# Global Information Assurance Certification Paper

# GIAC GCWN Practical Assignment
# Version 5.0

Option 1: Solving a Windows Problem

"Using Group Policy for efficient application patch management of a highly managed desktop"

SANS Virginia Beach Conference
August 29 – Sept 3, 2004

By Bilyana Doslo

# Table of Contents



# List of Figures

## Abstract

The contents of the following document have been compiled from pieces of various sources. A list of most relevant URLs and articles are included at the end of the assignment.
This research document is focusing on patch management of highly managed desktops for a medium-sized organization and is presenting scripted solution for application patch management. This fictional medium-sized organization implemented Microsoft Software Update Services (SUS). The focus of this paper is to describe how to deploy security updates that can not be handled by the Microsoft SUS: application patch and service packs distribution. The solution utilizes Group Policy, Microsoft Installer (MSI) package development and WinBatch scripting. All code required for implementation is provided. The underlying Active Directory and Group Policy design that supports this Security Update mechanism are outlined within the paper emphasizing design of efficient application patch management.

## Document Conventions

When you read this practical assignment, you will see that certain words are represented in different fonts and typefaces. The types of words that are represented this way include the following:

| | |
|---|---|
| `command` | Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell. |
| `Source Code` | Program source code in perl, VBscript, or other programming languages. |
| `filename` | Filenames, paths, and directory names are represented in this style. |
| `computer output` | The results of a command and other computer output are in this style |
| URL | Web URL's are shown in this style. |
| *Quotation* | A citation or quotation from a book or web site is in this style. |

## Introduction / Executive Summary

An automated process for maintaining the desktop and server assets at current patch levels is a fundamental security requirement for any organization doing business today. The number of patches and hot fixes being released by software vendors has increased considerably. Attacks on computers running Microsoft Windows operating systems and software products are becoming more widespread and sophisticated. Application patch management has become as much important as operating system patch management. In response to newly identified operating system vulnerabilities, every major software vendor committed to security will release security patch as soon as vulnerability is discovered. Un-patched applications can adversely affect the applications, computers and other systems on the network.
This paper outlines simple strategies, and briefly describes several tools, for deploying and managing patches. The first part of this paper introduces the scenario of the medium-sized organization where a security problem is identified. In this section an imaginary example highlighting the issues in application patch deployment is presented. From this scenario, basic concepts and requirements are derived. A number of existing Microsoft technologies are then characterized against the hypothetical Windows scenario and product shortcomings are identified. The second part focuses on scripted solutions and its implementation. A final section completes the implementation guide and validation.

# Section One: Identification and Description of a Windows Security Problem

## *Organization background*

The scenario presented in this material is based on a medium-size an imaginary company called Sipad Komerc. Sipad Komerc is an Eastern European corporation with distribution centers located in Sarajevo (Bosnia), Belgrade (Serbia), Skopje (Macedonia), Ljubljana (Slovenia) and Zagreb (Croatia). Each center has roughly 500 users. The headquarters is located in Sarajevo (Bosnia). All the distribution centers are linked to the corporate head office with T1 connections. The company's network consists of a single Active Directory domain and facilitates centralized management with a delegated Organizational Unit (OU) structure for core Active Directory services and roles based on a geographical site administration model as shown in Figure 1.(child OU structure at each center as presented in Slovenia OU). Each distribution center contains at least one domain controller that is also configured as Global Catalog.
The overall network is totaling 95 Windows 2000\2003 servers and approximately 2400 desktops. Sipad Komerc uses both Windows 2000 Professional and Windows XP client computers which have been updated to the latest service pack. Desktop operating system patching is performed leveraging Microsoft's Software Update Services (SUS) with a primary and secondary update server.

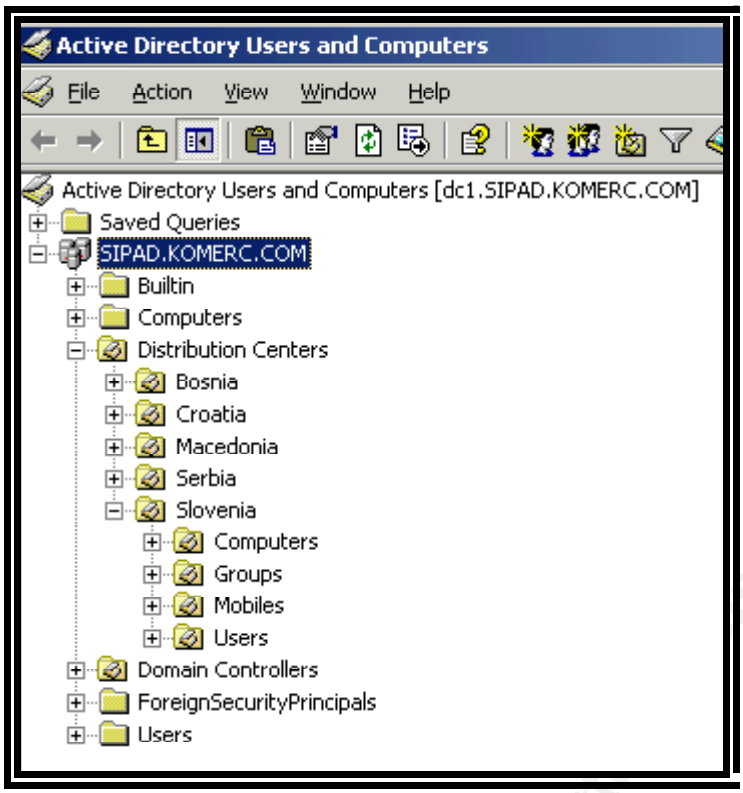


**Figure 1: SIPAD Komerc Active Directory structure**

## *Desktop and Application Support*

The organization's computing environment is host to a number of different applications. The headquarter server and the distribution center servers manage the main application database for the organization utilizing Distributed File System (DFS). All applications are replicated throughout organization. Sipad Komerc standardized on the Microsoft suite of applications and implements Microsoft Installer (MSI) packages for the deployment of software as it provides the foundation for true user portability by enabling application access from any desktop. Most standard applications are deployed using Group Policy and are assigned to computers in Distribution Centers Group Policy as shown in Figure 2. To ensure proper, timely installation all managed desktops are scheduled for a periodic reboot.

