# Global Information Assurance Certification Paper

# Do-It-Yourself Utilities to Simplify Changing Windows Local Administrator Account Passwords Across a Domain

Kurt Hinson
January 16, 2005

GSEC Practical, Version 1.4, Option 2

## Disclaimer

With respect to this document, I do not make any warranty, express or implied, including the warranties of merchantability and fitness for a particular purpose, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Use of a term or brand name in this document should not be regarded as an endorsement.

## Abstract

The purpose of this paper is not to explain how to secure Windows-based workstations and servers. There are numerous checklists, articles and guidelines available that show how to do this. It is also not intended to be a tutorial on programming or scripting, although I hope you catch the vision of what it can do for you. I am only going to cover one of the many items that requires attention to secure Windows systems, and that is changing the passwords on the local Administrator accounts across a domain. I recommend renaming this account although there are different opinions as to the effectiveness of this practice.

Whether it is through the use of scripts, programs, or existing utilities, there is an easier way to change all the local Administrator account passwords across your entire domain. This will enable you to close one of the major holes that can be easily exploited.

*Always evaluate any scripts or programs in a test environment before you execute them against an entire domain!*

## Introduction

I can vividly remember my first task as a "newbie" Information Security team member at my company. All I had to do was change the local Administrator passwords on all workstations and servers across our domain. It sounded easy, but it was anything but that in reality. I spent many endless days connecting to PCs with User Manager for Domains and manually changing each password. Throughout that first experience I started thinking that there had to be a much better way to do this. It seems that there is a never-ending stream of work for a Security Administrator so why not automate this task both to ensure security and make the best uses of our stretched resources? A few years later after a lot of trial and error I have created a full-blown Visual Basic GUI after working with scripts to get the job done in the past. The goals were to create a solution that

would secure the environment, improve the current process by automating the task and do it for a very limited amount of money. It was also to create a stand-alone tool that could easily be used by those without programming or scripting experience. It's one thing to create a utility to automate a task for yourself, but it's even better if you can create a tool that others will want to use. Now that's what I call a solution!

I like to think of it as the "Tom Sawyer" school of system administration. Take something that's tedious and that no one wants to do, and make it easier and even exciting. Now that's a real accomplishment when you can get others excited about security!

## The problem

By default the Windows operating system (Windows NT, 2000 and XP) installs an Administrator account with a blank password. This is common knowledge and perhaps the easiest way for someone to gain unauthorized access to a PC, whether locally or through a remote exploit. In the past, even if the initial account password was changed after the OS installation, it was difficult and tedious to change the account passwords across a domain on a regular basis. Worse yet, the existing password always seemed to get spread around by help desk personnel or others who may have had a legitimate need for it at the time.

The Administrator account cannot be deleted or even locked out, giving anyone unlimited attempts to crack the account. Note that in Windows XP Professional you now have the option to disable the Administrator account which gives us an added option for security. In Windows 2000 you can deny access to the Administrator account by changing the Local Security Policy to Deny Access to this computer from the Administrator user account. In both of these cases please make sure that another user account (local or domain) has Administrator privileges.

According to SANS "The Twenty Most Critical Internet Security Vulnerabilities (Updated)," the two top vulnerabilities include default installs of operating systems and applications, and accounts with blank or weak passwords. In fact, securing the local Administrator accounts by changing the passwords can be one of the quickest ways to close a major hole. There have been documented cases where attackers used this exploit to plant Trojans, backdoors and even key loggers onto systems at universities and other locations. And don't forget that Windows NT, 2000 and XP computers' privileged accounts have access to the hidden hard drive shares like C$. An account with no password gives anyone instant access to the entire drive.

It is also equally important that we change these passwords when anyone in a position to know them leaves. They should also be changed after an event where the password may have been given out, such as in a crisis where others are brought in to help restore or repair virus-infected PCs. A little forethought can prevent having to do this but we all know that in a crisis situation the priorities are on getting the systems back up as soon as possible.

