



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

## Interested in learning more?

Check out the list of upcoming events offering  
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"  
at <http://www.giac.org/registration/gsec>

# Adapting Windows Security for Legacy Applications

Edward Myers, SANS Track 1  
GSEC Practical, Version 1.4, Option 1

5/10/2004

## Abstract:

### *Lockdown your Computers by UNLOCKING Rights*

SECEDIT can use templates to quickly modify Windows XP security permissions on the NTFS file system, the registry, services, groups, and Local Security Policies. Generally, these templates are generated to enhance security by adding tighter restrictions to Windows XP based computers. Counter-intuitively, the SECEDIT templates can also loosen security on these same items for the greater good - namely removing Administrator rights from users on client computers. A common problem with enhanced (tighter) security is that legacy applications will not run without Power User or Administrator rights. Although granting these rights solves the problem, it is generally overkill, risky, and potentially costly.

This document explores the issues with legacy application rights requirements and provides information to discover the exact requirements an application needs. Once these rights are understood, a security template can be created and applied to give the explicit rights required without providing users with "Power User" or "Administrator" rights. Effectively, security will be loosened in specific areas to prevent global security risks.

## Table of Contents:

- A. Introduction**
- B. Problems with Legacy Windows Applications**
  - 1. Definition of “Legacy” Application
  - 2. Administrator Rights – Why It Is So Risky
  - 3. Common Problems With Legacy Applications
  - 4. Proposed Solution – Adapt Security
- C. Introduction to SECEDIT**
  - 1. The SECEDIT Tool Command Line
  - 2. Using SECEDIT To Adapt Security
- D. Discovery Tools**
  - 1. The Basic Discovery Process
  - 2. Tools 1: Error Dialog Boxes, Application Logs, and Windows Event Viewer
  - 3. Tools 2: Sysinternals.com – REGMON, FILEMON
  - 4. Tools 3: Repackaging Tools – SMS Installer
  - 5. Other Techniques
- E. Building Security Templates**
  - 1. File System and Registry Security Changes
  - 2. Services, Users, Groups
  - 3. Copy the Template
- F. Applying and Testing Results**
  - 1. Applying the SECEDIT Template
  - 2. Testing Methodology
  - 3. Modify Results based on testing
  - 4. Applying the results
- G. Conclusion**
- H. References**

## **INTRODUCTION**

This document describes a process to adapt Windows client security to accommodate “Legacy Applications” which do not honor the security model for modern Windows based operating systems.

Section 1 discusses the issues with Legacy Applications, the choice to adapt security to accommodate them when there is no other choice.

Section 2 introduces Microsoft’s SECEDIT tool and describes what it can do to adapt security.

Section 3 provides an overall process that can be followed to discover the exact changes that are required for a particular legacy application and provides recommendations for software tools that can be used during the discovery process.

Section 4 describes the process of building a security template for SECEDIT, using the facts from the discovery process.

Section 5 reviews the testing and application of the security changes.

This document addresses Windows Client Operating systems, and provides a basic process with examples. No specific application was reviewed, as it was not the intention to highlight any particular vendor’s mistakes. This document will provide guidance and a proposed series of steps to address legacy applications.

## **SECTION 1: Problems with Legacy Windows Applications**

### **Definition of “Legacy Application”**

The term “Legacy Application” is open to significant interpretation. In this document, the term refers to any application with one or more of the following characteristics:

- Does not account for NTFS file system permissions
- Does not account for Windows NT Registry ACL permissions
- Requires manipulations to System Services, without a built-in security mechanism for un-privileged accounts
- Requires write access to protected Operating System locations

A less detailed definition is: A Legacy Application is any application which does not fully function when run by a user with normal user rights (non-Administrator) on a Windows NT based computer.

Generally, these legacy applications were designed for unsecured operating systems, such as Windows 95, 98, or Millennium (referred to as Win 9x.) Windows NT 4.0, 2000 and XP/2003 (referred to as Windows NT based operating systems or WinNT) enforce a security model that was not previously seen in Win9x. [1]

In some cases, these applications have a poor design. Developers may be writing applications with full administrator rights on their workstations and they test using administrator rights. The developer may be unaware that he is designing an application that will fail in a secured environment.

Regardless of the source of the problem, there are four choices for dealing with these applications: ignore, replace, retire, or adapt. Ignoring the legacy applications will require providing normal users with full Administrator rights to their PCs. The risks of this choice should be weighed against the cost of working around the problem, using SECEDIT. Replacing the application can mean purchasing or building a new version of the application, or purchasing a competitive version with similar features. Retirement of an obsolete application is also a choice.

The final choice is to adapt. This document deals with adapting to the limitations of an existing application which can not be replaced or retired. I do not suggest ignoring the problem, or granting Administrator rights to all users. The next section will discuss the risks of ignoring the granting of rights.

### **Administrator Rights – why it is so risky?**

One of the first questions asked by people about this issue is generally, “Why is it important to restrict users?” This is an excellent question because locking down desktop operating systems is a timely and costly expense. Organizations that are struggling with unlocked desktops may be reluctant to spend the time, energy and cost associated with reviewing and correcting each user rights issue which prevents locking desktops down.