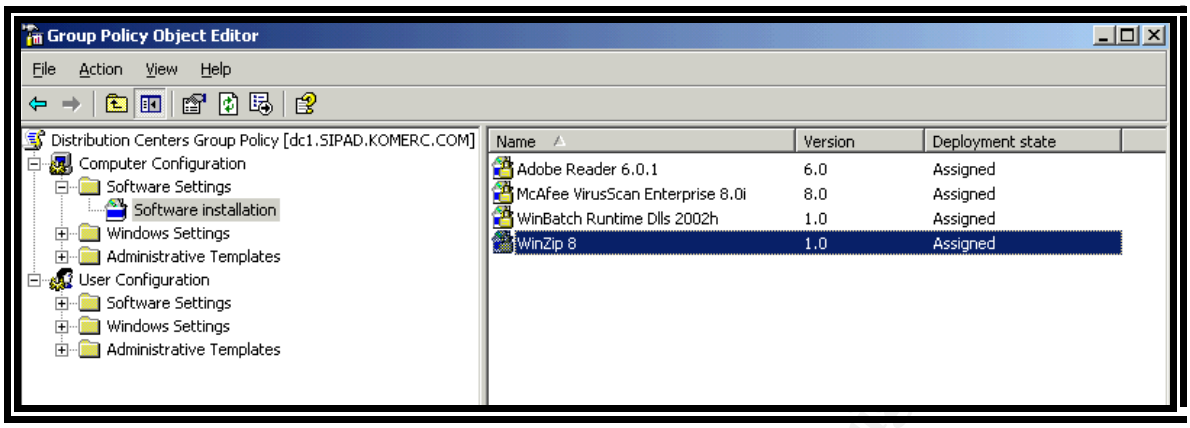


**Figure 2: Computer Assigned application – Distribution Centers Group Policy**

This organization is using an image-based operating system installation for new computer deployments and reinstallations. Due to the nature of their business every desktop comes with a pre-installed KondaliSoft application and Microsoft Office 2003 as a part of desktop image. To enable application repair from local cache, Microsoft Office 2003 is installed from compressed CD image rather then an administrative installation point. All other business related applications are published to users from each distribution center's Group Policy. Access to published applications is controlled by Application group membership.
Each distribution center requires specific application regional settings and is given the ability to customize its local environment to meet the specific needs of its business. Specifically, each distribution center is responsible for the configuration of branch-specific systems and information, such as database applications, stand-alone tools and user profiles.
Logon\startup script developed using the WinBatch scripting tool is configured to run from Distribution Centers Group Policy. Users are required to regularly log off from their desktops, otherwise a forced log off occurs after one hour of inactivity. The custom WinBatch script also scans networks on a regular basis to assess the overall health, builds a software inventory list and is monitoring the network for vulnerabilities to determine if patches have been effectively applied.

## *Objective*

Due to increased number of critical security software patches released, Sipad Komerc needs to implement an application patch management process to protect the security of its applications and workstations.
Throughout the process of testing and review, the following application security patches have been identified as most critical and can not be installed via SUS:

1. Microsoft Office 2003 Security updates:
   - KB838905, fixes security vulnerability in the graphics interpreter code
2. Microsoft Visio 2002 Security update:
   - KB831932, fixes a security flaw in Visio 2002
3. KondaliSoft critical security updates:
   - Critical Security Rollup pack 11

The organization needs a centralized solution where released security patches

not handled by Microsoft SUS can easily be deployed to both test and production environments using available technologies.
Since the most cost effective solution is mandated, a set of business procedures are required to ensure knowledge transfer and to ensure that future enhancements to the patch management architecture can be implemented in a consistent manner.

## *Patch management – Current technologies*

The IT department had been utilizing Microsoft SUS for several months. Once approved, the operating system patches are installed automatically from SUS server. The process is handled via Group Policy and the Active Directory. Due to SUS product limitation, application patch management is performed manually. The manual process is decentralized and lacks efficiency and accountability. Prior to SUS v 1.0 implementation, Sipad Komerc surveyed and assessed various existing technologies which essentially fall into 2 categories: Microsoft solutions or third party solutions.
This organization standardized on Microsoft products therefore they compared features and products supported by Microsoft first. Although most Microsoft products today have integrated "Product Update" option, the regular corporate user can't use it due to insufficient privileges. Installing the patches requires end users to have local administrative privileges on their computers. *Although this requirement was normal on the Windows 9x OS, most organizations using Windows NT, Windows 2000, or Windows XP desktops do not grant local administrative privileges to end users, primarily because it's cheaper to support workstations that have standardized and locked-down configurations. Furthermore, even when end users have administrative rights, there is no way to ensure that all users will update their workstations correctly and only when instructed. For this reason, many organizations completely block Windows Update access to end users.*[*1] Directions on Microsoft, Peter Pawlak
Needles to say, Sipad Komerc's primary desktop configuration is identified as highly managed, categorized by users job requirements and locations. Most users operate in User Security context without special privileges and sufficient rights to perform patch installation or any other system changes.

