x range.

In this chapter we will learn:

- How the Application Compatibility Infrastructure uses shims to modify incompatible applications on the fly
- Why using shims provides the best balance between compatibility and security
- How to create shims using Application Compatibility Toolkit 5.5
- Distributing compatibility databases to devices across the enterprise

For More Information:
Quick compatibility fixes using the Program Compatibility Wizard

In small enterprise environments or on home computers, compatibility fixes can be applied without using the Application Compatibility Toolkit through the user interface. If you want to know how the Windows Application Compatibility Infrastructure works in detail or if you want to deploy fixes to multiple devices, skip straight to the second part of this chapter: Achieving application compatibility in enterprise environments.

Applying compatibility modes to legacy applications

Compatibility fixes can be grouped together to form compatibility modes. Windows XP, Vista, and Windows 7 all come with a default set of compatibility modes out of the box. System administrators can also use Compatibility Administrator to create their own compatibility modes.

Compatibility modes can be applied directly from the user interface either by using the Program Compatibility Wizard in Windows XP, or by right-clicking on an executable file, or a shortcut to an executable file, and selecting Properties from the menu.

Windows XP contains the following compatibility modes:

- Windows 95
- Windows 98 / Windows ME
- Windows NT 4.0 (Service Pack 5)
- Windows 2000

As shown in the following screenshot, the Compatibility tab in Vista and Windows 7 (left) looks a little different from Windows XP (right).
Chapter 3

Privilege level in Vista and Windows 7
Note that the Run this program as an administrator option will be grayed out if an application manifest is included with the program.

Program Compatibility Wizard
Included in Windows Vista and later, the Program Compatibility Wizard allows users or system administrators to test legacy applications running against different compatibility modes.

1. To launch the Program Compatibility Wizard in Windows XP, select Start | Accessories | Program Compatibility Wizard.

For More Information:
To launch the Program Compatibility Wizard in Windows Vista, type the following command into the Search box on the Start menu:
%systemroot%\System32\mshta.exe res://acprgwiz.dll/compatmode.hta

To launch the Program Compatibility Wizard in Windows 7, search for program compatibility troubleshooter in Start | Help and Support.

2. Click Next on the welcome screen in Help and Support Center.
3. Leave the default I want to choose from a list of programs option selected and click on Next.
4. Select the application you want to test, in this case Maxthon, from the Select a program list and click on Next.

5. Choose a compatibility mode from the list, such as Microsoft Windows 95 or Windows 2000, and click on Next.
Compatibility modes set using the **Program Compatibility Wizard** only apply to the currently logged in user. Compatibility settings in Vista and Windows 7 can be applied to the logged in user or to all users. Administrative privileges are required to apply compatibility settings for a given application to all users of a system.

6. If required, select display settings (256 colors or 640 x 480 screen resolution) to be used with the program and then click on **Next**.

7. Click **Next** to test the compatibility mode against your application.

The application will launch and you can test it to see if the selected compatibility mode resolves the identified problems. Once you’ve finished testing, close the application. Back in the **Help and Support Center**, choose to either apply the compatibility settings or try different settings.

**Browsing compatibility settings for each user**

If compatibility modes are set on programs for specific users, you can use **Compatibility Administrator** to browse the settings for each user under the **Per User Compatibility Settings** node. This node only appears if per user settings are found on the system.

**Program Compatibility Assistant**

Introduced in Windows Vista, the **Program Compatibility Assistant** helps to automate the process of applying compatibility fixes to legacy applications by monitoring for known problems when programs are running. This feature runs as a service, and prompts users to apply suggested fixes for applications, either during the setup phase or when running an installed application. The Program Compatibility Assistant can detect the following problems automatically:

- Errors when launching setup programs
- Failures in install routines
- Failures caused by **User Account Control**
- An install needing to run as an administrator
- A control panel applet requiring administrative privileges
- Errors caused because a component is not present in the current version of Windows

For More Information:
Solving Least Privilege Problems with the Application Compatibility Toolkit

- Notifying users about unsigned drivers on 64-bit versions of Windows
- Matching applications against a list of programs with known problems and notifying the user at program startup

The Program Compatibility Assistant intercepts an installation routine, prompting the user to try again using recommended settings.

