Realtime publishers
"Leading the Conversation"

April 2008
Volume 1, Number 3

Windows Administration
in Realtime

Forget about acquisitions and mergers for a moment and think about your electricity and all that you rely on your computer for: personal and business files, financial information, broadband access, videos, photos, music, and more. Increasingly, computers are the hub for managing our lives. And more people rely on APC to protect their hardware and data than any other uninterruptible power supply (UPS) brand.

Why is APC the world's best selling power protection?

For 20 years, we have pioneered power protection technology. Our Legendary Reliability® enables you to save your data, protect your hardware, and prevent downtime. It also guards against a power grid that is growing less reliable every day.

According to the Department of Energy, electricity consumption will increase by 40% over the next 10 years. Yet today, investment in utilities is at an all-time low. It's a "perfect storm" for computer users, one that makes APC protection even more essential.

APC has a complete line of power protection solutions to suit a range of applications. Already an APC user? Get the latest replacement battery cartridge for your unit or upgrade to a newer model.

Find out why 30 million people don't need to worry about losing their data to power problems

APC Solutions for Every Level of Protection:

Home Starting at $59
Best value battery backup and surge protection for home computers.
Back-UPS® ES 550R
8 outlets, DSL protection, 44 minutes of runtime*

Home Office Starting at $99
Complete protection for home and small business computers.
Back-UPS® ES 750
10 outlets, DSL and Coax protection, 70 minutes of runtime*

Small Business Starting at $459
High-performance network power protection with best-in-class manageability for servers.
Smart-UPS® 1000

APC power protection products are available at: BEST BUY, CDW, circuit city, Office DEPOT, STAPLES that was easy.

Register to win an APC 1500VA Battery Back-UPS® system (Model: BX1500LCD) a $199 Value!

Visit www.apc.com/promo Key Code a932w or Call 888.289.APCC x9434 or Fax 401.788.2797

APC® Legendary Reliability®

©2008 American Power Conversion Corporation. All rights reserved. All trademarks are the property of their respective owners.
e-mail: esupport@apc.com • 132 Fairgrounds Road, West Kingston, RI 02892 USA • BK2C7EF-EN *Runtimes may vary depending on load. **Actual prices may vary

Letter from the Editor

Quickly Slowing Down or Slowly Moving Faster?

by Greg Shields

Help me out with this one. I can't decide if I'm getting older, or wiser, or maybe just wider around the middle. But these days I just can't decide if the rate of change in IT technology is getting faster or actually just a little bit slower.

Later today, I'll be shooting out the door to give a Sneak Peek presentation on Windows Server 2008. Now there's a change, or is it? I remember Windows Server 2000, and even NT before that. Those were big changes! I remember back before we had "domains" and when logging into a computer was logging into a computer and not some massive directory that spans the world.

I remember when, to get my email, I used a complicated little program called "mail." Or maybe it was Eudora, or cc:Mail, or even mailx. I remember back when the Internet had about 67 different Web sites instead of 67 gazillion. Back then, to search we used a site called Spider. Half the time you'd run a search for what you thought was a common keyword and good old Spider gave you "No Results."

I remember when a big tool for finding information was our furry friend Gopher. I yearn for the days when we made all kinds of inappropriate jokes about the "finger" command. Heck, I still chuckle thinking about the first time I learned to use "wall" and nearly lost my computer lab privileges when I shared what I learned with friends and strangers alike.

Not long ago I had a day off in Washington DC and stopped by the Smithsonian. I really felt the wheels of change when I saw the Computer History display showing none other than my old Commodore 64.

Although these days I seem to sense fewer radical shifts in IT technology, I still see regular evolutionary changes. In fact, we discuss a few of them in this issue. This month, Don Jones talks about Server 2008 as an evolutionary step over Server 2003. Peter discusses what's new in Exchange 2007, specifically in its anti-spam capabilities. And Jeff takes a step forward with an old topic—reporting server uptime—and gives it a fresh new tool for management in PowerShell.

What about you? Is the world of IT getting just a little more familiar these days or is it still wildly reconfiguring—but in a way that's just comfortable to us. Drop me an email at feedback@realtimepublishers.com and let me know, because I'm still trying to figure this one out.

The Industry Outlook

Hot Topics in the IT Arena

by Don Jones

Vista Problem

Responding to reports of endlessly rebooting PCs, Microsoft pulled an update that was designed to prepare Windows Vista for Service Pack 1, creating another speed bump for the often-maligned new operating system (OS). Although the problem affected a relatively small number of users, extensive media and Internet coverage of the problem focused a spotlight on what some described as another reason to remain shy of Vista.

XP Fixes

For those sticking with Windows XP for the time being, a new XP Service Pack 3 build was posted for public download. Dubbed Release Candidate (RC) 2, a number of signs—including revised release notes and the timing of the download availability—point to SP3 being available soon, perhaps by the time you read this. SP3 is scheduled to be the last major update for XP, which is slated to be unavailable for further retail sales on June 30th.

Fight the Cloud

Microsoft continues to fight the growth of "cloud computing" offered by companies such as Google and Amazon, this time insisting that Amazon's S3 Web-based storage service isn't ready for prime-time, and insisting that "going to the cloud" brings problems to non-tech-smart small and medium businesses. Microsoft continues to promote Windows Server bundles created for the small-to-medium business market as a superior choice, and announced that the revised Windows Small Business Server 2008 will include Office Live Small Business.

Win2008: Security Moves You

A recent CDW poll found that 49% of IT decision-makers cite security features as the benefit of most interest to their company when it comes to Windows Server 2008. 41% selected faster setup and configuration, and 40% said easier administration is a major feature. 35% are looking forward to the new Hyper-V virtualization technology.