## *Product Evaluation*

Microsoft provides other automated patch management and deployment solutions, specifically HFNetChk, Microsoft SMS and Microsoft SUS.
HFNetChk is free of charge inventory type tool. In order to automate the actual deployment of a patch, customers must purchase the Shavlik tool HFNetChkPRO.
Microsoft SMS with optional Feature Pack will distribute patches automatically, but it is expensive and requires an SMS client on every machine.
Small and medium-sized organizations looking for an inexpensive reliable solution should consider Microsoft SUS. Although it is not able to install service packs or handle Office Update patches, driver updates, or other noncritical

patches, such as normal updates to Windows Messenger or non-Microsoft product updates it is nevertheless valuable, especially when combined with a tool like MBSA (Microsoft Baseline Security Analyzer), which scans Microsoft systems for patch and vulnerability status and links to SUS to provide critical updates. Implementation of SUS is straightforward and well documented by Microsoft. Another excellent source of articles and supporting documents for SUS implementation and patch verification can be found on the SANS Website. The comparison chart below summarizes Microsoft Patch Management technologies.

*Comparison Chart of three Microsoft Security Patch Management technologies*[*2 University of Manitoba]

| Feature | Windows Update | SUS 2.0 | SMS with SUS FP |
|---|---|---|---|
| Core Scenario | Consumer desktops and unmanaged end-users systems connected to Internet, download and install updates as they become available. Designed for individual users who administer their own machines. | Corporate desktop systems install updates that are authorized by the corporate IT administrator. End user experience similar to Windows Update. Designed for small/medium sized enterprises with simple scheduling and targeting. | Corporate desktop systemsinstall updates that are authorized by the corporate IT administrator. End-user experience is controlled by the corporate administrator. Designed for large enterprises that need sophisticated scheduling, targeting, and bandwidth aware patch management. |
| Platform Support | Windows 2000 and later. | Windows 2000 and later. | Windows 98 and later. |
| Updates for Microsoft products | Updates only Windows operating system and its components. | Updates only Windows operating system and its components. | Updates to Windows operating system, its components, Exchange Server, SQL Server, and Microsoft Office. |

| | | | |
|---|---|---|---|
| Updates for non-Microsoft products. | Not supported. | Not supported. | Basic software distribution feature of SMS can be used to deploy updates for non-Microsoft products. |
| Control | End-user decides what updates to install and when. | Corporate administrator controls what updates are available to the end user who decides if and when to install them. | Corporate administrator controls all aspects of update installation. The end-user experience can be fully customized. |
| Enterprise features | None | Basic | Rich set of enterprise features for software and hardware inventory, bandwidth-aware software distribution, sophisticated scheduling, and targeting. |
| Reporting | Local computer reporting from each computer only. | Basic reporting at the enterprise level only is available. | Rich reporting driven by individual machine data in the SQL Server database. |
| Deployment | No deployment required. | Simple planning and ongoing process. | Requires detailed planning for new customers (simple for existing SMS customers). Ongoing administration process will depend upon richness and complexity of enterprise expectations. |

| | | | |
|---|---|---|---|
| Cost | Free of charge Microsoft web page. | Free of charge process which must run and be updated on a machine local to the enterprise. | Each workstation using this solution must be licenced and then run a procedure from a machine local to the enterprise.. |

Microsoft's Windows Update Services (WUS), now in open-evaluation stages with the final release expected in the first half of 2005, would be good choice for organization above to keep computers up-to-date and may be enough to provide critical systems with the latest patches for Microsoft Products. Microsoft says the final release of Windows Update Services "*will support updating Windows operating systems as well as all Microsoft corporate software over time. When initially released, Windows Update Services will support updating Windows XP Professional, Windows 2000, Windows Server 2003, Office XP, Office 2003, SQL Server™ 2000, MSDE 2000, and Exchange 2003. Support for additional Microsoft products will be added over time—without the need for upgrading or redeploying a Windows Update Services implementation*."[*3] Microsoft WUS FAQ For security reasons updates will be limited to Microsoft products only. From the statements above it's not clear what other products will be added and in what timeframe.

The bottom line is that small and medium sized organizations who can not justify costs of investing in SMS or other third party products have to come up with their own patch management solutions for all other products currently not supported by Software Update Services or Windows Update Services.

In the given scenario, an automated process of installing Microsoft and non-Microsoft patches across its infrastructure is required. Manually patching 2400 machines physically distributed is neither practical nor reliable, it requires excessive resources and can be very time consuming and prone to errors. Can this organization justify costs of investing in an alternate tool? What is the cost of downtime and the security risk involved if systems are not patched? Sensitive information leaks, lost credibility, stolen intellectual property? What is the cost of possible forensic investigation? Questions like these and others will drive decisions on the investments made by corporations. In many cases it will be based on the frequency of released patches, severity level and available budget. Assuming this organization aside from Microsoft released patches for those 2 products have need of 1or 2 updates a month for KondaliSoft application, with Windows Update Services on its way, designed for Microsoft product updates, this organization most likely will not be able to justify costs of investing in patch management tool for non-Microsoft applications only.