Disabling the Program Compatibility Assistant

The Program Compatibility Assistant is intended to help home users resolve problems with legacy applications. In an enterprise environment, to avoid potential problems with messages generated by the Program Compatibility Assistant, you should consider disabling the service in Group Policy. The Turn off Program Compatibility Assistant setting can be found in the Group Policy Management Editor under Computer or User Configuration | Policies | Administrative Templates | Windows Components | Application Compatibility.

For More Information:
Excluding executables from the Program Compatibility Assistant

The Program Compatibility Assistant doesn't monitor programs that have an application manifest that marks them as compatible with Vista or Windows 7. However, if you want to exclude a program but don't want to disable the Program Compatibility Assistant completely, you can create a `REG_MULTI_SZ` registry value named `ExecutablesToExclude` under the following key:

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility Assistant
```

For More Information:
Achieving application compatibility in enterprise environments

As new versions of Windows are being developed, improvements and changes mean that some applications designed for earlier editions of the operating system may experience compatibility problems in the later versions. Rather than restricting innovation and development of the OS in favor of compatibility, the Windows Application Compatibility Infrastructure was designed to allow the OS to move forward while retaining compatibility with legacy applications.

Compatibility fixes

Compatibility fixes, sometimes also known as shims, target specific legacy applications and allow them to run in current versions of the operating system. The advantage of using shims is that rather than maintaining legacy code in the operating system to ensure that all applications run without modification, which in turn makes the OS inherently more complicated and insecure, the legacy application code is modified by a shim instead of the OS.

Modifying applications using shims

Shims intercept Win32 API calls from legacy applications, as defined by system administrators, and then modify the call before passing the code to Windows for execution.

A legacy application fails to run in Windows Vista:

A legacy application successfully runs in Windows Vista:

For More Information:
Enhancing security using compatibility shims

As far as Windows is concerned, the modified code presented to it by the shim is code from the application itself. Therefore, it's important to note that any application code modified by a shim runs in the same security context as the application. As shim code is not part of the operating system, it can't be used to bypass Windows security features.

It's common place for system administrators to either grant administrative rights to users or loosen Access Control Lists (ACLs) on files, directories, or registry keys to force legacy applications to run on newer versions of Windows. This increases the attack surface and makes it more likely that systems will fall prey to some kind of security threat. Shims on the other hand, don't require users to hold any additional privileges or modification of security principals. Therefore, shims are inherently more secure than workarounds such as relaxing ACLs.

One example of improved security when using shims is if an application needs to write to a protected area of the file system. Rather than loosening the ACLs on the given directory to ensure that write operations made by the application succeed, a shim can be used to redirect the request to a virtualized copy of the protected directory.

Deciding whether to use a shim to solve a compatibility problem

From a security standpoint, a shim is always preferable to loosening ACLs on security principals for solving compatibility problems. While failed I/O operations made by applications are the most common reason why legacy programs fail in Windows XP, shims provide a solution for many additional compatibility problems that loosening ACLs alone cannot solve.
Vendor support
Bear in mind, when considering whether to deploy shims in your organization, that because the application's behavior will essentially be modified, you may invalidate any support agreement with the vendor. If vendor support is absolutely critical for your line-of-business application, you should check in advance whether support will be provided if a shim is used to solve a compatibility problem. Better still, ask the vendor if they will modify the application to support the current version of Windows.

In-house applications
In the same way that shims are preferable to loosening ACLs, modifying the original application code is better than using a shim, as it removes an extra layer of complexity. Applications developed in-house should be modified by the developers and then redeployed. Shims can be used as a temporary solution if compatibility problems with line-of-business applications are likely to block an upgrade project for a significant time period.

Kernel-mode applications
Shims cannot be used to solve compatibility problems with drivers and other software that hooks deep into the operating system, such as antivirus programs. Shims run in user-mode, so drivers and kernel-mode software must be modified at source if there are compatibility problems with the latest version of Windows.

Creating shims for your legacy applications
Custom databases are created using Compatibility Administrator, part of the Application Compatibility Toolkit, and contain:

- **Compatibility fixes (shims):** Compatibility fixes can be applied to specific applications installed on devices across your organization.
- **Custom compatibility modes:** Custom compatibility modes can be added to the default list that ships with Windows.