Firefox 3.0 Gets Defensive

The new malware-protection features in v3.0 of the popular Firefox browser may cause some grief for your users. The browser relies on a blacklist of known malware sites provided by Google, which includes several sites that provide Firefox add-ins. One of the sites, DownThemAll.net, admits that it has pushed malware in the past; others claim they don't understand why they're on the list. Violating "Google's software guidelines" is enough to get a site blacklisted, including providing links to sites that push malware. The new malware blocker in Firefox essentially puts Google in a position of police authority.

HD Format War Over

While few business users have been worried about whether HD-DVD or Blu-ray would win, the recent failure of the HD-DVD format does provide at least one business benefit: Businesses can now safely standardize on Blu-ray as a high-capacity optical storage format without worrying that the format will become defunct in the near future. Single-layer Blu-ray discs offer a 25GB capacity; a dual-layer disc holds up to 50GB. These capacities make the format a candidate for some types of data backup tasks.

Granular Password Policies

by Richard Siddaway

Each new version of Windows server introduces new functionality. Any new technology has the potential to be disruptive. It is the administrator's task to ensure the introduction of the technology is accompanied by the appropriate management tools to make certain the use of that technology is optimized. There are two very important tasks facing administrators when confronted with new technology. The first task involves understanding what the technology does and how it fits into the requirements of managing a particular environment. There are so many features in Windows that not all of them will be used by every organization. The administrator has to be able to answer the question "Do I understand how this benefits my organization?" The second task is learning how to manage the technology. This article is aimed at helping administrators understand one of the new features of Windows Server 2008, namely fine-grained password policies. In addition, a number of tools that can be used to manage these objects will be discussed.

Windows 2000 and 2003 Active Directory (AD) domains are governed by policies. When a domain is created, two policies are created by default. These are the Default Domain Controller Policy and the Default Domain Policy that are applied as Group Policy Objects (GPOs). The Default Domain Policy is of interest in this article as it holds several important settings, including the password policy and the account lockout policy settings. These settings can be altered or overwritten by other policies applied at the domain level, but these changes are cumulative. Thus, only a single password and account lockout policy can be applied to a domain.

Restricting domains to a single password policy is a constraint on AD design and implementation. Organizations have had to implement multi-domain forests to meet a requirement for multiple password policies. This leads to additional costs. Servers must be purchased to use as Domain Controllers. The environment will cost more to administer, as it is more complex. The multiple domain approach is less than optimal when Group Policy is considered, as it is more difficult, and therefore costly, to ensure that settings are consistent across domains. This can lead to security issues.

Windows Server 2008 changes this situation by introducing a more granular approach to password and account lockout policies, known as fine-grained password policies. Fine-grained password policies enable the settings to be defined for the following attributes:

- Minimum password length
- Maximum password age
- Password complexity requirements
- Minimum password age
- Password history
- Password storage using reversible encryption
- Account lockout duration
- Account lockout threshold
- Reset account lockout settings

All these attributes have the same meaning as when applied through the Default Domain Policy.

> Note that the Kerberos settings cannot be modified using fine-grained password polices. This limitation makes sense, as having multiple Kerberos skew times within a domain, for instance, would be a security and technical nightmare.

Before fine-grained password policies can be utilized in an AD domain, the domain must be at the Windows 2008 domain functional level.

Password policies in Windows 2000 and 2003 are applied via Group GPOs. Fine-grained password policies are not applied via GPOs. Instead, the policy is created as a Password Settings Object (PSO) in the Password Settings Container, which is visible under the System container in Active Directory Users and Computers.

> Note that the Advanced Features option must be enabled from the View menu and that only the Windows Server 2008 version of the tool shows these objects.

The fine-grained policy is then applied to a group or an individual. As with many aspects of Windows security, best practice is to apply the policy to a group rather than to individuals. It is not possible to apply a PSO to an Organizational Unit (OU) directly, as an OU is not a security principal. If a policy needs to be applied to all members of an OU, it will be necessary to create a group that reflects the content of the OU. The membership of the group will have to be kept up to date as the OU contents change. This can be accomplished by utilizing PowerShell scripts to perform the creation, modification, and deletion actions on user objects. In addition, the script could contain a section to update the group membership.

Multiple policies can be applied to a group or individual. A precedence parameter is defined that controls the order in which the policies are applied. Lowest precedence number (highest priority) wins. Settings from multiple fine-grained password policies are not merged. All the settings from a single PSO are applied. In the event that a user has multiple password policies applied (for example, as a member of multiple groups to which password policies are applied or having a password policy applied because of group membership as well as individually), the following rules are applied:

- The PSO linked directly to the user wins. In the event of multiple PSOs being linked to a user, the one with the lowest precedence wins.
- If there are no PSOs linked directly to the user, all the PSOs linked to all the global security groups of which that user is a member are evaluated and the policy with the lowest precedence wins.
- The Default Domain Policy is used if no PSOs are linked to a user account.

Managing Fine-Grained Password Policies with ADSIEdit

Windows Server 2008 does not include any tools specifically for managing fine-grained password policies. The only native tool capable of managing fine-grained password policies is ADSIEdit. To create a policy, it is necessary to first open ADSIEdit, navigate to the System\Password Settings container, and right-click the container object. From the context menu, select New, Object. The only choice available will be msDS-PasswordSettings, so click Next. The wizard then proceeds through a number of screens asking for:

- Cn (the name of the object to be created)
- Precedence
- Password history length
- Password complexity enabled
- Minimum password length
- Minimum password age*
- Maximum password age*
- Lockout threshold
- Reset Lockout Counter*
- Lockout Duration*

The option to modify other attributes is given at on the Finish screen. Click Finish to create the object.

Note that the information on each screen gives the data type (syntax) of the attribute as well as a description of the attribute. When using the Windows Server 2008 version of ADSIEdit, the attributes that represent time values (marked with * in the list) can be input using DD:HH:MM:SS format. Using older versions of ADSIEdit, it will be necessary to enter the values as multiples of 100 nanoseconds multiplied by -1 (a negative value is expected). This is a very good reason for using only the latest version of the tools!