The most basic answer is this: Defense-in-depth [2]. Defense in depth is a concept that no single method of network security is perfect. Relying on single point defense is asking for trouble. Building secure servers, secure networks, and secure firewall protected connections to the internet are common practices in a security aware organization. However, if a user can inadvertently install a virus or worm on their computer, then none of these defensive items will be able to completely protect the network resources. Every area of the network that the user can access is now vulnerable, using standard user rights. Files on a server can become corrupted or infected, email can be infected, and denial-of-service attacks are potential results from users. Although it is possible to minimize or neutralize those attacks, it would be much simpler to layer the defense and rely on **preventing** the user from these dangers from the start.

Another significant reason is that it is **cheaper** to support a large infrastructure when the computers are “locked down.” Preventing users from installing unauthorized software, modifying critical operating system files, or modifying their computers into a non-standard configuration means lower support cost. CIO Magazine had this to say [3]:

“Granting users extended rights on the desktop will increase support costs by 10 percent or more and create significant security risks. However, some applications or functions only work if users have elevated rights, so IT managers in most organizations will need to make some exceptions. IT should enforce strict policies to limit the number of exceptions and take steps to limit administrative access to desktops for IT personnel and users alike.”

CIO Magazine continues to describe a typical company study (in this case, from Marathon Oil). The results of locking down computers:

- 10% of the organization was unlocked, yet they represented 50% of the help desk call volume
- The unlocked desktops had help desk trouble tickets open 3 times longer than the locked down desktops.

Removing Administrator rights is the first and most critical step to defending networks, controlling costs and providing a secure computing environment.

Adapting security to accommodate legacy applications is generally the safest and cheapest solution. The next section will discuss what causes legacy applications to desire Administrator rights.

### **Common Problems with Legacy Windows Applications**

Applications that were not written with the WinNT security model in mind generally will experience one or more of the following problems:

- Attempting to write to the root directory of the system drive (%SystemDrive%)
- Attempting to write to one of many protected locations in the Windows directory (%SystemRoot%), especially the SYSTEM32 directory.
- Attempting to write to protected portions of the Windows Registry, such as the HKEY\_LOCAL\_MACHINE hive.
- Create or manage system services or service accounts

All of these actions are possible when an application is run using an Administrator account. However, these actions generally can not occur when a user has “User” level rights on a computer.

Since things work when you have Administrator rights, vendors and developers will generally make a blanket statement:

“You must be an Administrator to run [the application].”

This is an incorrect conclusion, and it is an unacceptable requirement. Unfortunately, all too often, this is accepted as a solution and users are elevated to Administrator without an investigation of the actual problem.

The bottom line: No application designed for a general user community should require Administrative rights to properly run. Anyone that states otherwise is avoiding the issue of redesign, rewrite, and workstation security in general.

### **Proposed Solution – Adapt Security Permissions**

Once a choice is made to adapt to the application limitations, a method is needed to modify the workstation security elements. This method must make changes without otherwise compromising general Windows workstation security. The goal is to modify the minimum number of security elements which allow the application to properly run, without relaxing security any more than is required.

Fortunately, Microsoft has provided a powerful tool which will facilitate adapting Windows workstations to relax security permissions, allowing the adaptation of security to accommodate legacy applications. The next section introduces the SECEDIT tool and describes its features.

## **SECTION 2: Introduction to SECEDIT**

The general types of problems we will encounter with legacy applications are known and a method to implement the changes is required. Fortunately, a solution is available: SECEDIT. SECEDIT is a command-line tool to apply security changes to a computer. It uses command line parameters and template files (in an INF file format) to determine the actions it will take to make these necessary changes. [4]

### **The SECEDIT Tool Command Line:**

The fact that SECEDIT is a command-line tool makes it an ideal choice for applying changes in a large organization. Distribution tools can be used to apply the desired changes, automatically, on hundreds or thousands of similar computers within an organization. The Microsoft help file for SECEDIT shows us following items may be modified:

Area name	Description
SECURITYPOLICY	Local policy and domain policy for the system, including account policies, audit policies, and so on.
GROUP_MGMT	Restricted group settings for any groups specified in the security template
USER_RIGHTS	User logon rights and granting of privileges
REGKEYS	Security on local registry keys
FILESTORE	Security on local file storage
SERVICES	Security for all defined services

The problems encountered with legacy applications can all be addressed with the SECEDIT tool.

The command-line syntax for using the SECEDIT tool, to modify computer security is [5]:

```
secedit /configure /db FileName [/cfg FileName ] [/overwrite][/areas  
area1 area2...] [/log FileName] [/quiet]
```

The vital part of the command-line listed above is the “/cfg Filename” portion. This is the INF template which contains the changes needed. Before changes can be applied, three things must be determined:

- What needs to be changed
- How should each item be changed
- What is the correct format for those changes in the INF file.

## Using SECEDIT to Adapt Security

A review of the options for SECEDIT indicates that it will meet the needs to modify security elements of workstations. It provides granular modifications allowing the change of security rights for items that are causing issues within a specific legacy application. [6]

Before using the SECEDIT tool, there must be an understanding of what needs to be changed, and how it needs to change. This may be granting “Users” read/write access to a file or directory. It might include relaxing permissions on certain registry keys. SECEDIT will not be used until the review of the specific application is complete.

The next section describes a discovery process and recommends several tools to use to determine **what** and **where** a security change will be needed. SECEDIT is the tool of choice for applying these changes, allowing the adaptation of security rights to the particular needs for this application.