## *Patch management using Group Policy*

Sipad Komerc is using Group Policy as the primary mechanism for application deployment utilizing Windows Installer technology, therefore every installation has to be either in MSI or ZAP format. Microsoft's Group Policy design prevents

ZAP based installation to be assigned to a machine. It can only be published and it runs in user context. For this reason a ZAP file based patch installation is eliminated as a viable solution in this case. As per Q320539, Microsoft hotfixes are not designed to run as repackaged Windows Installer files. The only supported installations of Microsoft hotfixes are those associated with the package that they are available from as downloads.
Today, most Microsoft patches are self-extracting executable files (EXE) where the Microsoft patch file (MSP) can be extracted and used for updating (patching) an administrative installation point. To patch an administrative installation point, first extract the MSP file from the full-file version of the software update using this command:

```
[path\name of EXE file] /c /t:[location for extracted MSP file]
```

Then run Windows Installer with options to apply the patch to the administrative installation point:

```
[start] msiexec /p [path\name of update MSP file] /a [path\name of
MSI file] SHORTFILENAMES=TRUE /qb /l* [path\name of log file]
```

The update instructs Windows Installer to add, update, or remove files in the administrative image. Microsoft Visio 2002 can be patched in this manner. The "Redeploy" option in Group Policy forces application re-installation along with patch deployment and is initialized as shown in Figure 3


**Figure 3: Using Redeploy function for updating Visio 2002 clients**

The strategy above is applicable to all Microsoft products' assuming an MSP file is designed to update certain product codes and assuming the application is originally installed by using IntelliMirror software installation and is managed by Group Policy, however a rollback is not supported. There is no way to uninstall the patch separately without removing the application completely and then reinstalling it

## Section Two: Solve a Windows Security Problem

Unfortunately, the above described strategy can not be applied for Microsoft Office 2003 security updates as this application, in Sipad Komerc case, is not managed by Group Policy and is not installed by IntelliMirror software installation. KondaliSoft security updates are released as a single client binary updates.
In the next section custom solution for patch distribution is presented that is utilizing currently adopted technologies, which is: Group Policy, Windows Installer technology, custom scripting codes using batch files and Wilson Windoware WinBatch scripting tool.

### *Solution description – Global overview*

This Security Update mechanism is designed with 2 strategic components in mind: a static local component and dynamic server component.
The static executable is delivered to users desktops via Group Policy as a Windows Installer package assigned to all workstations. This component is installed in the `Program Files` folder and is used to pull and launch dynamic part of the script located on the network.
The dynamic component of the script is updated on regular basis with codes for silent patch installation. Once compiled, the dynamic executable is placed in the FRS replicated folder on the server along with required patches and is available for installation almost instantly. The installation can be triggered in many ways, depending on the organizations requirement. It can be configured to run as a startup script at users logon, or as scheduled task, or as program shortcut or as a part of the logon script. For Sipad Komerc, the static script is designed to run when Windows starts, after the user logs on. Startup script is able to run in system context without impersonation, however is not used due to the fact that by design all logon, logoff and startup scripts collectively have a time limitation. By default, the timeout is 600 seconds and all processing must finish within 10 minutes. If this time is exceeded the execution can be terminated in mid-stream. This setting is configurable within Group Policy and if not adjusted properly, (assuming that this dynamic script can be a really long list of security patch installations), there is no guarantee that everything can be completed in a given timeframe. The dynamic script has built in logic to check if security update has run, the log file is created accordingly and the installation closes gracefully without user interaction. The downloaded dynamic script runs in an impersonated administrator mode and is deleted at the end of the installation for security reasons. A reboot flag is set if a workstation reset is required, prompting the user to reboot the workstation. The script is coded with a minimum set of functions and limited user interface for clear understanding, but it can be extended and customizable in many ways using other WinBatch functions. The purpose of this document is not to detail the use of every tool and technology used for resolution, but rather to present a workable solution leveraging different technologies that can be used as a baseline for future

development and ongoing support of application patch management.

# Section Three: Scripting / Automation of the Solution

Scripts allow system administrators to automate many repeatable administrative tasks. While there is no acknowledged "Best Scripting tool", there are some tools that are strictly designed for Windows environment. WinBatch's Windows Interface Language (WIL) is a library of code functions (called extenders) pre-defined to handle tasks that must be otherwise coded in other tools. In next few pages sample codes for patch automation written in WinBatch 2002h is presented. Also there is a sample of Windows Installer MSI package developed using WinInstall LE. It is assumed that the reader of this document possesses basic understanding of the Windows Registry, Windows Installer technology and some scripting. The following solution has been lab tested, but may not work in all environments, so it must be thoroughly tested. Also keep in mind that for the WinBatch script to run, the client needs to have access to the WinBatch system files. These files can be located in the same folder as the executable or on the system path. Sipad Komerc is using an assigned Windows installer MSI package to deliver all required WinBatch system files to all systems as presented earlier in Figure 1.2. The package is running from Distribution Centers Group Policy and it propagates all the way down to all machines located in child OUs affecting all objects inheriting the policy.

## *Static Component*

The static component `Static.exe` contains several lines of simple code. The script begins with user interface setup, network extender loading, main subroutine definition and variable setup. It copies and launches `Dynamic.exe` to temp location using the `RunWait` command. After the processing is completed, `Dynamic.exe` is deleted and the script closes.

```
WinHide("")»; hide this window
BoxOpen("Security update", "Please Wait...")
AddExtender("WWWNT34i.DLL")
;---------------------------------------------------------------------------
:Main
»  exclusive(@ON)
»  gosub Variables
»  gosub Download
»  exclusive(@OFF)
»  goto End
;---------------------------------------------------------------------------
:Variables
»  temp = Environment("TEMP")
return
;---------------------------------------------------------------------------
:Download      ;Download dynamic executable and launch patch installation
»  errormode(@NOTIFY)
»  BoxText("Running Security update...")» »
»  patchSourceFile = "\\SIPAD.KOMERC.com\netlogon\Updates\Dynamic.EXE"
»  patchDestinationFile = "%temp%\dynamic.exe"
»  Filecopy(patchSourceFile, patchDestinationFile,@FALSE)
»  RunWait("%temp%\dynamic.exe","")
»  filedelete("%temp%\dynamic.exe")
return»
;---------------------------------------------------------------------------
:END    ;  End of Script
BoxShut()
```