Make it a policy within your organization to routinely change the password. Now that you'll have a simpler way of accomplishing this task there's no reason not to. Remember defense in depth, the process of layering our security. The machine level can often be overlooked because we set up firewalls, secure network servers, use intrusion detection and other high visibility methods. When given a choice between the hard-to-crack system and one that is wide open, most will opt for the one that is the easiest to access.

## In search of…

I set out to find a solution that would be feasible from a time and budget perspective. Would it be cheaper in the end to purchase a product rather than spend the time to research and develop one myself? The answer became clear very quickly as I found no commercial products that did this without a ton of other features I didn't need or without breaking the budget piggybank. Searches of the Internet revealed many wonderful pages filled with scripts, how-to articles, and ideas after I finally crafted a search string that retrieved the kinds of results I needed. I tried many search strings but "local+administrator+passwords" seemed to retrieve the most applicable results. Unfortunately there were no freeware or shareware GUI applications at the time but I was content to start looking at scripts and code towards creating a solution that would work in my environment. With a handful of print outs, several scripting books, some sample code, a large group of book-marked Internet sites and a rough process defined, I set out to do what needed to be done.

## Renaming the Administrator account

Renaming the local Administrator account is a good idea to slow down an attacker. If we can take away something they know, it makes it that much harder for them to get in. You can then create another local account with no privileges named "Administrator" to further frustrate and confuse anyone attempting to gain unauthorized access. This can be scripted as well using the same techniques presented here for changing the password. Please note though that many scanning tools can and will list the accounts on a given PC. Still, as a general practice we felt it best to rename this account.

## Determining the password to use

It's important to decide on a password scheme before proceeding with the actual change. The password you decide on should be strong and contain a combination of upper and lower case characters, numbers and symbols. It should also be at least 8 characters long. It would be advisable to either generate a random password or adopt a substitution scheme where the password itself is easier to remember, however the letters have symbols or numbers substituted. An example of this might be the phrase "lMsecure". A substitution might be "1m$3CuR3" or something similar. I've simply substituted 1 for the l, $ for the s, and 3 for e, as well as used upper and lower case letters. This is sometimes referred to as the "hacker's alphabet". Have you ever been to a hacking site and noticed the odd way the text is written?

You can also create separate passwords, one to be used on all workstations and another to be used on servers. Normally this would mean even more work, but since we're automating this task it adds one more level of security since fewer people will have a need to know the server password.

Remember that this username and password should never be the same as a domain account in the event it is compromised. Also, there are scanning tools that can determine if the password used for an account is the same as the username or null so it's vital we eliminate this risk.

## The solutions

You don't have to be a programmer or have extensive knowledge of scripting to make the job of changing the passwords easier. In fact, there are many scripts and code listings freely available. It was through my initial searching of the Internet that I came across a multitude of ideas and canned scripts from other folks who have already done this. I've included a list of resources in the appendix.

There are many choices when it comes to finding a solution that will work best in your environment: existing tools, WSH (Windows Scripting Host), Perl, batch files, Visual Basic and many more. As a reminder, *always evaluate any scripts or programs in a test environment before you execute them against an entire domain!* I always test against one known machine and then attempt to logon with that username and password to verify it succeeded. Then I expand my trial to a few more test machines before using it on the production network.

### Existing tools

If you have a copy of the Windows 2000 Resource kit you're in luck because you already have a tool that can be utilized for the task. It's called Cusrmgr.exe. The use of this tool is covered in Microsoft Knowledge Base Article- Q272530. The tool can also be used in batch files and examples of how to do it are included in Q272530. If you use SMS Server you can utilize an .ipf file and Cusrmgr.exe to accomplish this as well. See http://www.swynk.com/smsscripts/password.asp for more detailed information on using Cusrmgr.exe with SMS. This tool can also be used to rename the Administrator accounts as well. Be careful you use the right switches though! Use –P (capitalized) because a –p (lowercase) sets a random password. The downside to this option is that you have to write a batch file that includes the name of each machine in the domain you want to change. So if you have a domain with more than a hundred workstations this may be a tedious task in itself. The good news is that there are freely available scripts to do this for you.