## **SECTION 3: Discovery Tools**

To remove Administrator rights from users that run legacy Windows applications, testing without Administrator rights must be performed and problems found and fixed. This process can be tedious, and in many cases there is no roadmap to follow from the vendor or developer. In cases like this, to correct the application, problems discovery must occur first.

Fortunately, several tools can make the discovery process less difficult. This section will explore the use of several effective discovery tools. There are other tools that are just as effective in the discovery process. The Systems Administrator will need to decide which tool is best for each discovery process.

The method to determine “What needs to change” may be as easy as pressing the vendor or developer with detailed questions about an application. Questions to ask might include:

- What files are updated by the application?
- What registry keys are updated by the application?
- What specific items are causing the need for Administrator rights?

That can save some time in this process. When all else fails, discovery tools are available to determine the source of the problem.

### **The Basic Discovery Process**

The basic discovery process is repetitive, in the following order:

1. Launch discovery tool in capture mode (Admin)
2. Launch test application or process, and repeat steps which caused failure (non-Admin)
3. Review captured results, and detail security changes which are required
4. Apply captured security results modifications (relax specific security items – Admin))
5. Repeat all steps until application functions properly (non-Admin)

Note: Step 2 must be run using a non-Administrator account. Steps 1 and 4 will generally require Administrator rights.

Steps 4 and 5 (apply changes manually and retest) are vital! Just because an error is found and a change is applied to the test workstation, there is no guarantee that the application will now work. Frequently, the application will generate an error message on its first failure, and then exit. The first pass of the discovery process should discover and correct this issue; however retests are required to ensure that the application does not have additional areas of the file system or registry which need unprotected access.

It is important to consider the discovery process as a repetitive process, which will ultimately lead to the final set of required changes. At the end of the discovery process, a detailed listing of changes that were required should be available for use in the next process: “Building the Security Template.”

The discovery process can be easy or very difficult, depending on the particular legacy application. The next sections will describe some discovery tools which start with the easy-to-find errors, and work up to the most difficult.

## Tools 1: Error Dialog Boxes, Application Logs, and Windows Event Viewer

The discovery process starts with the easy clues the application provides: Error messages or Application Logs. Although the legacy application may not properly run on NT based Operating Systems, it may provide a detailed log about what went wrong, or it may provide a descriptive error message in a dialog box. The Windows Event Log can also provide similar information.

An example error message: “Could not open C:\MyApp3.Log. Exiting Application.” This error implies that the legacy application needs access to write to the root of the C: drive. See Figure 1.

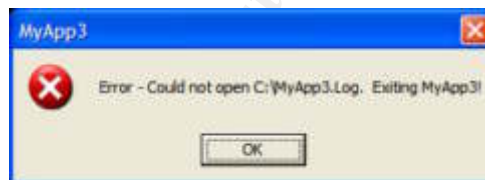


Figure 1 – Example Error Dialog Box.

If detailed error dialog boxes are not presented, the application may provide a detailed log of errors. Search the application directory for a log file with a current date/time stamp, to see if the errors are listed. See Figure 2.

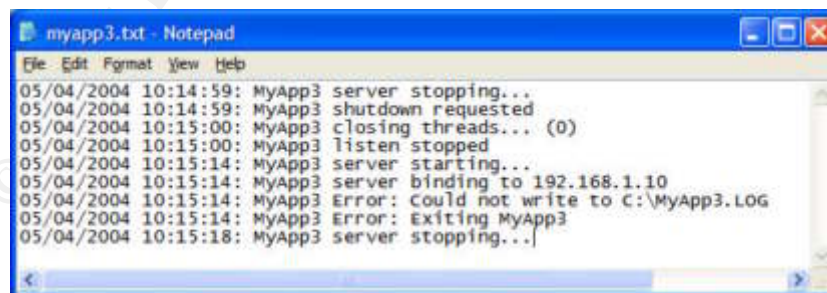


Figure 2 – Example Application Error Log

The Windows Event Log can also provide similar information. If system object auditing is enabled, Windows Event Log can be searched for error information. Proper configuration of auditing is beyond the scope of this paper. Auditing is a complex subject, and exact configurations depend on several variables, such as

Operating System type and the existing directory structures created by applications. Microsoft provides basic details for enabling auditing of system objects in the following Knowledge Base articles: 300549 and 310399. [7] [8]

Detailed filtering and reviews of the Windows Event Log, Security section can also provide information about failed access. The Event log can show this detail (see Figure 3):

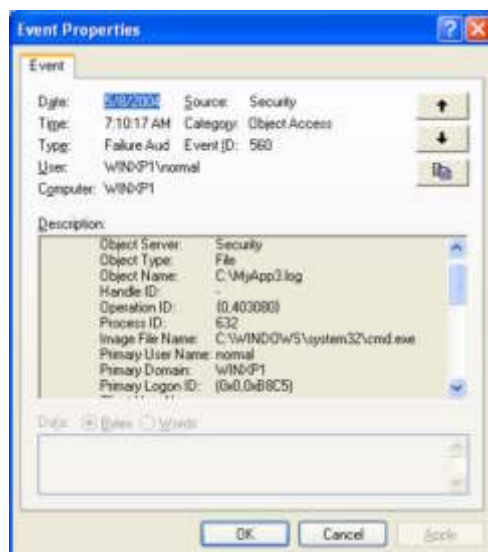


Figure 3 – Windows Event Log Failure, Object Access

Figure 3 illustrates the entry is a Failure Audit on Object Access, Event ID 560. In the details we see the Object Name of “C:\MyApp3.Log” which failed when the application CMD.EXE attempted to access it.