Once a policy has been created, it is necessary to apply it to a group. Open the properties of the Password Settings Object that you want to apply using ADSIEdit. Scroll down to the msDS-PSOAppliesTo attribute. Select it, and click Edit. Click Add Windows Account and the standard AD search dialog will be displayed. Find the group or groups to which you want to add the policy, and click OK.

Note that in ADSIEdit, the group SID is displayed rather than the group name. When you go back into the properties, the group distinguished name is actually shown. Remember that when using the Windows Server 2008 version of Active Directory Users and Computers, if you right-click an object and then examine the properties, there is a tab labeled Attribute Editor which in effect is ADSIEdit built-in to Active Directory Users and Computers. This allows you to modify and apply the policies directly from Active Directory Users and Computers. Sadly, it is not possible to create PSOs directly from Active Directory Users and Computers.

More information about creating and applying these objects can be found in the "Step-by-Step Guide for Fine-Grained Password and Account Lockout Policy Configuration" article at http://technet2.microsoft.com/windowsserver2008/en/library/2199dcf7-68fd-4315-87cc-ade35f8978ea1033.mspx?mfr=true.

Additional Tools for Managing Fine-Grained Password Policies

The introduction of fine-grained password policies is a step forward in terms of optimizing the administration of security policies. It also brings benefits by enabling the long-term reduction in the number of domains an organization needs to maintain with the associated reduction in direct costs (for example, numbers of Domain Controllers and indirect costs in terms of administrative overhead). Managing the policies is possible by using ADSIEdit, but it is not a very satisfactory solution. The creation of policies in particular is a relatively laborious process especially if it is intended to create a number of policies. There are several free non-Microsoft tools available to manage fine-grained password policies, including PSOMgr from Joe Richards, and PowerShell-based tools from Quest and Special Operations Software. It is also be possible to write scripts to perform management actions on these objects; however, using the PowerShell tools mentioned gives a richer experience and saves a lot of work. All the tools mentioned in subsequent sections are free to download.

Please note that in the following examples, the scripts have been split across multiple lines to enhance readability. In practice, the scripts would be treated as a single line of code that would wrap as required. It is assumed that you have a working knowledge of PowerShell and understand the concept of using the pipeline in PowerShell.

PSOMgr

Joe Richards has produced a series of command-line tools for managing AD. PSOMgr.exe is intended to create and manage fine-grained password policies. It can be downloaded from http://www.joeware.net/freetools/tools/psomgr/index.htm.

Inputing

```
psomgr -view -pso
```

will give us a list of the existing fine-grained password policies whereas

```
psomgr -view -dompol
```

will give us a view of the domain policy applied via GPO. This tool also allows us to see both sets of policies by using

```
psomgr -view -allpwdpols
```

The syntax to create a policy is a little complicated. To create a policy called newpso1 with all default settings and a precedence of 8, use

```
psomgr /add newpso1::8 /forreal
Note the use of the /forreal switch to actually perform the action. The following
defaults are used
Using name newpso1 for displayName.
Using default 91 days for maximum password age.
Using default 15 characters for minimum password length.
Using default 24 passwords for minimum password history.
Using default 25 bad passwords for lockout threshold.
Using default 5 minutes for lockout duration.
Using default 5 minutes for lockout observation.
Using default 0 days for minimum password age.
Using default TRUE for password complexity.
Using default FALSE for reversible encryption passwords.
```

To generate a policy with different settings

```
psomgr /add newpso4::15 /lockout 10:10:10 /pwdage 30:0
 /pwdlen 13 /forreal
```

Where precedence is 15; the password ages are a maximum of 30 days and a minimum of 0 with a password length of 13. The lockout duration, threshold, and observation are all set to 10. Other switches exist to apply and remove the policy from groups and to delete policies where required.

PowerShell

PowerShell is Microsoft's new command shell and scripting language. It is .NET based and functions as a set of commands, known as cmdlets, that have a verb-noun syntax using a standardized set of verbs. Information can be piped from one cmdlet to the next in a similar manner to other shells. However, in PowerShell, the information being piped are .NET objects rather than text. PowerShell is part of Microsoft's Common Engineering Criteria and all future major products will have to include PowerShell support. In Windows Server 2008, PowerShell is an optional install feature rather than a downloadable extra. PowerShell version 1 shipped in November 2006. A Community Technology Preview of PowerShell version 2 became available in November 2007.

Quest PowerShell Cmdlets

Microsoft does not produce any cmdlets to manage AD. Quest has produced a suite of PowerShell cmdlets for managing AD that is available for free download from http://www.quest.com/activeroles-server/arms.aspx. This suite includes a number of cmdlets for managing fine-grained password policies. Once the AD cmdlets have been installed, add the following line to your PowerShell profile to have the cmdlets available every time you start PowerShell.

```
Add-PSSnapin Quest.ActiveRoles.ADManagement
```

To view the policies associated with fine-grained password policies, the Get-Command cmdlet can be used as shown:

```
PS> Get-Command -PSSnapin Quest.ActiveRoles.ADManagement | Where{$_.Name -like
"*Password*"} | Select Name

Name
----
Add-QADPasswordSettingsObjectAppliesTo
Get-QADPasswordSettingsObject
New-QADPasswordSettingsObject
Remove-QADPasswordSettingsObjectAppliesTo
```

This illustrates one of the real strengths of PowerShell, namely the capability of using the built-in utility tools to discover more about PowerShell itself. Use:

```
Get-Help cmdlet_name -full
```

in order to display the Help file associated with a particular cmdlet.