- **Apphelp messages:** Apphelp messages inform users about compatibility problems with specific software and optionally block programs from running.

Shims can be packaged with each application as you deploy software on your network, or you can manage a central database that contains all the shims required for your applications and deploy the database to every machine.
Viewing the fixes included in a compatibility mode
Follow the instructions to add a compatibility fix to a custom database using **Compatibility Manager**, but instead of pressing Ctrl+P to add a fix, press Ctrl+L to add a compatibility mode. At the bottom of the **Create a Custom Compatibility Mode** dialog box, click on **Copy Mode**. The fixes included in the selected compatibility mode are then displayed under **Compatibility fixes for the mode** on the right.

On small networks, where only a handful of legacy applications require shims for compatibility purposes, it may be feasible to package a **custom database** with each application. This requires a developer or system administrator who is familiar with application packaging technology, and each package must be tested thoroughly. Shims can be difficult to track if they're embedded into application installer packages. If many applications require shims, a more scalable solution is to maintain a single central custom database and update it on client machines as necessary. Your custom database can be deployed using a Group Policy startup script. Very large organizations can manage several sets of custom databases and deploy them to defined categories of users for ease of administration.

Custom databases must be deployed under the security context of a user with administrative privileges.

You could consider adding a custom database to the OS image that your organization uses to deploy Windows. Custom databases then only need to be updated using Group Policy when a change is made to the central database.

**Solving compatibility problems with shims**
This section describes the compatibility fixes that are most commonly used to solve application compatibility problems caused by least privilege security.

You should note that you cannot create your own compatibility fixes, only compile collections of fixes (custom databases) modified to work with specific applications in addition to the default system database of customized fixes.
LUA compatibility mode fixes in Windows XP

The LUAredirectFS and LUAredirectReg compatibility fixes provide the two main workarounds for forcing legacy applications that write to protected areas of the file system and registry to work correctly using a standard user account.

LUAredirectFS

If a legacy application tries to write to a protected system folder such as c:\windows or c:\program files, the LUAredirectFS compatibility fix can be used to redirect all writes and subsequent read operations to the user's profile: %systemDrive%\Documents and Settings\%username%\LocalAppData\Redirected\drive\filepath. When configuring this fix for a specific application, you need to launch and run the program during the configuration process so that the Compatibility Administrator can detect which files should be redirected.

LUAredirectFS_Cleanup

Intended for use when an application is removed from a computer, the LUAredirectFS_Cleanup compatibility fix can be used to remove application files that were redirected to user profiles.

LUAredirectReg

Similar to LUAredirectFS, LUAredirectReg can be used to detect when a legacy application tries to write to the protected HKEY_LOCAL_MACHINE registry hive and redirect the write and subsequent read operations to the user's HKEY_CURRENT_USER hive. The application being fixed must be launched and run during the configuration process so that Compatibility Administrator can detect which keys should be redirected.

LUAredirectReg_Cleanup

For use when an application is being uninstalled, the LUAredirectReg_Cleanup fix removes redirected registry keys from users' profiles.

LUATrackFS

The LUATrackFS compatibility fix is for monitoring which files and folders a legacy application accesses when run. This information is recorded in a file so that developers can modify the application to work with a standard user account.

For More Information:
Creating your own custom database

Start by installing the Application Compatibility Toolkit 5.5 (ACT). Refer to Chapter 11, Preparing Vista and Windows 7 for Least Privilege Security, for installation instructions. The only component of ACT required to create a custom database is the Compatibility Administrator. There's no need to install SQL, SQL Express, or AppVerifier to run the Compatibility Administrator tool.

Compatibility fixes are specific to different versions of Windows, so if you want to create a custom database with shims for Windows XP, you should create your custom database using Compatibility Administrator installed on Windows XP.

Windows XP-specific compatibility fixes are not available when Compatibility Administrator is installed on Vista or Windows 7.

Maxthon on Windows XP

A legacy version of Maxthon, a replacement shell for Internet Explorer, doesn't retain customizations made by standard users because settings are stored in files in the protected Program Files directory. Let's create a custom database with compatibility fixes configured to target Maxthon that redirects file and registry I/O operations to virtualized per-user file and registry stores.

1. Log in to Windows XP as an administrator and select Start | All Programs | Microsoft Application Compatibility Toolkit 5.5 | Compatibility Administrator.

2. In the left pane of Compatibility Administrator, you'll see System Database, which contains Applications (see the following information box), Compatibility Fixes, and Compatibility Modes, and a new custom database.

For More Information:
The Applications folder in the System Database contains customized compatibility fixes that address known problems with commonly used programs.


4. In the Create new Application Fix dialog box, type Maxthon in the Name of the program to be fixed field.

5. To the right of Program file location, click on Browse and locate the Maxthon executable file in the Program Files directory. Select it and click on Open.

6. In the Create new Application Fix dialog box, click on Next.

7. We're going to apply specific compatibility fixes to Maxthon, so under Operating System Modes select None and in the additional compatibility modes list on the right select LUA and then click on Next.

8. As we have already selected LUA (Least privilege User Account) in the previous dialog box, the appropriate compatibility fixes, LUARedirectFS, and LUARedirectReg are already selected for us. Therefore, no configuration is required on this screen and we can click on Next to continue.

For More Information:
LUARedirectFS and LUARedirectReg are specific to Windows XP. Starting in Vista, UAC File and Registry Virtualization provides automatic redirection. The CorrectFilePaths or VirtualRegistry compatibility fixes can be used for more granular control over redirection.

9. On the Matching Information screen, we can control how the shim identifies the Maxthon executable. Let's leave the default settings and click Next.

10. Select Yes, Customize these fixes now and then click Finish.

11. Click Next on the Monitor the Program screen.

12. In the Test run application dialog box, click OK to launch Maxthon.

13. Once Maxthon has been launched, select Maxthon Options from the Options menu, check Allow only one instance of Maxthon, and click OK.

14. Now, close Maxthon. On the Exclude File Extensions screen, leave the File extensions to be excluded field blank and click Next.

15. The Edit the File Redirection List screen shows us all the files that Compatibility Administrator has determined to be redirected to a per-user location. Let's select all the files listed and click Next.

For More Information:
16. Click **Finish** on the **Redirection Locations** screen.

As **Compatibility Administrator** didn't detect any failed registry I/O operations, there was no screen showing a list of registry key redirections.

17. Maxthon will now appear under the **Applications** node in the Maxthon custom database. Let's save the Maxthon database by pressing Ctrl+S. Name the database Maxthon and save it to your desktop.

18. Right-click on the Maxthon database in **Compatibility Administrator** and select **Install** from the menu.

19. Click **OK** in the **Compatibility Administrator** dialog box. This confirms that the database was successfully installed.

20. To test whether the shim is being applied, we need to start Maxthon as a standard user and see if it retains our customizations. Right-click on the Maxthon shortcut on your desktop, or the maxthon.exe file in c:\program files and select **Run as...** from the menu.

21. In the **Run As** dialog box, select **The following user** and then type the username and password of a local or AD user who is a standard user on the machine.

22. Maxthon will start. Select **Maxthon Options** from the **Options** menu, check or uncheck **Allow only one instance of Maxthon**, and click **OK**.

23. Close Maxthon and then restart it using **Run As** and the same limited user account.
You should now see that Maxthon retains configuration changes made by a standard user as the shim is redirecting the file I/O operations to a per-user location.

**Working with other commonly used compatibility fixes**
The compatibility fixes demonstrated in this section are available in all versions of Windows, starting with XP, unless otherwise stated.

**ForceAdminAccess**

| Problem | The application has a hard-coded check to see if the logged in user is a member of the local Administrators group. |
| Fix     | Apply the ForceAdminAccess compatibility fix using Compatibility Administrator. |

1. Log in to Windows as a standard user. Start **Standard User Analyzer (SUA)** in **Developer and Tester Tools** in the Application Compatibility Toolkit under **All Programs** in the **Start** menu.
2. On the **App Info** tab, click on **Browse** and locate the **CorporateApp** executable file.

**AppVerifier**

Remember that **AppVerifier**, which comes with the Application Compatibility Toolkit 5.5, must be installed to use **Standard User Analyzer**.

For More Information:  
3. Click Launch, enter administrator credentials into the Run As dialog box, and click OK to elevate the SUA service. If SUA prompts that **All existing AppVerifier logs will be deleted, do you want to continue?**, click Yes.

4. Enter administrator credentials into the Run As dialog box and click OK again to elevate the CorporateApp executable file.

5. Run CorporateApp and make sure that the function that checks for administrative privileges runs. Now, close CorporateApp.

6. Return to the **Standard User Analyzer** program and you'll see that the issue count on the **Token** tab has increased to **1**.

7. Click the **Token** tab to see detailed information about the problem. Double-click the entry on the **Token** tab and three extra windows will appear with details about the problem.

In the **Detailed Information** box, you'll see that SUA has identified that CorporateApp calls the **IsUserAnAdmin** API to check for administrative privileges. This information is important, because the **ForceAdminAccess** check fix in Windows XP only intercepts the **CheckTokenInformation** API. The **ForceAdminAccess** fix in Vista and Windows 7 can intercept the following APIs that might be used by applications to check whether the user is a member of the local Administrators group:

- AccessCheck
- CheckTokenMembership (**CheckTokenInformation** in Windows XP)
- RegOpenKeyExA