If built-in logs and dialog messages are not providing the details about the failures needed for discovery, additional, third party tools can be used. The next two sections provide details about some third party tools which may be helpful.

## Tools 2: Sysinternals.com – REGMON, FILEMON

Two excellent tools are available as Freeware from [www.sysinternals.com](http://www.sysinternals.com): REGMON.EXE and FILEMON.EXE. These tools are very easy to use, yet can provide powerful monitoring, and detailed results. [9] [10]

As the names suggest, REGMON is used to monitor access to the Windows Registry. FILEMON is used to monitor access to the Windows local File Systems.

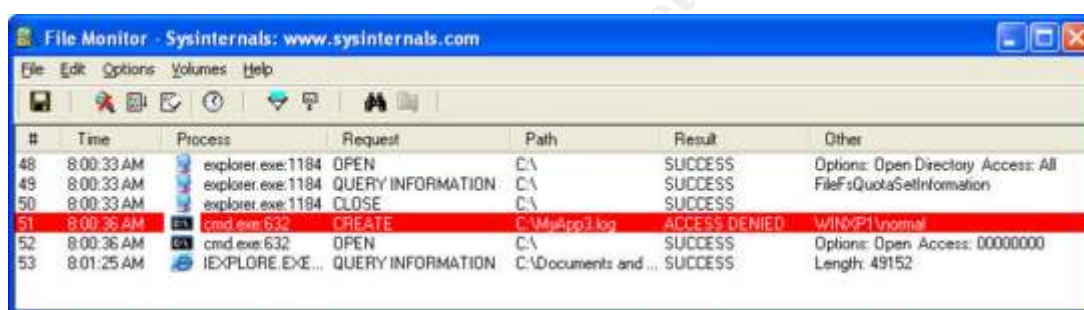
Bill Shaffer wrote a good GSEC paper [11] on the use of these tools in his particular environment in 2002. However, his description for remote use of FILEMON and REGMON requires purchase of the Enterprise Edition tools. See

References for details. The Enterprise Edition (not Freeware) tools support remote monitoring from an Administrative Workstation (with an Admin user), monitoring a Test Workstation (with a Normal User) and the test application.

An alternative using Windows 2000 or Windows XP is to use the “RunAs” function, with an Administrator account, while logged onto the desktop as the normal user which is being used to test the legacy application. This allows the use of the Freeware versions of the tools.

The key to using both tools is to discover the “Access Denied” errors related to the test legacy application. Once the file or registry key is identified, the security ACL for this item is documented with required changes, and then modified on the test workstation. (See steps 3 and 4 of the Basic Discovery Process.)

When using FILEMON, focus on the “Access Denied” errors. It is useful to set the “Highlight” feature for FILEMON, as this tool will capture all file access events – the list can be extensive. See Figure 4.



The screenshot shows the File Monitor application window with a table of file access events. The table has columns for #, Time, Process, Request, Path, Result, and Other. Row 51 is highlighted in red, showing an 'ACCESS DENIED' error for the process 'cmd.exe' attempting to 'CREATE' a file at 'C:\MyApp3.log'.

#	Time	Process	Request	Path	Result	Other
48	8:00:33 AM	explorer.exe:1184	OPEN	C:\	SUCCESS	Options: Open Directory Access: All
49	8:00:33 AM	explorer.exe:1184	QUERY INFORMATION	C:\	SUCCESS	FileFsQuotaSetInformation
50	8:00:33 AM	explorer.exe:1184	CLOSE	C:\	SUCCESS	
51	8:00:36 AM	cmd.exe:632	CREATE	C:\MyApp3.log	ACCESS DENIED	WINXP1\normal
52	8:00:36 AM	cmd.exe:632	OPEN	C:\	SUCCESS	Options: Open Access: 00000000
53	8:01:25 AM	IEXPLORE.EXE...	QUERY INFORMATION	C:\Documents and ...	SUCCESS	Length: 49152

Figure 4 – FILEMON Access Denied errors

Figure 4 illustrates that File Monitor has detected the process “cmd.exe” attempted to create a file “C:\MyApp3.Log” and the result was an access denied error for the normal user.

When using REGMON, focus on the “AccDenied” errors. It is useful to set the “Highlight” feature for REGMON, as this tool will capture all Registry access events – the list can be very long. See Figure 5.