### WSH and ADSI

Windows Scripting Host is an Administrator's dream and came as an add-in prior to Windows 2000. ADSI (Active Directory Services Interface) is also an add-in that comes included with Windows 2000 and later. ADSI was developed to easily query for and manipulate directory service objects in scripts or even C and C++. This works in an NT domain as well. The only requirement is that the

machine you run the script from has WSH and ADSI installed. Both of these are freely available for download.

I highly recommend Thomas Eck's book Windows NT/2000 ADSI Scripting for System Administration. It includes a full explanation of how ADSI works as well as code samples for both WSH and Visual Basic. There are also numerous scripts available that you can use or modify for your environment. If there was an Administrator's guidebook to automating every day tasks this would be it.

Best of all, you can write the needed code in Notepad and save as a .VBS file. There are even text editors that are free that you can use for the job. Some of these editors have the ability to show you syntax errors based on the type of script you are writing. Here's a sample of what the code looks like to list all the machines in a domain:

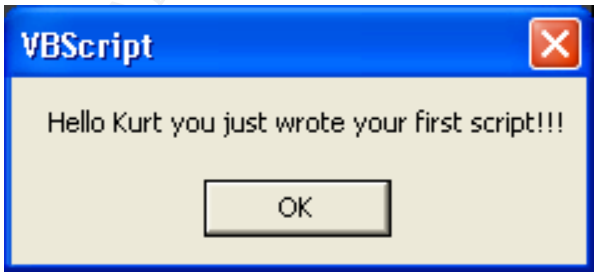
```
Dim MyDomain
Dim LocalComputer
Set MyDomain=GetOject("WinNT://Name_of_my_Domain")

MyDomain.Filter=Array("computer")

For Each LocalComputer in MyDomain
Wscript.Echo LocalComputer.AdsPath
Next
```

Remember that you need to have WSH and ADSI installed on the machine you run this from in order for it to work. You also have to have Administrator rights on your domain. Please see the appendix for code and links for WSH/ADSI password change scripts.

Try it yourself. Write the following code in notepad and then save with a .vbs extension. Then double-click the file to run it.

```
myname=inputbox("What is your name?")
msgbox "Hello "& myname &" you just wrote your first script!!!"
```

It should look like this when you run it:



**WMI (Windows Management Instrumentation)**

This is the new kid on the block. Although it has been around for a few years WMI scripting is becoming more popular. WMI is included in Windows 2000, XP and .Net. It must be installed on the *target* PC as well as on the PC you are running the script from. WMI runs as a service. At its bare bones WMI is another interface to objects you can use in scripts to accomplish administrative tasks.

**PERL scripts**

PERL (Practical Extraction and Report Language) is another script language that originated for UNIX but is now available for Windows. It's free and worth learning in my opinion. This is not exactly the same PERL used in web pages but a way to write scripts that will run on any Windows PC that has PERL installed. There are numerous resources available for code and scripts you can download as well as books, websites, newsgroups and other venues all dedicated to PERL for Windows. PERL can also be compiled into executables. Just like WSH you can write the code in Notepad or any text editor and save with a .pl extension. To get PERL for Windows you can go to http://www.activestate.com. See the appendix for code samples.

**Visual Basic**

When I was in college I took a Visual Basic class and I've been hooked ever since. Visual Basic gives you the ability to write a GUI that can be as simple as filling in a couple of text boxes and clicking on a button (although you must write the code behind the scenes that makes it work). I wrote a Visual Basic 6.0 GUI for changing the passwords across my domain so the other members of my team could easily help out with the process. Windows NT/2000 ADSI Scripting for System Administration covers writing VB applications using ADSI to accomplish this and was the inspiration behind my password changer application.

My application was the result of many hours of work and rework and I'm still adding here and there. It is available for download at http://www.thekurt.net/NetworkUtilities.htm and I'd be happy to share the source code with anyone who asks. See the appendix for a screen shot of it.