- GetTokenInformation
- IsUserAnAdmin
- NetUserGetInfo
- SetActivePwrScheme

The **ForceAdminAccess** fix will not intercept the **IsUserAnAdmin** API in Windows XP. So, it can only be used to solve our problem with CorporateApp in Vista and Windows 7.

1. Right-click on **Compatibility Administrator** in the Start menu under All Programs | Microsoft Application Compatibility Toolkit 5.5 and then click on Run as from the menu.

2. In the **Run As** dialog box, click on **The following user:** and enter the username and password of a local or domain administrator account.
3. Under the **Custom Databases** node in **Compatibility Administrator**, click **New Database**, press Ctrl+R, and rename it as **CorporateApp**.

4. Click on the **CorporateApp** database and press Ctrl+P to create a new fix. Under **Program information**, enter the application's details including the path to the main executable file.

5. Click on **Next** to continue. As we know the specific compatibility fix that we want to apply, select **None** under **Operating System Modes** on the **Compatibility Modes** screen and then click **Next**.

6. On the **Compatibility Fixes** screen, scroll down the list of fixes and check **ForceAdminAccess**.

7. Now, click **Parameters** and then type an asterisk (*) in the **Module name** field and click **Add**.

---

For More Information:
Files located in C:\windows\system32 are excluded from the compatibility infrastructure by default. So if an application calls an API from a file located in the system32 folder, any shims you've configured will be ignored. While this default configuration doesn't usually cause a problem, applications coded with Visual Basic (VB) 6 or earlier may ignore shims as the VB runtime is located in system32. CorporateApp is a Visual Basic application, so we must ensure that all modules launched from the system32 folder are included.

8. Click OK in the Options for ForceAdminAccess dialog box and then click on Next to continue in the Create new Application Fix wizard.


10. Select the CorporateApp database in Compatibility Administrator and click the Save icon at the top of the window.

11. Save the database as corporateapp.sdb to C:\

12. To test the application, right-click the CorporateApp database in Compatibility Administrator and select Install from the menu.

13. Click OK in the confirmation dialog box.

You can now run CorporateApp as a standard user on this system and the hard-coded check for administrative privileges will not prevent the program from running.

**CorrectFilePaths**

| Problem | The application tries to create or write to a file in a protected system directory, such as C:\Program Files, where the application is installed. |
| Fix     | Apply the CorrectFilePaths compatibility fix using Compatibility Administrator. |

Log in as a standard user but start Compatibility Administrator with administrative privileges, as described in the steps for configuring ForceAdminAccess fix and follow these instructions:

1. Click Open at the top of Compatibility Administrator and open the CorporateApp database (c:\corporateapp.sdb).

2. Expand the CorporateApp database under the Custom Databases node and click CorporateApp in the Applications folder in the left pane.

For More Information:  
3. Click CorporateApp.exe at the top of the right pane. Now select Edit | Modify | Edit Application Fix.

4. Click Next past the Program information screen, and Next again past Compatibility Modes.

5. On the Compatibility Fixes screen, scroll down the list of fixes and check CorrectFilePaths.

6. Click Parameters and then type the path of the file to be redirected and its new location as follows: old path;new path into the Command line field. For example, CorporateApp is trying to create and write to a file called config.ini in c:\program files\CorporateApp. We need to redirect the file to a location where standard users have permission to write. So an example redirection might be as follows:

   %programfiles%\CorporateApp\config.ini;%userappdata%\config.ini

   Multiple file redirections
   One CorrectFilePaths application fix can contain multiple redirections on the command line separated by a space.

7. Once you've added the file redirection, type an asterisk (*) in the Module name field and click Add to include all modules in the system32 folder.

8. Click OK in the Edit Application Fix dialog box and then click Next to continue in the Create new Application Fix wizard.


10. Click Save at the top of the Compatibility Administrator window and then right-click the CorporateApp database and select Install from the menu.

11. Click OK in the confirmation dialog box.

**VirtualRegistry**

The VirtualRegistry fix is one of the most flexible compatibility fixes and can be modified by specifying command-line options. In this section, we'll look at the functions of VirtualRegistry that are of most interest in solving problems related to least privilege security.
ADDREDIRECT

The ADDREDIRECT option was introduced in Windows Vista. It allows you to redirect a specific registry key to a different location as specified in the command-line options. Let's have a look at how to add command-line options to the VirtualRegistry fix. In this example, we'll redirect a registry key that reads and writes to the protected HKEY_LOCAL_MACHINE hive to the HKEY_CURRENT_USER hive, which standard users have read and write access to.

Follow these instructions, logged in as a standard user but running Compatibility Administrator with administrative privileges:

1. Click Open at the top of Compatibility Administrator and open the CorporateApp database (c:\corporateapp.sdb).
2. Expand the CorporateApp database under the Custom Databases node and click CorporateApp in the Applications folder in the left pane.
3. Click on CorporateApp.exe at the top of the right pane. Now select Edit | Modify | Edit Application Fix.