#	Time	Process	Request	Path	Result	Other
1	3.54858872	cmd.exe:632	OpenKey	HKCU\Software\Policies\Microsoft\Control Panel\Desktop	NOTFOUND	
2	3.54871611	cmd.exe:632	QueryValue	HKCU\Control Panel\Desktop\MultiUILanguageId	NOTFOUND	
3	3.55216459	cmd.exe:632	OpenKey	HKCU\Software\Microsoft\Windows NT\CurrentVersion\App...	NOTFOUND	
4	3.55254481	cmd.exe:632	QueryValue	HKLM\Software\Policies\Microsoft\Windows\Saler\CodeIde...	NOTFOUND	
5	3.55328792	cmd.exe:632	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Ima...	NOTFOUND	
6	3.56117886	reg.exe:1064	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Ima...	NOTFOUND	
7	3.56863596	reg.exe:1064	QueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\...	NOTFOUND	
8	3.56872955	reg.exe:1064	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Dia...	NOTFOUND	
9	3.56900835	reg.exe:1064	OpenKey	HKLM\System\CurrentControlSet\Control\Error Message Inst...	NOTFOUND	
10	3.56953664	reg.exe:1064	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Co...	NOTFOUND	
11	3.57058956	reg.exe:1064	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\IME...	NOTFOUND	
12	3.57167461	reg.exe:1064	QueryValue	HKLM\System\CurrentControlSet\Control\Session Manager\...	NOTFOUND	
13	3.57187911	reg.exe:1064	OpenKey	HKCU\Software\Policies\Microsoft\Control Panel\Desktop	NOTFOUND	
14	3.57192883	reg.exe:1064	QueryValue	HKCU\Control Panel\Desktop\MultiUILanguageId	NOTFOUND	
15	3.57319017	reg.exe:1064	CreateKey	HKLM\software\myapps3	ACCESS DENIED	Access
16	3.57347680	reg.exe:1064	CreateKey	HKLM\software\myapp3	ACCESS DENIED	Access
17	3.57847883	cmd.exe:632	OpenKey	HKCU\Software\Policies\Microsoft\Control Panel\Desktop	NOTFOUND	
18	3.57853247	cmd.exe:632	QueryValue	HKCU\Control Panel\Desktop\MultiUILanguageId	NOTFOUND	
19	3.67694162	explorer.exe:1912	OpenKey	HKCU\Applications\cmd.exe	NOTFOUND	
20	3.67877482	explorer.exe:1912	OpenKey	HKCR\Applications\cmd.exe	NOTFOUND	

Figure 5 – REGMON Access Denied errors

Figure 4 shows that Registry Monitor has detected the process “reg.exe” attempted to create a registry key “HKLM\software\myapps3” and the result was an access denied error for the normal user.

Both tools allow filtering and saving the results as text files, which can assist in the documentation process.

### Tools 3: Repackaging Tools – SMS Installer

Larger companies and organizations may have advanced tools available for use. One example is SMS Installer. There are several similar tools available from third parties which perform repackaging functions, which may also be useful. Note: SMS Installer is a part of the Systems Management Server application from Microsoft. It is licensed software, separate from the Operating System.

SMS installer has a unique feature which may assist in investigating the source of errors or security problems, if other tools fail. The “Watch” function can help guide the user into areas we would not have expected the application to require access rights. [12]

It should be noted, this technique is not what SMS Installer (or any other packager application) was originally designed to do. The use of this feature is intended to find clues to what other security items might have been overlooked. Using this technique may provide information such as directories to review or other files that are accessed which were not discovered using other techniques.



Note: This technique requires running the application as an Administrator. The key principle here is: Try to discover the action the application takes when it is running correctly.

Launch SMS Installer in Watch mode. See Figures 5 and 6 for details.

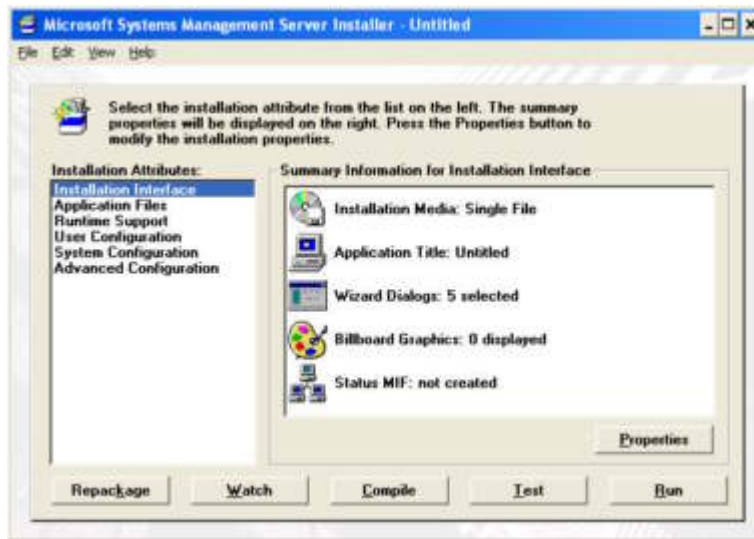


Figure 5 - SMS Installer Interface (Installation Expert Mode).

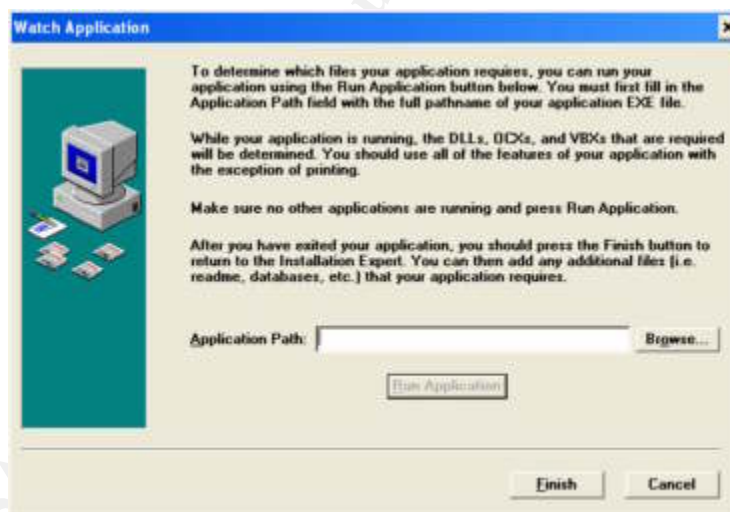


Figure 6 – SMS Installer Watch Application.