**Executing your password changer and follow up**

From experience, I have learned that the best time to run a password changer is during the work day. That's when most of the PCs are turned on. The end user is not affected at all by the process and has no idea it is even occurring. You may find a particular day that is better than others as well. I have found that Mondays and Fridays are when a majority of the employees at my company are off due to modified work schedules. You'll also want to make sure you follow up on any PCs that cannot be changed. They may have been removed from your network, the Administrator account may not have been renamed or there may be other possible causes. Once the initial password run is made and any corrections to individual machines are completed, the whole

process becomes as simple as starting the script or program and checking back later. The WSH script that is listed in the appendix writes the results for failed machines to a text file. You can then rename that file and start again, attempting to change the password on the failed machines only. I also included similar functionality in the Visual Basic GUI I wrote. I have a dedicated PC that I use to run this and other scripts, as it can be resource-intensive depending on the method you use. However, the PC I use is not top-of-the-line and would have gone to salvage had I not claimed it.

## Commercial Tools (freeware currently)

While researching this paper I came across a freeware GUI for changing passwords. The company is currently seeking testers for what will become a retail product. The product is called DC Password Changer (DCPC) and more information is available at http://www.danish-company.com/dcwcm/page/{4D40EC77-0788-48E7-9FB6-B81A51F70CD2}.html I have yet to fully test this application but my initial attempt found that it did not see all the machines on my domain. It is encouraging to see others creating solutions to help make our jobs easier though. For some, paying for utilities may be a small price in the big picture when it comes to securing our environments and saving time.

## Further automation

There are several options for you to consider regarding a mostly "hands-off" approach to changing passwords. One is writing a WSH script that can start itself at a preset interval. There are several ways to accomplish this and an Internet search will reveal ideas and code samples.

You can also take advantage of the Windows Task Scheduler to run your script or program at a preset day and time. This is by far the easiest and is readily available. Of course you'll still need to do your follow-up at some point for the PCs that can't be changed.

## Assistance and Maintenance

Don't overlook the resources you may already have around you for assistance with writing code or scripting. Many of the developers in my company have become my sounding boards and can usually help with syntax problems or when I just get stuck. While they may not fully understand the purpose of the script or program, they are usually quick to point me to a solution that will resolve the issue I have. The Internet is also a great source of ideas, code samples, downloadable scripts and programs, tutorials and even forums where you can post specific questions. There are also rows of books related to this at your bookstore or even public library.

If you write it, maintain it. A college professor I had used to pound this into our heads each class. Ask for a code review with one of the developers or one of your peers. Take the feedback and improve your script or program. It's not as time-consuming as it might sound.

Lastly, be proud of what you have accomplished. Share your work with other Administrators and those who can learn from it. Remember to take out any usernames and passwords prior to sharing your code though. Or better yet, have your script or program prompt you for the username and password at run time.

## Secure the code and the PC used to run it

Regardless of whether you use a script, stand-alone executable or other method to change your local Administrator account passwords, it is totally useless if the code used is available to anyone who wishes to view it. The code contains the account name and password so great care should be taken to mask the password and secure the PC used to run it. This would include physical security of the machine as well as making sure the screen is locked out to anyone wandering by. If you have to place the script in a shared location for others to use, make sure you delete the account and password information and substitute something like "put password here" if you need a reminder for later.

It would be a good idea to run a sniffer while executing your password changer to see if the username and/or password are sent in clear text! There are ways to make sure you are programmatically using secure connections but it depends on the method you are using, so you may need to do some research. I ran a sniffer against my Visual Basic GUI (which uses ADSI) and found that the password is not sent as clear the username. The Ethereal (http://www.ethereal.com) sniffer results I received showed me that my application is creating SAMR (Security Accounts Manager Remote) connections. I did a quick Internet search and found that SAMR is the back-end method used by User Manager and Server Manager to communicate with the Primary Domain Controller. Looking at the ASCII portion of the data verified that the password was not being sent clear text. In my case I did not specifically need to alter my program to do this. The ADS library in Visual Basic 6.0 took care of that for me.

## The impacts

The results of automating local Administrator account password changes appear immediately. It runs in the background freeing up your time to attend to other tasks. You literally conserve time and money for your company as well as creating a more secure environment in the process (this might be the time to ask for that raise). Your peers will think you are brilliant and they may even volunteer to use your new tool.