4. Click Next past the Program information screen, and Next again past Compatibility Modes.
5. On the Compatibility Fixes screen, scroll down the list of fixes and check VirtualRegistry.
6. Click Parameters at the top of the screen and type the following option into the command line field:

`ADDREDIRECT (HKLM\SOFTWARE\Microsoft\PHealth\ErrorReporting\DW^HKCU\Software\DemoApp\DW)`

7. Type an asterisk (*) in the Module name field and click Add to include all modules in the system32 folder. Click OK to continue.

You'll now see the command line added to VirtualRegistry on the Compatibility Fixes screen and you can continue as per the instructions in the Create your own custom database section.

RedirectFiles and RedirectRegistry

The RedirectFiles and RedirectRegistry compatibility fixes are designed specifically for Internet Explorer. CorrectFilePaths and VirtualRegistry should be used to redirect files and registry keys.

For More Information:

Working with custom databases

When updating or maintaining a central custom database in your organization, you may need to deploy or redeploy the database, and merge new applications fixes.

Adding new shims to your custom database (merging custom databases)

If you decide to maintain a central custom database that contains all your application shims, it's likely that you'll need to update it from time to time when new programs are deployed. Compatibility Administrator allows you to open two databases simultaneously and copy and paste fixes between them. Let's create a new application fix and add it to a custom database. First you should follow the steps described in Creating your own custom database to add a new shim to a custom database and save it. You cannot create a shim without a database.

1. In Compatibility Administrator, open the database that contains the shim you want to add to the central database, in this case Shim to merge, and open the central database.
2. In this example, we want to merge the fix for Demoapp into the central database that already contains a fix for Maxthon. Right-click on the Demoapp fix under Applications and select Copy from the menu.
3. Now, right-click the Applications folder under the central database node and select Paste from the menu.
4. The fixes for Demoapp and Maxthon now appear together in the central database and you can click Save at the top of Compatibility Administrator to save changes to the central database.

Temporarily disabling compatibility fixes

If an application is updated to address a known compatibility problem that one of your shims addresses, you may want to temporarily disable a fix in the custom database while testing the new version of the program.

1. Click Open at the top of Compatibility Administrator, locate your central database .sdb file and click Open.

For More Information:
2. Select the fix you want to disable in the Applications folder in the left pane and then select Disable Entry from the Database menu at the top of the window.

![Compatibility Administrator interface](image)

3. To re-enable the disabled fix, repeat the procedure as described earlier, but select Enable Entry from the Database menu instead.

**Installing a custom database from Compatibility Administrator**

When developing shims using Compatibility Administrator, you will need to test if the fixes solve the issue(s) at hand. To do this on the same machine on which Compatibility Administrator is running, you must install the custom database that contains the fixes.

1. To install a custom database locally, right-click on your database in the left pane of Compatibility Administrator and select Install from the menu.
2. To uninstall a custom database, expand the Installed Databases node in the left pane of Compatibility Administrator, right-click on the database to remove, and select Uninstall from the menu.

---

**For More Information:**

Database locations
The System Database, which is included as part of the operating system and contains shims that resolve known compatibility problems with popular applications, is located in the AppPatch directory in c:windows and is updated by Microsoft via Windows Update. The AppPatch directory contains a subdirectory called Custom, which contains any custom databases installed on a machine.

Deploying a database to multiple devices
While Compatibility Administrator can be used to install or remove a database on the machine where it's installed, Microsoft provides a command-line utility called sdbinst.exe for deploying custom databases across multiple machines.

Finding the GUID of custom database
Once a custom database has been saved in Compatibility Administrator, it's assigned a Globally Unique Identifier (GUID) that can be referenced from the command line when installing or removing databases. To find the GUID of a given database, right-click the database in the left pane of Compatibility Administrator and select Properties from the menu. The database's GUID will be shown in the Database Properties window.

For More Information:
SDBINST command-line switches

There are four basic switches that can be used with sdbinst.exe and they are outlined in the following table:

<table>
<thead>
<tr>
<th>Switch</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n &quot;filename&quot;</td>
<td>Used to specify the filename of the custom database to install or uninstall.</td>
</tr>
<tr>
<td>-g GUID</td>
<td>Used to specify the custom database by GUID to install or uninstall.</td>
</tr>
<tr>
<td>-q</td>
<td>Silent install. No information is presented to the user during an install or uninstall operation. Used mainly for scripts.</td>
</tr>
<tr>
<td>-u</td>
<td>Performs an uninstall operation as opposed to the default install.</td>
</tr>
</tbody>
</table>

For example, we can install our custom central database from the command line using the following switches:

```
sdbinst.exe -g \{1810c392-765d-457f-873e-8e176ee09298\}
```

Alternatively, we can uninstall the same database by filename silently:

```
sdbinst.exe -u -q -n "\\networkshare\centralshimdb.sdb"
```

SDBINST must be run with administrative privileges to install and remove custom databases.

Distributing or updating a custom database using Group Policy

Custom databases can be included as part of your organization's standard image. Alternatively, SDBINST can be distributed using a Group Policy computer startup script. The following batch file calls reg.exe and sdbinst.exe, both command-line programs that are included in Windows XP and later by default, to first check the existence of a custom database by verifying the presence of an uninstall registry key, and if the registry key is not found, install the database:

```
reg.exe query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion \Uninstall\{GUID}\.sdb" &&(goto: eof)
sdbinst.exe -q %~dp0centralshimdb.sdb
```

For More Information:

The `%~dp0` variable

%~dp0 sets the working directory to the same path as the location of the batch file. This enables us to drop the `centralshimdb.sdb` file into the same location as the batch file called from the Group Policy startup script without having to specify the exact path.

If you want to update an installed custom database, as long as the new version of the database has the same GUID as the one to be replaced, `sdbinst.exe` will automatically remove the database and install the new version. In an update situation, you will need to remove the logic in the batch file that checks for the presence of an installed database with the specified GUID. This can be easily achieved by adding a hash (`#`) in front of the `reg.exe` command to temporarily prevent it from running.