To use the “Watch” feature, enter the application path (or browse to it), and select “Run Application.” Once the application is running, use all of the application features available (Do not Print – this will add files to the script which are not part of the application.) Once all of the application features have been accessed, exit the application and then select “Finish” in SMS Install Watch Mode. Then select “View, Script Editor.” See Figure 7 for details on the Script Editor View.

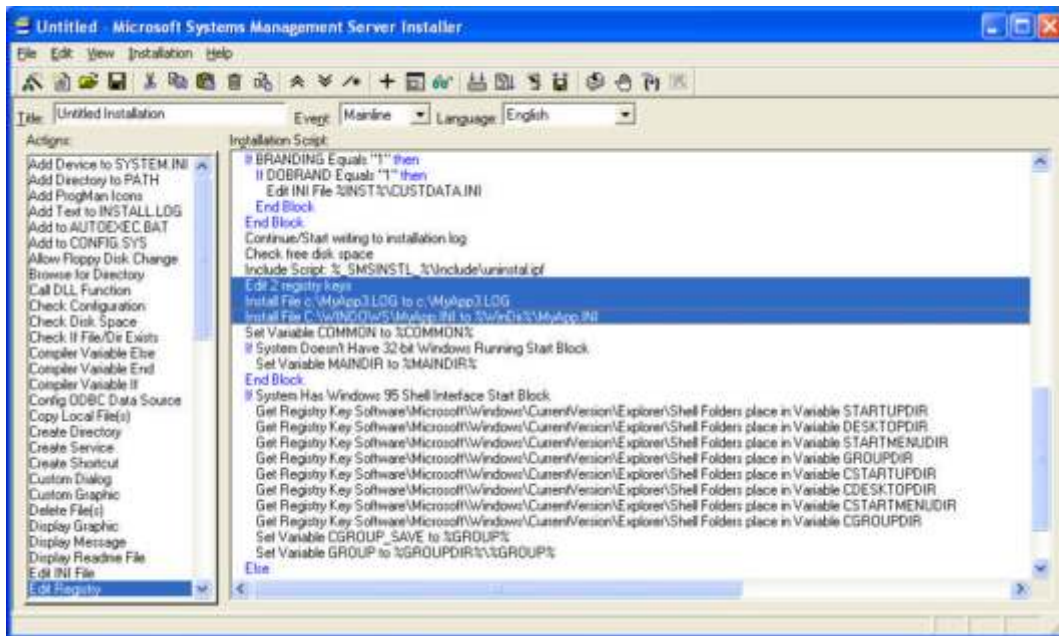


Figure 7 – SMS Installer Script Editor View

Once the Script Editor View is available, browse through the script and review the actions performed by the application. Direct attention to the “Edit Registry Keys” and “Install File” items. The script will contain many more options that should be ignored. Attempt to find another area which requires investigation, such as new directories, files or registry keys not previously discovered.

## Other Techniques

There are many other techniques and methods for discovering the actions an application takes when running. The key is to find actions which would not be permitted by NT security when run as a normal user, but which are allowed when running as an Administrator.

Other techniques which are less technical or scientific may be used:

- Run the application as Administrator, and then close it. Search the Hard Drive for all files modified within the past hour.
- Loosen security in a broader manner. If the application is installed in C:\MyApp, attempt to provide all Users “Full Control” access to the C:\MyApp folder. This technique can also apply to Registry Keys in HKey\_Local\_Machine.
- Export the entire registry before and after running the Application as an Administrator, and compare the changes.
- Use SMS Installer (or equivalent repackaging software) in “Repackage” mode, vs. “Watch Mode.”
- On a test computer, use trial and error to relax permissions on larger parts of the file system and registry, until we discover what large

change allows the application to run. For example, give all Users “Full Control” to the “Program Files” directory. Once it works, attempt to scale back.

- Use additional software discovery products. SYSDIFF.EXE, WINDIFF.EXE, and other Microsoft tools may provide additional details. These tools are cruder and less intuitive; however they might help in particularly difficult cases.

The discovery process can be simple, or it can be very tedious, depending upon the exact application that is being reviewed. The easiest approaches should be attempted first. They are: contact the company, review the application logs, and review any specific error messages that appear. These clues will significantly simplify the process. When all else fails, advanced discovery tools can provide the missing details. Once the application security changes are understood, the next step is to build a security template which contains these changes. Section 4 describes building the security template, for use with the SECEDIT tool.

## **SECTION 4: Building Security Templates**

Once the discovery process in Section 3 is complete, the next step is to use the information about the application's security changes to building a security template for SECEDIT. The security template will be built using the MMC add-in “Security Templates.” [13] [14]

A new template will be built for testing and distribution. Select “Action, New Template” from the menu, and provide a descriptive name and template description. See Figures 8 and 9 for an example.