In my environment this has been the perfect solution and I continually look for ways I can automate more of the tedious, yet necessary tasks for myself and my team. With the ever-growing list of responsibilities, and shrinking budgets that we have to contend with, this has made a huge difference in being able to regularly complete this task with virtually little or no effort.

## Conclusion

Through a little creativity, Internet searching, scripting or even occasional programming (for those so inclined), the task of changing local Administrator account passwords across a domain can be simplified. Once you have a method that works for you, you can adapt it to various other tasks within your organization such as identifying inactive accounts, enumerating a list of all machine accounts, disabling accounts and so much more. Information security is a never-ending job that requires a constant vigilance. Most of us lack a surplus of time and I hope that the information contained here has inspired you to not only change your local Administrator account passwords regularly but to automate the process. We're all in this together. The security practices we implement today could be the virus or hack you don't wake up to tomorrow!

When I first became a Security Administrator I knew very little about scripting and had only some basic knowledge of programming, but it has been the key to doing more with fewer resources. There is an abundance of material to guide you along the way. The best advice I can give you is to find something that works for you and stick to it. The results speak for themselves: your environment will be more secure, you'll have more time to attend to other matters, and you'll be looked upon with high regard by your peers.

## Appendix

### WSH/ADSI Password Changer

A really good description and code at:
http://www.swynk.com/friends/policht/Art062502.asp

One of the best password change scripts I had prior to writing my VB application, by Adrian Grigorof (with a few modifications by me). Copy and paste this code into notepad and save as .vbs file. Requires an input file named Computers.txt that lists the machines in the domain. You could add code here to do that for you or run one of the many scripts already available that will do that for you:

```
'~~Author~~.      Adrian Grigorof
'~~Script_Type~~.   vbscript
'~~Sub_Type~~.      DomainAdministration
'~~Keywords~~.      local user, password, change, adsi

'~~Comment~~.
' This script can be used to remotely change the password for local users on
'several NT workstations or NT servers.
'  It requires that the the account used to submit this script has the rights to
'change local passwords
'  (typically, a member of the Domain Admins group will have this right) The script
'requires Windows Scripting Host
```

```
'  and ADSI installed on the computer used to run it.

'~~Script~~.

' This script can be used to remotely change the password for local users on
‘several NT workstations or
' NT servers. It requires that the account used to submit this script has the rights
‘to
' change local passwords (typically, a member of the Domain Admins group will
‘have this right)
' The script requires Windows Scripting Host and ADSI installed on the computer
‘used to run it.
' Both can be downloaded for free from Microsoft or other scripting sites.
' The script is using a list of computers as input file. The list of computers has to
‘look something
' like:
'
' computer1
' computer2
' ...
'
' If for some reason, one of the computers could not be accessed by the script, it
‘will be saved in
' another file for later submission
'
' The script will output a confirmation of the password change to the screen or a
‘warning
' message if the computer could not be contacted. DO NOT double click on the
‘script or use ‘wscript
' to launch it unless you want a confirmation of each password change to
‘pop-up on the screen.
' Instead, use "cscript setpass.vbs" command from a dos prompt.

' The names of the files, user id and new password have to be initialized (see
‘below)

On Error Resume Next
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Dim fso, fsox, fx
Dim inputFile, outputFile, computerIndex, myComputer,myUser, usr, mDSPath

' set inputFile to match the name of the text file with the list of computers to be
‘changed
' set outputFile to match the name of the text file that will contain the names of
‘the
' computers that could not be accessed by the script (like powered off)
```

```
' later, this file can be used as inputFile to reissue the password change to the
‘computers that
' were not available initially
' set myUser to the name of the local account that needs to have the password
‘changed
' set newPassword to the new password for the user
inputFile = "computers.txt"
outputFile = "Xcomputers.txt"
myUser = "PUT ACCOUNT NAME HERE"
newPassword = "PUT PASSWORD HERE"

Set fso = CreateObject("Scripting.FileSystemObject")
Set f = fso.OpenTextFile("computers.txt", ForReading, True)
Set fsox = CreateObject("Scripting.FileSystemObject")
Set fx = fsox.OpenTextFile("Xcomputers.txt", ForWriting, True)
computerIndex = 1

Do While f.AtEndOfLine <> True
   myComputer = f.ReadLine
        mDSPath = "WinNT://" & myComputer & "/" & myUser & ",user"
   Set usr = GetObject(mDSPath)
        If Err Then
                fx.WriteLine(myComputer)
      ' Comment out the next line if no output on the screen is required
                WScript.Echo CStr(computerIndex) & ". " & myComputer & " could
not be contacted"
                Err.Clear
        Else
           usr.SetPassword newPassword
           ' Comment out the next line if no output on the screen is required
                WScript.Echo CStr(computerIndex) & ".User: " & myComputer & "\"
& myUser & ": password changed"
        End If
        computerIndex = computerIndex + 1

Loop
f.Close
fx.Close

WScript.Echo "Finished"
```