```bash
#reg.exe query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion \Uninstall\{GUID}.sdb" & & (goto: eof)
sdbinst.exe –q %~dp0centralshimdb.sdb
:eof
exit
```

Now that we've got our batch file ready, let's create a Group Policy Object and add it as a computer startup script. First, save the batch file in Notepad and call it `installdb.bat`.

1. Open the **Group Policy Management Console** as a domain administrator from **Administrative Tools** on the **Start** menu and expand your **Active Directory** forest and domain in the left pane.
2. Locate your **Clients** OU, right-click it, and select **Create a GPO in this domain, and Link it here** from the menu.
3. In the **New GPO** dialog box, type **App Compat DB startup script** in the **Name** field and click **OK**.
4. Expand the **Clients** OU in the left pane, right-click the **App Compat DB startup script** GPO, and select **Edit** from the menu.
5. In the **Group Policy Management Editor** window, expand **Policies** under **Computer Configuration** in the left pane.
6. Expand **Windows Settings** and click **Scripts (Startup/Shutdown)**.
7. Double-click **Startup** in the right pane and click **Show Files**....
8. A new window will open with the location of script files for this Group Policy Object. Copy the `installdb.bat` and `centralshimdb.sdb` to the Startup folder and then close the window.

9. Click Add... in the Startup Properties dialog box and then Browse... to the right of the Script Name field.

10. Select the `installdb.bat` file in the Browse dialog box and click Open.

11. Click OK in the Add a Script dialog box and the `installdb.bat` file should now appear in the Startup Properties window.

12. Click OK in the Startup Properties dialog box and close the Group Policy Management Editor.

Now if you restart a machine in Clients OU, the Startup Script will run as the computer boots.

Always wait for the network at computer startup and logon to the computer

Unless configured otherwise, it may take several reboots before the Startup Script runs. This is because, by default, Windows does not wait for the network to respond to all requests, speeding up the boot process. The Always wait for the network at computer startup and logon to the computer policy setting is located in Computer Configuration | Administrative Templates | System | Logon. If a device is configured to connect to a wireless network only after a user logs on, Group Policy startup scripts will not run.

Summary

In this chapter, we’ve discussed all the basic concepts of the Windows Application Compatibility Infrastructure and how to manage compatibility fixes in an enterprise environment. Before continuing to the next chapter you should:

- Understand how to use Compatibility Administrator to create a custom database
- Be able to deploy and update a custom database to multiple devices across a network using Group Policy in conjunction with the `sdbinst.exe` command-line utility

For More Information:

• Know how to update and add shims to a **custom database** using **Compatibility Administrator**
• Be familiar with the most common compatibility fixes used for solving least privilege security problems in legacy applications and how to deploy them

In the next chapter we'll look more closely at the technologies which compromise **User Account Control** in Vista and Windows 7.
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.