We discover that there are four cmdlets available to us to work with fine-grained password policies. Looking at the standard verbs, we can create a new PSO, examine the available PSOs, and add or remove the linkage of a particular PSO to a group. These cmdlets cannot manage or view the default domain policy. This can be viewed by using a script or the Get-BasicDomainPasswordPolicy from Special Operations Software, which is discussed later. It is usually preferable to use cmdlets, if available, rather than write scripts, and in this case, I would definitely recommend using the cmdlet.

The life cycle of a PSO usually starts with the creation of a new fine-grained password policy. This we would achieve as follows:

```
New-QADPasswordSettingsObject -Name 'Medium' -Precedence 5 -PasswordHistoryLength 30
-PasswordComplexityEnabled $true -MinimumPasswordLength 10
-MinimumPasswordAge (New-TimeSpan -Days 1 )
-MaximumPasswordAge ( New-TimeSpan -Days 20)
-LockoutThreshold 3 -LockoutDuration ( New-TimeSpan -Minutes 40 )
-ResetLockoutCounterAfter ( New-TimeSpan -Minutes 30 )
-Description 'Medium level policy'
```

It is possible to link a policy directly to one or more groups during creation by using the -AppliesTo parameter. I prefer to create the policy and then apply it as a second stage once I am sure that the policy meets my needs.

To link a fine-grained password policy to a group

```
Add-QADPasswordSettingsObjectAppliesTo -Identity 'Medium'
-AppliesTo 'Manticore\Finance'
```

This applies the fine-grained password policy that we created in the previous step to a group called Finance in the Manticore domain. The policy could be linked to an individual as well as a group. To remove a linkage

```
Remove-QADPasswordSettingsObjectAppliesTo -Identity 'Medium'
-AppliesTo 'Manticore\Richard'
```

To view the fine-grained password policies in use

```
Get-QADPasswordSettingsObject | Format-List
```

To compare policies

```
Get-QADPasswordSettingsObject | Foreach {Get-QADPasswordSettingsObject $_.Name |
```

If more detail is required

```
Select Name, MinimumPasswordlength, MaximumPasswordAge, PasswordComplexityEnabled} |
Format-Table -auto
```

will provide a formatted display showing the policy name, minimum password length, maximum password age, and whether password complexity is enabled. If it becomes necessary to delete a policy, the Remove-QADObject cmdlet can be used.

PowerGUI

PowerGUI is a graphical tool that sits on top of PowerShell. It is possible to create PowerShell scripts that run inside PowerGUI and rather than displaying the results at the command line, they can be output to a data grid from which further actions can be performed based on the data. One of the really powerful points of PowerGUI is that the code being run to produce the data is viewable in a separate tab, making it much easier to understand what is actually happening. PowerGUI is created by Quest and can be downloaded from www.powergui.org.

As part of the bundle, a fully featured PowerShell editor comes with PowerGUI. It features color highlighting, tab completion, and intellisense-like features for PowerShell cmdlets, .NET classes, variables, and WMI classes. PowerGUI scripts are stored as a set of nodes in a Windows Explorer-type view. These nodes can be exported as snap-ins to be used by other PowerGUI users. A number of snap-ins are available for download from the PowerGUI site, including one for managing fine-grained password policies that uses the Quest cmdlets discussed earlier (see Figure 1).