**PERL Password Changer**

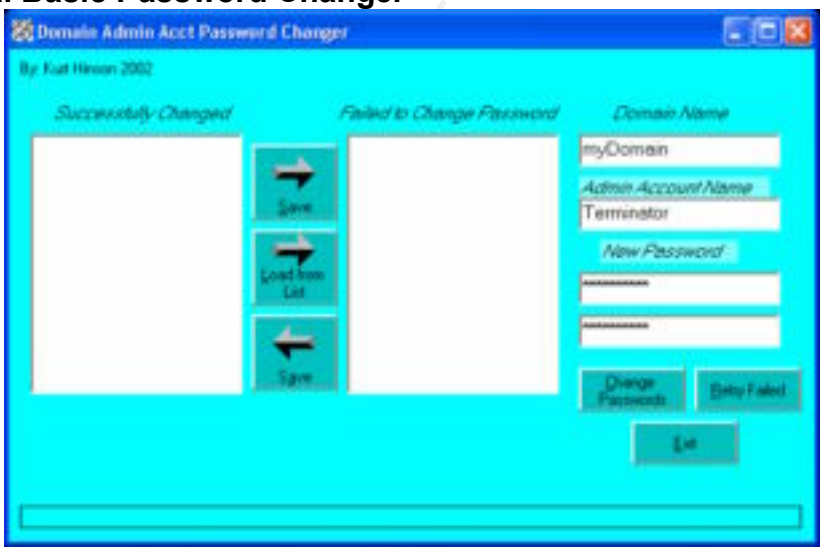
Copy the text below and paste into notepad. Rename with .pl extension. You will need to have PERL installed on the PC you run this from. See http://www.activestate.com for a free version of PERL for Windows.

**Example PERL script from:**
http://www.winscriptingsolutions.com/Articles/Index.cfm?ArticleID=25623
You'll also need to install the Win32 PERL extension
Listing 1: BulkChangePw.pl

```
use Win32::NetAdmin;
use strict;

BEGIN COMMENT LINE
# Declare the script's variables.
END COMMENT LINE
my($Node, $AcctName, $OldPw, $NewPw, $InputFile, $OutputFile, $line,
@INFILE_array) = @_;

BEGIN COMMENT LINE
# Configure the input and output file locations.
END COMMENT LINE
$InputFile = "C:\\Scripts\\inputfile.csv";

$OutputFile = "C:\\Scripts\\logfile.csv";

BEGIN COMMENT LINE
#### No configuration is needed beyond this point. ####
END COMMENT LINE

BEGIN CALLOUT A
BEGIN COMMENT LINE
# Open the input file and populate an array with the server, account, and
password information.
END COMMENT LINE
open(INFILE,"<$InputFile") || die "open0: $!";
BEGIN COMMENT LINE
# Populate the array with all the lines in the input text file.
END COMMENT LINE
@INFILE_array=<INFILE>;
close(INFILE) || die "close0: $!";
END CALLOUT A

BEGIN CALLOUT B
BEGIN COMMENT LINE
# For each of the lines in the array, attempt a password change.
END COMMENT LINE
foreach $line (@INFILE_array) {
```

```
chomp($line);
$line =~ /\S/ or next;
($Node,$AcctName,$OldPw,$NewPw) = split (/,/ , $line);
END CALLOUT B

BEGIN CALLOUT C
BEGIN COMMENT LINE
# Log the results.
END COMMENT LINE
   if( Win32::NetAdmin::UserChangePassword("\\\\$Node", "$AcctName",
"$OldPw","$NewPw")){
      print "Password has been changed on $Node.\n";
      `echo $Node,Success>>"$OutputFile"`;
   }else{
      print "Password could not be changed on $Node.\n";
      `echo $Node,Failure>>"$OutputFile"`;
   }
}
print "PW change run complete!";
END CALLOUT C
```