**Figure 4: Static component – WinBatch source code**

## Distributing Static.exe to users desktop

This section provides a brief description of how IntelliMirror and Group Policy achieve static component compiled as `Static.exe` distribution to the users desktop. IntelliMirror was introduced with Windows 2000 and is set of powerful technologies designed to increase availability and reduce total cost of ownership (TCO). The ability of the automated software distribution to make automatic repairs to any damaged application and to uninstall application without harming the shared files that other applications use is built into the Windows 2000\XP operating systems by leveraging Windows Installer technology. The Windows installer service manages the installation processes using the dynamic link library `msi.dll` to read the MSI package file. There are 2 types of MSI: native, provided by software vendor or custom, an in-house developed (or repackaged) MSI file. Various installer-capable packaging tools exist for re-authoring an application setup and an MSI development. Veritas WinInstall LE is a free tool; shipped with Windows 2000 Advanced Server CD-ROM has been improved and is now available as WinInstall LE 2003. The MSI package component containing 2 settings is created using WinInstall LE 2003 as listed below.

1. Add `Static.exe` in [ProgramFilesFolder]\Security Update folder as shown in Figure 5

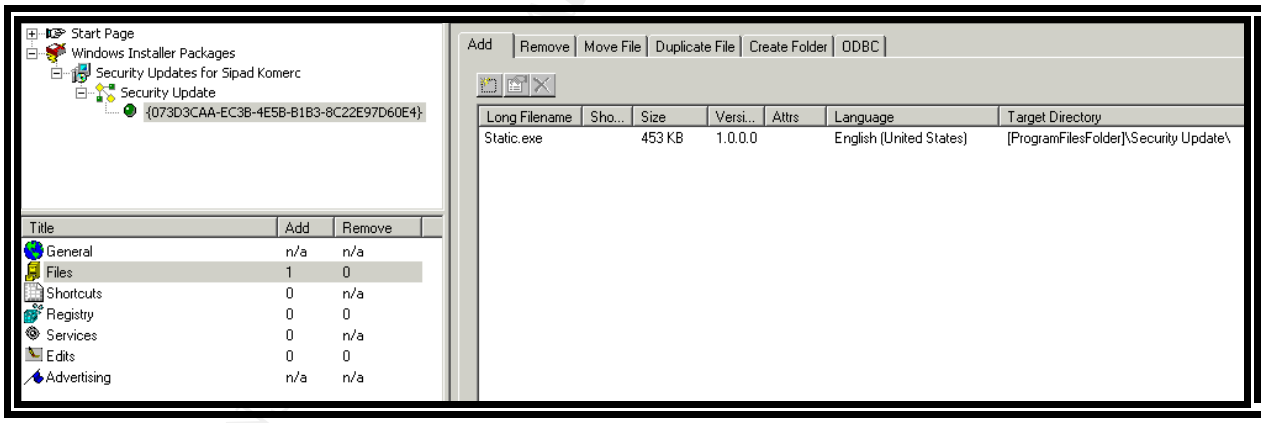


**Figure 5: MSI development – Add Static.exe**

Program invocation can be activated in many ways. By creating shortcut to `Static.exe` users can be enabled to launch the executable from Program Menu. Although, nothing can prevent users from browsing `Program Files` and launching `Static.exe`, the shortcut in this scenario is left un-configured and remains hidden from users Program Menu. This program is designed to run without user intervention and only after users log on. To address the situation where users do not log off configure `Static.exe` to run as a scheduled task. For the Sipad Komerc scenario, in a highly managed desktop environment, the user is forced to log off after 1 hour of inactivity. Periodic reboots also help to

ensure that certain components delivered via Group Policy remain current and always in place.

2. Create Registry key value “Security Update” under HKLM\Software\Microsoft\Windows\CurrentVersion\Run with string data value set to:
   [ProgramFilesFolder]\Security Update\Static.exe
   as depicted in Figure 6