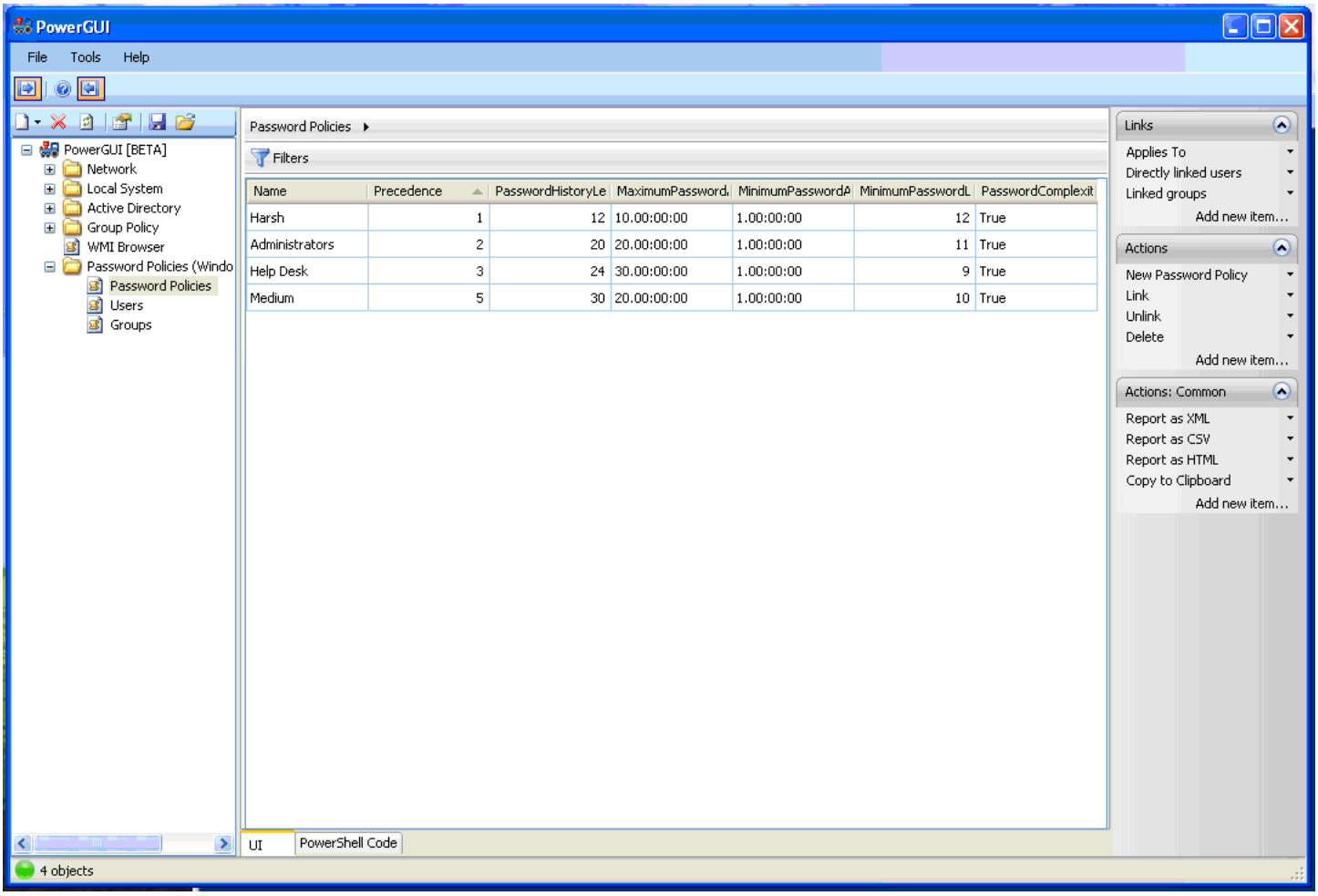

Figure 1: PowerGUI password policy snap-in.

PowerGUI gives us all of the Windows forms features that we expect in modern GUIs. The attributes that are displayed can be controlled by right-clicking a column header and then selecting the attributes that you want to see. The display can be sorted by clicking the column header, and column order in the display can be altered by dragging the column to the desired position.

The strength of this tool is not just its ability to display PowerShell data in a graphical display but the fact that many, if not all, of the commonly used administration scripts can be stored in PowerGUI nodes to be instantly accessible and usable. The nodes can be exported to XML files so that other users can import the functionality. As the nodes can be easily edited, it is a simple matter to add functionality by configuring additional nodes.

PowerGUI offers a number of other features in terms of administering fine-grained password policies, in that the right panes gives us functionality to view which users and groups the policies apply to, create a new policy, and link or unlink a policy. We can also delete a policy.

SpecOps Command and SpecOps Password Policy

Special Operations Software of Sweden, www.specopssoft.com , produces a suite of tools for managing your environment. One of the most intriguing and powerful is SpecOPs Command, which enables the delivery and execution of PowerShell scripts via GPO. A different product, SpecOPs Password Policy, can be used to manage fine-grained password policies. As well as a very neat GUI (see Figure 2) for managing these objects, a PowerShell snap-in is installed.

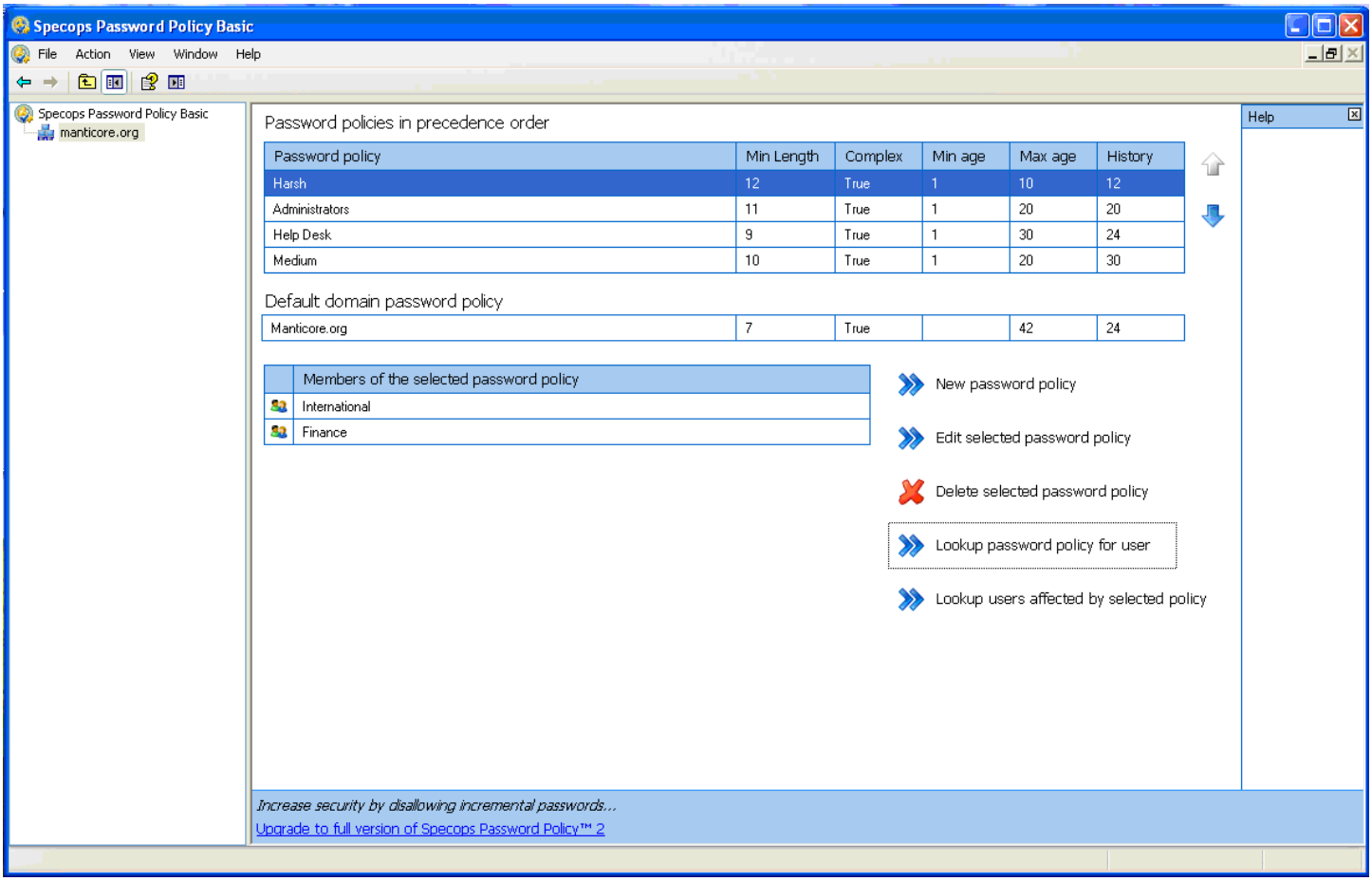

Figure 2: SpecOps GUI for managing fine-grained password policies.