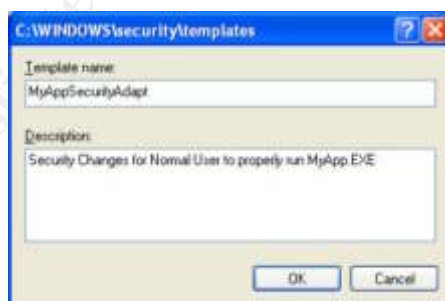


Figure 8 – Example New Template



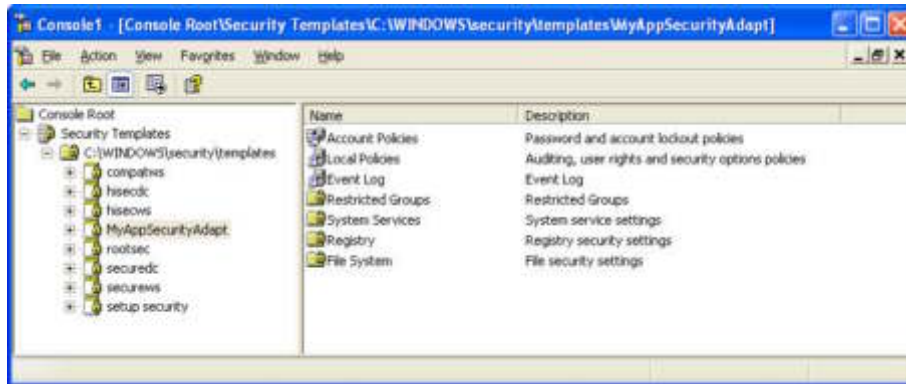


Figure 9 – MMC Security Templates Interface

The new template is now ready for modifications. There are 7 major areas that can be modified, but generally only the registry and file system permissions will require changes to allow legacy applications to execute for normal users. [15] [16]

## File System and Registry Changes

Select File System, and from the menu, select “Action, New File.” Browse to the specific file or folder to modify permissions. Once a file or folder is selected, the permissions are displayed for that file object. Make the appropriate changes to the file or folder. The next dialog box display relates to inheritable permissions. See Figures 10 and 11 for a normal example.



Figure 10 – Selecting File Permissions to modify in template

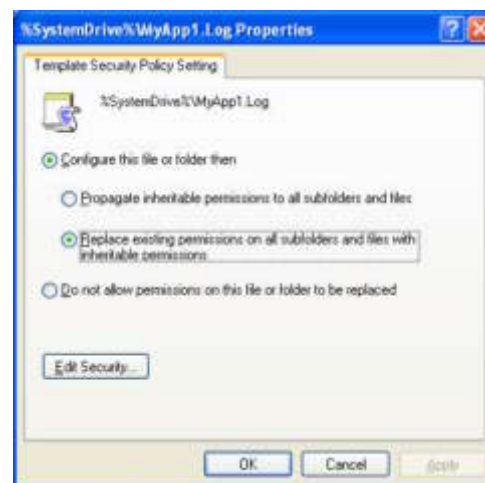


Figure 11 – Selecting Inheritable Options in template

Repeat these steps for each file or directory which requires modification. Once all the file system objects are completed, registry settings can be added.

Registry settings are very similar; however the selection browse and the available permissions are specific to the Registry. Once all of the registry objects are completed, the template can be saved.

Select the Template, and then from the menu, select "Actions, Save." A file will be created with the template name and an INF file extension. The default location for the file is C:\WINDOWS\Security\Templates. [17]

## **Services, Users, Groups**

Note: If this application was particularly complex, selection of additional items for services, users, groups or account policies may be required. Generally, these are not a problem with most legacy applications. In unusual cases, the discovery process would detect these issues. Using the Security Templates editor, these additional modifications can easily be added in a similar manner.

## **Copy the Template**

Copy this INF file to a network location or to removable storage for use on a new test workstation in the next section. Although it appears a valid template has been produced to address all of the security needs for this legacy application it still must be tested. The next section discusses testing and completion of the template.

## **SECTION 5: Applying and Testing Results**

The final step is the test process. Execution of the built template must be applied to a test workstation to confirm that the issues of this legacy application have been properly addressed.

## **Applying the SECEDIT Template**

Note: Applying the template requires Administrator rights.

The template should be copied to the C:\WINDOWS\SECURITY\TEMPLATES folder on the new test workstation. Section 2 discussed the command line for SECEDIT. In this particular example, following command line will be used:

```
C:\>secedit /configure /db C:\WINDOWS\SECURITY\TEMPLATES\myappadapt.sdb /cfg  
C:\WINDOWS\SECURITY\TEMPLATES\myappadapt.inf /overwrite
```

SECEDIT will prompt for confirmation to overwrite the security database. Answer yes to this question.

Note: When distributing this template to multiple computers, the “/quiet” command-line parameter can be added to suppress this prompt.

If no errors are reported by SECEDIT, test the legacy application to see if the modifications are complete.

## Testing Methodology

The testing of the application is necessary prior to sending the SECEDIT solution to production computers. Once the changes have been applied, the administrator account should log off, and a normal user (non-Administrator) should logon and test the application.

If any additional issues or errors are encountered at this point, repeat the discovery process (see Section 3) to determine what items need to be added to the template. This process of discover-test-build should be repeated until a successful test occurs and no new issues or errors occur.

Once the errors have all been addressed, it is good practice to apply the proposed final template to one or more computers that were not previously involved in the discovery or testing phases. These “clean machines” act as a final pilot of the proposed solution.

In some environments, this will be the time for customer testing and acceptance of the proposed changes. Remember, the testing is to occur with the users running the application without Administrator rights. That was the original goal!