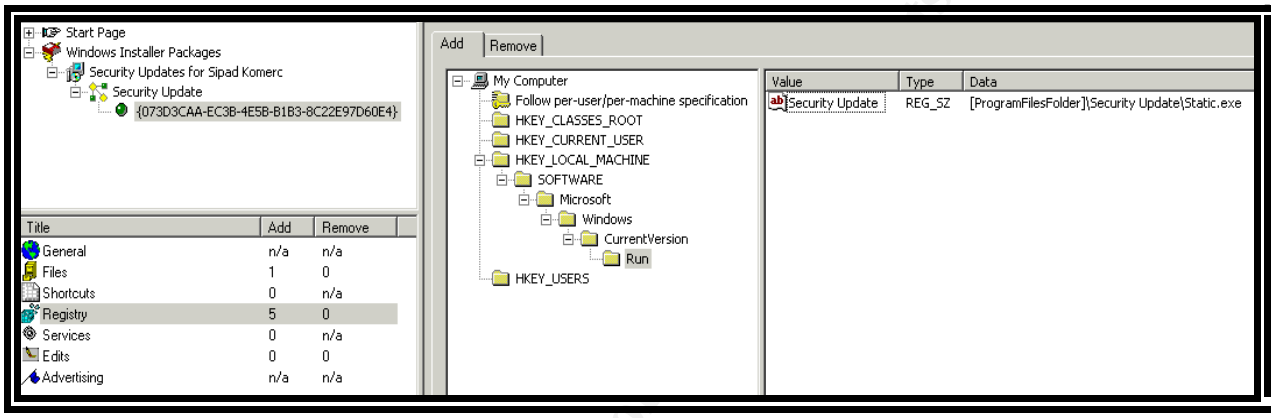


**Figure 6: MSI development – Add Registry value**

The Run key in the package above can be set to run `Dynamic.exe` directly from network location `\\sipad\netlogon\updates` instead of loading `Static.exe`. However, there are some drawbacks to consider with this strategy. One is to avoid situation where `Dynamic.exe` can become locked by another system and may become inaccessible to others. If this file is in use it may not easily be replaced with newly compiled version of executable. Furthermore this is to ensure that this Security Update mechanism is always in place and when corrupted is easily restored to original state. By setting key file path to `Static.exe` this executable can repair itself if missing or corrupted. Shortcuts to network located programs can not self-heal and if missing there is nothing that would trigger re-installation.
This concludes `Security Update.msi` package development. An MSI package is a collection of components where a component is a file, a Registry key, a shortcut or any other such object. The actual installation process is performed locally on the user’s machine by Windows Installer service routines. Windows Installer packages are designed to seamlessly integrate into the Active Directory. By assigning this package to the Distribution Centers computer configuration Group Policy this Security Update mechanism becomes mandatory for all computer population. To assign this package, open Group Policy snap-in window, select Computer Configuration, Software Settings, Software installation. Right click the Software installation node and select New, Package. Use File Open dialog box, to select the path to the MSI file. (in this case DFS path). The system processes this application upon machine startup. Upon machine reboot, progress bar appears stating “Installing Managed

Application Security Updates…".
In combination with Group Policy and the Windows Installer Service the un-installation can be automated in the same manner as the installation.

## *Dynamic Component*

The server component is more complex and requires additional functions to run, but it is repeatable and its subroutine can be re-used with the same logic for newly added updates. Updates are needed as frequently as patches are released and have to be deployed into production. In summary, the script's primary function is the `RunWithLogon` where impersonation is required. Actual patch installation runs in silent mode using credentials of a user with administrator privileges. The script starts in much the same way as previous, with user interface (window) setup, main section with subroutines, network extender loading and variables setup as shown in Figure 7

```
; ------------------------------------------------------------------------
WinHide("")»; hide this window
If Param0 != "0"
   If StrIndexWild(StrLower(Param1), "debug", 1) > 0 Then Debug(@On)
Endif
errormode(@OFF)
:Main
     gosub Extenders
     gosub InitVariables
»    gosub q838905
»    gosub packll
»    gosub ENDofscript» »
exit
; ------------------------------------------------------------------------
     :Extenders
» »    »»  AddExtender("WWWNT34i.DLL")
» »  return»
; ------------------------------------------------------------------------
     :InitVariables
» »    reboot » »  »  = 0
       runas_user » »  = "agent007"
» »    runas_pswd » »  = "LostDomovina_92"
» »    runas_domain »  = "SIPAD"
·· ------------------------------------------------------------------------
```

**Figure 7: Dynamic component – WinBatch source code part 1**

The next few sections are the actual patch installation code. Each section starts with a search for certain values in the file system or Registry to identify whether the patch has been installed earlier. In this case, as per Microsoft KB838905, Office 2003 security vulnerability patch installation is determined by `gdiplus.dll` file version.
The `FileVerInfo` function is used to query for the file version value. The installation is coded to start only if the patched version of this file is not found. This is where impersonation and `RunWithLogon` function is used. See Figure 8 Agent007 is domain user ID with local administrative privileges. The installation

of Office 2003 patch is wrapped into Install.cmd file with these commands:

```
msiexec /p "\\sipad\netlogon\Updates\kb838905.msp" /Lime %temp%\kb838905.txt /qb-!
```

In summary, this command ensures silent patch installation generating log file. Most of the major patch installation settings can be controlled dynamically from a central configuration file Install.cmd. In fact batch file can handle every patch installation via Install.cmd. Batch files are simple to edit, do not require compiling, but offer limited functionality.

```
:q838905    ;   Office 2003 Security patch Q838905
GDI = FileVerInfo("c:\Program Files\Microsoft Office\OFFICE11\gdiplus.dll","","#ProductVersion")
if GDI == "6,0,3264,0"
   return
else
patchCMD = "\\SIPAD.KOMERC.com\netlogon\Updates\install.cmd"
Boxopen("","")
BoxText("Security Patch installation running...")
RunWithLogon(patchCMD, "","", @HIDDEN, @WAIT, runas_user, runas_domain, runas_pswd, 0)
boxshut()
return
-------------------------------------------------------------------------------------------------
:packll
packll = RegQueryValue(@REGMACHINE,"SOFTWARE\KondaliSoft\Updates\Packll[Description]")
if packll == "Security Rollup pack ll" then
   return
Else
Boxopen("","")
BoxText("Security Patch installation running...")
packll = "\\SIPAD.KOMERC.COM\netlogon\updates\packll\setup.exe"
RunWithLogon(packll, "/s","", @HIDDEN, @WAIT, runas_user, runas_domain, runas_pswd, 0)
boxshut()
return
-------------------------------------------------------------------------------------------------
:EndOfScript
if reboot == 1 then
BoxOpen("","Preparing to restart system.........")
TimeDelay(15)
IntControl (67, 0, 1, 0, 0)
endif
BoxShut()
return»
```

**Figure 8: Dynamic component – WinBatch source code part 2**

KondaliSoft application stores its patch installation values in the Registry, thus the RegQueryValue function queries for the Registry key value. If not found, the Security Rollup pack 11 installation runs silently using RunWithLogon function. Both RunWithLogon functions are coded to run in hidden mode and set to wait for the installation to complete. The last section is for reboot control. A reboot flag can be set at the end of each patch installation section. This code is compiled as Dynamic.exe. The location of Dynamic.exe can be any location where domain users have appropriate permissions. As presented in Figure 4, Static component – Winbatch source code, the Dynamic.exe was placed in the Netlogon subfolder. The Netlogon share is replicated across all domain controllers using the File Replication Service (FRS) and is accessible to all domain users. FRS is a near real-time multimaster replication system. All