The GUI offers very similar functionality to that offered by PowerGUI. Note that it is a very well-behaved install and that it will install on the PowerShell V2 CTP. As usual with PowerShell snap-ins, you need to need to add it into PowerShell. Put the following line in your profile and it will be loaded every time you start PowerShell.

```
Add-PSSnapin SpecopsSoft.PasswordPolicyBasic
```

The snap-in supplies a number of extra cmdlets.

```
PS> Get-Command -PSSnapin SpecopsSoft.PasswordPolicyBasic
Name
--------
Get-BasicAffectedUsersForPasswordPolicy
Get-BasicAffectingPasswordPolicy
Get-BasicDomainPasswordPolicy
Get-BasicPasswordPolicy
New-BasicPasswordPolicy
Remove-BasicPasswordPolicy
```

Note the Basic prefix to all of the cmdlets. A lot of third-party cmdlets follow this pattern and have a prefix to identify their cmdlets.

migration | optimisation | measurement & management

centiq

Centiq Ltd is an IT Services company focussed on premium level IT consulting, architecture, design and implementation services. We are experts in Migration, Optimisation and Measurement & Management.

Centiq Ltd
Unit 1 Charles Park
Charles Way
Cinderhill Road
Nottingham
NG6 8RF

Tel: 0115 951 9666
Fax: 0115 951 9555
email: info@centiq.co.uk

www.centiq.co.uk

Simplify your IT, meet your business needs better

Many businesses grow rapidly in unplanned ways, often the burden of complexity is carried by their IT function.

Centiq helps you maximise your IT investments while ensuring that the IT function needs all your organisation's overall needs more effectively.

Our tried and tested methodologies help you achieve these objectives, reduce the complexity of your computing environments, and contain the risks of growth and IT expenditure.

Talk to us about

- Infrastructure Optimisation
- Virtualisation
- Active Directory – planning and implementation
- Exchange 2007 upgrades
- SQL server upgrades

Microsoft GOLD CERTIFIED Partner

The default domain policy can be displayed using a single cmdlet:

```
PS> Get-BasicDomainPasswordPolicy
Name                                Manticore.org
MinimumPasswordAge              1
MaximumPasswordAge              42
MinimumPasswordLength               7
ComplexityEnabled               True
PasswordHistoryLength               24
ReversibleEncryptionEnabled     False
LockoutDuration                     30
LockoutResetCounter                 30
LockoutThreshold
```

A new policy can be created by using

```
New-BasicPasswordPolicy -ComplexityEnabled $true -Name test4 -MaximumPasswordAge 5
-MinimumPasswordAge 1 -Precedence 6 -PasswordHistoryLength 12 -MinimumPasswordLength 9
```

Notice that timespans are given as a number of days or minutes as appropriate, rather than using a timespan object as the Quest cmdlets do. One method is not more correct than the other; it just depends on your particular needs as to which is the more efficient for you.

There does not seem to be any cmdlets to modify or apply the password policies. That task can be accomplished through the GUI . A big plus is the cmdlets to show which users are affected by a policy and which policies affect a user. To see the individual users affected by a particular policy use

```
Get-BasicAffectedUsersForPasswordPolicy -Name 'Medium'
```

but note that it only displays users that have the policy applied directly; it does not show the users that have the policy applied because of membership to a group to which the policy is applied. To see where policies are applied, we have two options

```
Get-QADPasswordSettingsObject | Select Name, Appliesto | Format-List
```

and

```
Get-BasicPasswordPolicy | Select Name, Members
```

Either will display a list of polices and the target groups.

To investigate the resultant policy applied to an individual user, the following code will display the policies applied to an individual user. Only the policies directly applied to an individual will be listed. Policies that may be applied because of group membership will not be shown.

```
Get-QADUser gaway -IncludedProperties Msds-ResultantPSO |
Format-Table Name, Msds-ResultantPSO
```

If we want to view the users to which a policy is applied by virtue of their group membership, we need to resort to a little piece of scripting. This is one of the strengths of PowerShell in that the cmdlets can be used as standalone commands (in a similar way to traditional DOS commands) but by use of the pipeline capability, the functionality can be combined and then the same cmdlets can be used in scripts to further extend our administration capabilities. In the following script, we retrieve the information on a particular PSO and then using the members property (which holds the groups to which we have applied the policy), we iterate through the groups and use one of Quest cmdlets to dump the group membership list. This could be achieved by scripting only but would be quite a few lines longer.

```
Foreach ($member in (
(Get-BasicPasswordPolicy -Name 'NewPolicy').Members)){
   Get-QADGroup $member.distinguishedname.ToString() |
Select Name
   Get-QADGroupMember $member.distinguishedname.ToString()
   }
```

This still does not allow us to discover which policies affect a user via group membership. To discover this information, we need another small script as the following example illustrates. We retrieve the user information from AD and then iterate through the list of groups in the memberOf attribute. For each group in that list, we retrieve the fine-grained password policy information using Get-QADPasswordSettingsObject and check the AppliesTo attribute to determine whether the policy applies to that particular group.

```
Foreach ($group in ((Get-QADUser 'Gone Away').MemberOf )){
   $group
   Get-QADPasswordSettingsObject |
Where {$_.AppliesTo -contains $group} | Select Name
 }
```

The GUI has one feature that may be useful. When it starts up, it will warn you whether there are fine-grained password policies in the domain that have the same precedence number and offer to alter the precedence values to prevent this happening. The actual precedence values are not displayed but the policies are listed in precedence order in the GUI and it is possible to move individual policies up and down in the precedence order. To view and, if necessary, modify a policy, double-click the policy in the list and a dialog box showing the settings will appear. The dialog box can also be used to manage the groups to which a policy is applied. It is also possible to use the GUI to delete policies and lookup users affected by a particular policy or to discover which policy affects a particular user.