**Visual Basic Password Changer**




Available from http://www.thekurt.net/NetworkUtilities.htm

## Resources

Active State
URL: http://www.activestate.com
PERL for Windows and tons of other helpful information

ADSI
URL: http://www.15seconds.com/focus/ADSI.htm
ADSI Information, Code snippets etc.

DCPC Password Changer
URL: http://www.danish-company.com/dcwcm/page/{4D40EC77-0788-48E7-9FB6-B81A51F70CD2}.html

Desktop Engineer's Junk drawer
URL: http://desktopengineer.com/
Scripting, tips, VBScript, WMI

Free VB Code
URL: http://www.freevbcode.com
Code, How-To Articles

Swynk.com Windows Server Script Library
URL: http://www.swynk.com/winscript/
WSH Scripts

Win32 Scripting Site
URL: http://cwashington.netreach.net/main/default.asp?topic=news
Scripts, code samples, forum for just about every scripting language out there.

Windows Scripting Solutions
URL: http://www.winscriptingsolutions.com/
Articles, code and everything in between

Winscripter
URL: http://www.winscripter.com/

WMI
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnclinic/html/scripting06112002.asp

## References


"Change a Subnet's Administrator Password". March 2001.
URL: http://www.winscriptingsolutions.com/Articles/Index.cfm?ArticleID=19934.
(August 17, 2002)

Eck, Thomas. Windows NT/2000 ADSI Scripting for System Administration.
New Riders Publishing. March 1, 2000.

"How to Use the Cusrmgr.exe Tool to Change Administrator Account Password on Multiple Computers". August 29, 2000
URL: http://support.microsoft.com/default.aspx?scid=KB;EN-US;Q272530&
(August 17, 2002)

McClure, Stuart and Scambray, Joel. Hacking Exposed. Berkeley, California.
Osborne/McGraw-Hill, 1999.

Neohapsis Archives. April 15, 2002.
URL: http://archives.neohapsis.com/archives/microsoft/various/cifs/2002-q2/0013.html (August 25, 2002)

"Practical Usage of ADSI: Password Management for Machine Accounts and Local Administrators". January, 2001.
URL: http://www.winscriptingsolutions.com/Articles/Index.cfm?ArticleID=16407.
(August 17, 2002)

"Real-World Scripting: Testing and Changing Administrator Account Passwords".
November, 1999.
URL: http://www.winscriptingsolutions.com/Articles/Index.cfm?ArticleID=7461.
(August 17, 2002)

Rist, Oliver. "Lock the Doors on Windows NT in 10 Steps". November 11, 1998
URL: http://www.zdnet.com/windows/stories/main/0,4728,2163194,00.html
(August 17, 1999)

"Script for Changing the Local Administrator Password" (for SMS Server)
URL: http://www.swynk.com/smsscripts/password.asp (August 17, 2002)

"The Twenty Most Critical Internet Security Vulnerabilities (Updated)." Version 2.504 May 2, 2002.
URL: http://www.sans.org/top20.htm (August 17, 2002)

"Windows 2000 Security Alert Blank password in Administrator accounts results in break-ins". June 12, 2002

URL: http://www.unm.edu/cirt/security/win2k1.html (August 17, 200)