`required patch updates, batch file` and `Dynamic.exe` should be placed in the `\netlogon\updates` folder.
For security reasons consider establishing a secure channel replication between domain controllers by implementing IPSec and using static port for all RPC TCP/IP traffic. (KB319553)

## Section Four: Implementation Guide

These instructions assume familiarity with the prescribed software administration and development tools. Section Three of this document is integrated part of the implementation guide. It describes solution design and establishes baseline for future development and maintenance. A method for identifying vulnerabilities is a separate process and is not part of this guide. Assuming vulnerability is discovered, and a patch is released and is considered critical then the following sequence of steps is recommended:

- All software updates should be downloaded and reviewed in an isolated, quarantined nonproduction environment.
- Download software updates only from trusted, virus free sources and confirm authenticity by verifying the digital signature
- Read the security bulletin (or any other vendor documentation) carefully to ensure that no known incompatibilities exist. At the bottom of the bulletin's Web page is the summary of changes, the list of files and versions expected with that update.
- Evaluate individual patches with various system configurations
- After a patch has been installed verify system changes, ensure the file and system changes are as per the released documentation.
- Ensure that all applications continue to function correctly
- Identify "key" system Registry or file system value(s) that will be used later for validation of successful installation.
- Verify if the software update requires a restart to complete the installation
- Plan for rollback even if testing is successful and no incidents occur
- Test de-installation and application functionality after de-installation
- Monitor event logs for possible errors/warnings during installation
- Obtain and test parameters for silent mode installation/de-installation
- Generate and review log files for all security updates
- After testing in an isolated, non-production environment has been completed transfer patch source codes to a semi-production environment
- Develop a WinBatch scripted solution, keep source codes in a designated development area with restricted access
- Current service packs usually include previously released security patches and hotfixes, update your installation script accordingly to avoid redundancy
- Ensure periodic password change of the generic user ID in the script
- Maintain patch script versioning and descriptions
- Use a semi-production environment as described in subsection below for script testing that initially introduces the patch to a small group of computers
- Enforce change control methodology to track application patch management. A well documented and accurate change log is required for auditing.
- Confirm deployment success by ensuring there is no negative impact on

other systems

## *Establishing semi-production environment*

The solution described in Section Three is intended as a for production implementation. This subsection describes how to test scripted solution on isolated number of machines without affecting production environment. To accomplish that semi-production environment has to be established. Unfortunately, every installation component described in Section Three has to be modified slightly. Once in place, ongoing maintenance is minimal and will require only proper coding of patch installs.
To establish semi-production environment create separate OU in Active Directory for testing purposes. The new Test OU container can be created anywhere in Active Directory structure, but for this scenario the Test OU is a child of Distribution Centers OU. Test OU inherits the Distribution Centers Group Policy thus inheriting production assigned applications. To prevent that:

1. use the block policy inheritance option to block all GPO's from parent container and
2. create Test OU Group Policy with new set of applications

The "Block policy inheritance" option prevents GPO's linked to higher level Active Directory containers i.e.Distribution Centers GPO and Domain GPO from applying. Distribution Centers Group Policy is not set with "No override" option and has no other system settings that would interfere with this installation.
The new `Security Update.msi` package should call test version of `Dynamic.exe`, therefore modify `Static.exe` to call test version of `Dynamic.exe`. After patch installation is coded, compile the script as `TestDyn.exe` and place it in `\\sipad\netlogon\updates`. Modify `Static.exe` to call `TestDyn.exe`, add `Static.exe` to newly created `Security Update.msi` package and assign it to computers in Test OU. Ensure all test computer objects are in Test OU container.
Upon reboot the MSI will install `Static.exe` and on first logon `Static.exe` will call `TestDyn.exe`. Once functionality is verified and the installation is scheduled for corporate-wide implementation, rename `Dynamic.exe` to `Dynamic_v1.exe`, then rename `TestDyn.exe` to `Dynamic.exe` at which point test script becomes available corporate-wide. With FRS replication system the change propagates throughout domain almost instantly. From this point, ongoing maintenance (coding and compilation) is required for the `TestDyn.exe` script only.

## *Validation*

Verifying installation and achieving 100% deployment rate has been constant challenge. To ensure every desktop on any location is patched properly various tools and technologies can be used. Since Microsoft SUS provides basic reporting, even for its own deployed patches, small and medium size

organizations, with Microsoft and non-Microsoft products and security updates, would probably develop custom in-house screening method. As the patching solution is not one size fits all, most likely one verification tool will not either. Use WMIC to verify the existence of the `Security Update.msi` on particular machine. WMIC is another command line utility that uses the power of Windows Management Instrumentation (WMI). WMIC is scripting interface based on aliases. Aliases represent a friendly way of accessing WMI Classes. For example Win32_Product class is referenced by the PRODUCT alias. To verify successful installation of the `Security Update` described in this paper type Startup at the wmic command prompt. The output should list `Static.exe`:

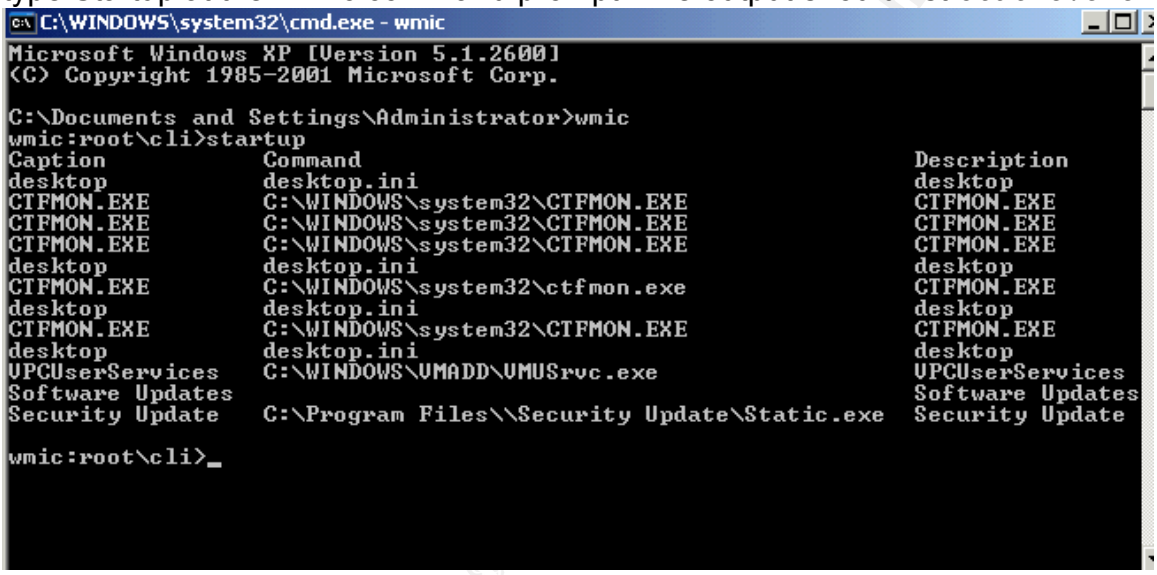


**Figure 9: WMIC output – Security Update mechanism verification**

Hotfix installation can be also verified by typing QFE (quick fix engineering) at wmic command prompt. The result can be directed to an output file that can be even converted to html file format. For more information on WMIC commands refer to the referenced link at the end of this document.

To verify the success of Microsoft SUS handled patches analyze IIS logs. SUS provides IIS log files that can be parsed and imported in Microsoft SQL Server. Again, this might not be solution for small and medium-sized organizations. Analyzing client machines and HTTP GET requests from the `wutrack.bin` file is another way of retrieving some useful information. In addition Microsoft Baseline Security Analyzer (MBSA) can be useful tool for some analyses but it can not retrieve application patch information on remote systems. For more information on scripting with Microsoft Baseline Security Analyzer refer to the link below.

Sipad Komerc is using a custom WinBatch script for software inventory and patch verification. The script periodically scans all machines by checking for existence of specific Registry values or file versions to determine if patch has been applied. In general, the script needs two files: the list of machines to be scanned and the list of values to query. The list of machine names to be scanned can be obtained using the export option in Active Directory Users and

Computers console. The list of values to query is the list of predefined Registry values or file names to be searched. The Winbatch script makes use of the `StrScan` function to seach for values. The output can be later used for further analysis. Systems identified without proper set of installs are then addressed.

## Conclusions

Security patch management has become a technology management imperative. It represents the set of tools, utilities and processes to resolve Windows security problems. Most tools and patch management technologies supported by Microsoft are already well documented and wide variety of supporting documents can be found on the Web. In this paper a way of using other technologies in tandem with Microsoft supported technologies was presented. There is also reasonable expectation that an organization with highly managed desktop environment is following security Best practices. Physical security of domain controllers, secure channel replication between domain controllers, end to end encryption, secure authentication, etc are all Best practices. Application patch management is just one aspect of the larger standard security operational suite of processes. To address application patch management issue multiple tools and technologies are presented.

Presented solution is a sound choice for Sipad Komerc. It utilizes free embedded technologies and successfully leverages with other technologies to fill in the gaps and provide complete patch management solution. It is cost effective and it does not require any additional investment. It reduces the cost associated with manual patch management and unifying all patch management processes to one single entity. The solution is scalable, extensible and can easily incorporate other scripting technologies as needed. Fast application patching results can be achieved relatively easily to mitigate the risks and threats to the overall health and security state of Sipad Komerc Infrastructure resolving multiple Windows Security problems.

## References

[*1] Directions on Microsoft, Peter Pawlak “Software Update service to ease patch distribution” http://www.directionsonmicrosoft.com/sample/DOMIS/update/2002/05may/0502sustep.htm

[*2] University of Manitoba, “What is Microsoft’s Patch management strategy” URL http://netware.umanitoba.ca/forusers/info/winpatch.shtml

[*3] Microsoft, “Microsoft Windows Update Services Frequently Asked Questions” URL http://www.microsoft.com/windowsserversystem/sus/wusfaq.mspx

INS, Felicia M. Nicastro “Implementing a Security Patch Management Process” URL: http://www.ins.com/downloads/whitepapers/ins_white_paper_patch_management_0504.pdf

Microsoft, “Updating Office XP clients from a Patched Administrative Image” URL http://www.microsoft.com/office/ork/updates/oxppatchadmin.htm

Winbatch, “WinBatch White Papers” URL http://www.winbatch.com/whitepapers/automation_software_compar.html

Microsoft TechNet, “Microsoft Security Bulletin Search” URL http://www.microsoft.com/technet/security/current.aspx

Software OnDemand, “WinInstall LE 2003” URL http://www.ondemandsoftware.com/freele.asp

Microsoft TechNet, “Scripting with the Microsoft Baseline Security Analyzer V1.2” URL http://www.microsoft.com/technet/security/tools/mbsascript.mspx

Microsoft TechNet, “WMIC – Take Command-line Control over WMI” URL http://www.microsoft.com/technet/prodtechnol/windows2000serv/maintain/featusability/wmic.mspx