The PowerShell Community Extensions produce a provider for AD that allows you to access the data store as if it was the file system (that is, you can perform a "dir" action on your AD and use other file system-orientated PowerShell cmdlets). The extensions are a free download from http://www.codeplex.com/PowerShellCX. This cannot be used to administer fine-grained password policies, as the provider does not recognize the Password Settings Container.

Summary

One of the great advantages of all the tools discussed in this article is that they do not need to be installed on a Domain Controller. Install any, or all, of them on your administrative workstation to gain access to this functionality. The tools will work on Windows XP and Windows Vista. Personally, I prefer to use the PowerShell-based tools as they have become a consistent part of my administration toolset. This means I can use the same tools and techniques to manage all the objects in my AD so that I obtain maximum management benefit from the toolset and the time I spent learning to use the toolset. I can also leverage the learning effort to administer other Microsoft products that are PowerShell-enabled such as Exchange 2007.

Fine-grained password policies seem like a wonderful idea on the surface, as they remove one of the restrictions currently constraining AD design and implementation. However, as with any tool of this nature, there are good things and bad things that can be done with them. Good ideas for their use include increasing the strength of the password used by accounts with high-level administrative access to your domain or modifying the account lockout policies for Administrator to reduce the number of unsuccessful attempts before the account is locked. One very bad idea that should be resisted at all costs is the concept of reducing the length of the password for a particular group of individuals; again, administrators come to mind as they are ones who are most likely to know about this feature!

Used in a sensible manner, fine-grained password policies can add real benefits to your organization and help optimize the management of your AD domains.

Richard Siddaway, Microsoft practice leader, Centiq Ltd, can be contacted at RSiddaway@centiq.co.uk.

The Deep Dive

Becoming Proactive — Taking the Firefighting out of the Network

by Greg Shields

"Beep, beep."

Deep down inside, don't you just hate that sound? It always goes off in the middle of the night, right after you've finally drifted off to sleep. Whether it's a pager, your cell phone, or one of those new fancy Blackberries or Windows Mobile devices, it all sounds the same—something's broken on the network and you won't be sleeping again tonight.

What's interesting about that sound is that in many cases it needn't happen at all. In a lot of cases, the reason the Blackberry goes bump in the night has more to do with preventative things that weren't done on the network than actual faults in the systems themselves. Managing a Windows network is a complicated activity involving lots of little tasks, and the proof of that is in the minutiae.

The hard part is that many administrators find themselves getting caught up in just those minutiae, constantly running from problem to problem, band-aiding where they can and outright ignoring those they can get away with. Reactive-mode network administration is an exercise in gut strength—simply surviving each day's onslaught of Help desk tickets and complaining users.

Proactive Isn't Easy

Now, all of that being said, the goal for nearly every environment is to get proactive: flush the firefighting out of the system and solve problems before they happen. Understanding this, the first rule of getting proactive is an admission that the process to get there will involve an initial burst of extra time and effort before the savings kick in.

That's the hardest part for many administrators. Learning the proper scripting knowledge you need to automate tasks involves hours spent studying and building your script quiver. Getting to know the Windows registry to the comfort level necessary to directly manipulate it is a long haul of trial and error. Developing your skills in building software packages for automated software installations is no trivial task. So spending the time to figure out exactly how to do it will involve an equal expense of time and brain matter. Turning on systems monitoring is one thing, but tailoring its reports to only alert when necessary means a lot of fiddling in front of monitoring consoles. All of these can be considered the "advanced" tasks of systems administration.

The second rule of getting from reactive to proactive is that doing it will require a conscious effort to move away from the Windows graphical user interface (GUI) whenever possible. If you find yourself constantly performing tasks without using the command line or scripts, you're not performing those tasks in a way that can be automated. Virtually every management task on Windows systems has a command-line adjunct or some level of scripting exposure. The benefit here is that once the extra few minutes are spent finding a scriptable way to accomplish a task, every time that tasks needs to be done again, it can be done so rapidly.

Proactivities

So we've drilled home the idea that the first step in getting proactive is learning to automate. That automation involves learning scripting skills such as VBScript and PowerShell and management skills such as the working with the Windows registry and packaging software. But it also involves bringing in-house the necessary systems management platforms that ease some of the tasks mentioned in the previous paragraph. With the right management tools in place, creating and automatically distributing software packages is trivial, securing systems is a piece of

cake, and centrally controlling their configuration comes naturally.

Let's take a look at 10 common tasks that in the unautomated and reactive network are painful and time-consuming. These same tasks, with the right tools and experience in place, can also be highly automated.

Hardware Inventory

Can you remember the last time you walked the building to write down the serial numbers and computer names for every desktop on your network? With Microsoft's Windows Management Instrumentation (WMI), much of this information is stored locally to each desktop. Using WMI queries either through scripts or via an asset management solution can mean no longer haunting the hallways.

Software & License Inventory

The same information that WMI and the registry expose for hardware information can be used to identify where software is installed. The right management platforms can scan the contents of the registry to locate and log any installed application. That centralized inventory means you're sure that the number of licenses you've purchased equals the amount of software you've installed.