## Applying the Results

Once testing is complete, the results are ready to distribute to all computers in the organization which use this application. If this was the last legacy application to be adapted, Administrator rights can be removed from the normal production users.

The distribution of the template(s) can begin using standard software distribution methods. The example command line would be executed (under an Administrative context) on each workstation.

## **CONCLUSION**

Removing Administrator rights from standard users in an organization is a vital security step. Enhanced security defense and reduced support costs will be realized once users operate with lower levels of security rights.

Frequently an organization will find that legacy applications stand in the way of this goal. SECEDIT is the tool provided by Microsoft to adapt individual security items on workstations to allow relaxed security on individual security objects, without granting general Administrator rights.

Although the discovery process for an undocumented application can be difficult, there are many tools which can help reveal why an application initially fails without Administrator rights. Using these discovery tools in a systematic process will allow a list of requirements for the application to be built. Once the discovery is complete a security template for the application can be built which can be applied with the SECEDIT tool.

Following proper testing and distribution of these security changes, the removal of Administrator rights from the users can be completed. Removing the Administrator rights will greatly enhance the organization's security, and should reduce support costs.

## REFERENCES:

- [1] Loomes, John. "Default Security Settings in Windows 2000 May Cause Legacy Applications to Fail." ServerWatch.com Tutorials. 6/14/2000.  
URL: <http://www.serverwatch.com/tutorials/article.php/1549091>  
(5/8/2004).
- [2] National Security Agency. "National Security Agency Security Recommendation Guides." NSA, USA. 1/16/2004.  
URL: <http://nsa1.www.conxion.com/win2k/download.htm>  
(5/8/2004).
- [3] Friedlander, David, with Jan Sundgren and Natalie Lambert. "Restricting User Desktop Rights Is Critical." CIO Magazine. 3/23/2004.  
URL: <http://www2.cio.com/analyst/report2346.html>  
(5/8/2004).
- [4] Christman, Jeff. "Windows 2000 Built-in Security Tools." 12/13/2001.  
URL: [http://www.giac.org/practical/jeff\\_christman\\_gcnet.doc](http://www.giac.org/practical/jeff_christman_gcnet.doc)  
(5/8/2004).
- [5] Minasi, Mark. "Assemble a Security Template - Test-drive a template in XP and Win2K." Windows & .NET Magazine. May 2003.  
URL: <http://www.winnetmag.com/Articles/ArticleID/38497/pg/1/1.html>  
(5/8/2004).
- [6] Minasi, Mark. "What Security Templates Can Do for You." Windows & .NET Magazine. Feb. 2003.  
URL: <http://www.winnetmag.com/Articles/ArticleID/37604/pg/1/1.html>  
(5/8/2004).
- [7] Microsoft Corporation. "HOW TO: Enable and Apply Security Auditing in Windows 2000." 11/20/2003. Microsoft Knowledge Base.  
URL: <http://support.microsoft.com/default.aspx?scid=kb;en-us;300549>  
(5/8/2004).
- [8] Microsoft Corporation. "HOW TO: Audit User Access of Files, Folders, and Printers in Windows XP." Microsoft Knowledge Base. 10/26/2002.  
URL: <http://support.microsoft.com/default.aspx?scid=kb;en-us;310399>  
(5/8/2004).
- [9] Cogswell, Bryce and Mark Russinovich. "Filemon for Windows – v. 6.06." 3/18/2003.  
URL: <http://www.sysinternals.com/ntw2k/source/filemon.shtml>  
(5/8/2004).

[10] Cogswell, Bryce and Mark Russinovich. "Regmon for Windows NT/9x – v. 6.06." 6/13/2003.

URL: <http://www.sysinternals.com/ntw2k/source/regmon.shtml>  
(5/8/2004).

[11] Shaffer, Bill. "Just give them Administrator rights that will allow them to run our software!" 5/18/2003.

URL: [http://www.giac.org/practical/Bill\\_Shaffer\\_GSEC.doc](http://www.giac.org/practical/Bill_Shaffer_GSEC.doc)  
(5/8/2004).

[12] Trent, Rod. "SMS Installer Walkthrough: Watching an Application." 5/31/2002.

URL: <http://www.myitforum.com/articles/12/view.asp?id=2934>  
(5/8/2004).

[13] Aitken, Robert. "Taking the Confusion Out of Security Templates." 2/27/2003.

URL: <http://www.sans.org/rr/papers/67/989.pdf>  
(5/8/2004).

[14] Kelly, Bob. "Using Secedit To Apply Security Templates." AppDeploy.com Scripting Tips. 6/14/02.

URL: <http://appdeploy.com/tips/detail.asp?id=22>  
(5/8/2004).

[15] Microsoft Corporation. "HOW TO: Apply Predefined Security Templates in Windows 2000." Microsoft Knowledge Base. 11/4/2003.

URL: <http://support.microsoft.com/?kbid=309689>  
(5/8/2004).

[16] Microsoft Corporation. "Step-by-Step Guide to Using the Security Configuration Tool Set." Undated.

URL: <http://www.microsoft.com/technet/prodtechnol/windows2000serv/howto/seconfig.msp>  
(5/8/2004).

[17] Kelly, Bob. "How To Create A Custom Security Template." AppDeploy.com Scripting Tips. 6/14/02.

URL: <http://appdeploy.com/tips/detail.asp?id=22>  
(5/8/2004).