Remote Support

Windows XP introduced the idea of Remote Assistance to the network, but getting users to correctly use it was and is challenging. Thus, additional toolsets have been developed to take the guesswork out of over-the-shoulder support. Advanced (and yet inexpensive) platforms can even extend the reach of remote support outside the LAN to anywhere on the Internet. If your network includes the use of laptops off the LAN, consider a management platform that can work with those machines as if they were local.

Configuration Control

One of the biggest causes of desktop failure is when configurations deviate from established baselines. When your users install the latest versions of quasi-legitimate software (you know the kinds!), or when they perform unmonitored Internet surfing, they introduce the threat of corruptive code. More bad code equals more problems which equals more of your time fixing them. The problem with this is that without technical controls, users will never be incentivized to care about their desktop configuration. Finding the right tools that lock down desktops and enforce configuration control will immediately reduce your Help desk caseload.

Compliance

Governmental and industry regulations are numerous and complicated. Just understanding what and how they want you to secure can be a nightmare. You're a Windows administrator and not a lawyer, so trying to wade through and decipher the thousands of pages of recommendations shouldn't be your job. Some of the best systems management platforms have already done this for you. These platforms have identified down to the individual setting exactly what you need to configure to ensure compliance in your network. Best of all, they're management platforms, so they can identify down to the individual machine which ones aren't up to snuff.

Automated Application Deployment

For a long time considered the holy grail of systems management, automated application deployment is now commonplace in virtually all systems management platforms. Automated application deployment has three major steps: repackaging a software install so it runs "quietly" without prompting the user, deploying the software through a management interface, and then reporting on the success of the mass install. Back before the widespread embrace of Windows Installer (think .MSI files), the repackaging process was very much the hard part, but over time, even this step has become trivial. If your organization is still sending technicians out to desktops to install software, you're wasting your techs' and your users' time. Putting an effective automated deployment solution in place will save shoe leather and reduce the amount of time you're spending on this otherwise highly manual activity.

Patch Management

Patch management is another activity that once was complicated. Today, toolsets for approving and deploying patches are quite good. Finding the right patch management tool means potentially looking away from no-cost products towards those that integrate with the other activities discussed here. Patching also encompasses software other than products from Microsoft. Thus, good patch management also means looking for solutions that support the update of non-Microsoft software as well, because vulnerabilities exist in virtually all installed software.

Network Monitoring

In the unmonitored network, the worst kind of Help desk ticket is the one that starts, "The network is slow today." The data that traverses a Windows network is nothing that you can see or feel. So tools that can peer into traffic conditions and report on what they see is critical to being able to truly resolve these kinds of Help desk tickets. Network monitoring toolsets these days can be inexpensive (even free) and fully featured. Implementing them into your network means knowing when bandwidth is saturated and users will start calling.

Windows Event Alerting

Network monitoring is only one way to see the behavior and health of the network. Another is aggregating data from across all the event logs on all your servers. The Windows Event Log is a wonderful tool that natively lacks one major feature—a good way to collect and filter on all events across all machines at once. Thus additional tools are necessary. Effective systems management and monitoring tools can yank these events from individual servers and logs into a searchable and alertable interface so that you know what's going on everywhere at once.

Integrated Reports

Lastly, being able to report on all these activities is of great importance. There's nothing like doing a great job administering your network only to find out that no one knows or cares what you've done. A best-in-class management solution will give you the reports you need to hand to your boss, your auditor, and your users to enlighten them as to the quality and health of your network, as well as the great job you've done.

The Moral

The moral of this story is that getting from reactive to proactive in your network administration isn't necessarily an easy task. There's a lot to learn and plenty of ways to do it incorrectly. But with that downer also comes the knowledge that tools—both comprehensive and inexpensive—exist to ease your learning curve and improve your ability to automate. Finding just the right toolset is the hard part. Consider these 10 tasks when you're looking at management tools, and you just might find the right one at the right price that suits your needs and ultimately makes the pager stop beeping.

Greg Shields, MCSE: Security, CCEA, is an independent author, speaker, and consultant, based in Denver, Colorado. With more than 10 years of experience in information technology, Greg has developed extensive experience in systems administration, engineering, and architecture. Greg is a contributing editor for both Redmond magazine and MCPmag.com, authoring two regular columns along with numerous feature articles, webcasts, and white papers. He is also the resident editor for Realtime Publishers' Windows Server Community at www.realtime-windowsserver.com. Greg is currently finishing his new book Windows 2008: What's New, What's Changed through SAPIEN Press.

Focal Point

Why You Will (or Won't) Upgrade to Windows Server 2008

by Don Jones

It's finally here: Windows Server 2008, the five-years-in-the-making new server operating system (OS) from Microsoft. By this point, you've doubtless read a dozen articles on what's new in Win2008 (and if you haven't, fellow columnist Greg Shields has a book that only covers what's new and changed in it), so I'm going to steer clear of the "new feature list." Instead, I want to look at the top reasons many companies will be using Win2008 in the very near future—and the top reasons why many *won't*. You see, Win2008, much like Windows Vista, isn't an "instant win" for Microsoft. Sure, it's a good OS, and they put a lot of work into it, but it doesn't automatically follow that companies should rush right out and upgrade all their servers. The sheer number of Win2000 servers still out there are proof that companies need compelling reasons to upgrade, and they need a *lack* of "negatives" that keep them from upgrading.

Two Words: Server Core

I think the biggest argument for using Win2008 sooner rather than later is the new Server Core installation option. This is *not* an edition of Win2008; rather, each edition—Standard, Enterprise, and Datacenter—can be installed in Server Core "mode," if you will. This is a one-time, one-way decision that you make during the initial install; you can't upgrade Server Core to the full install later without starting over from scratch. Server Core is intended to be a small-footprint, as-few-moving-parts-as-possible installation. Its primary business benefits are fewer vulnerabilities and fewer patches because there are fewer pieces of code actually running. Like Win2008, Server Core is organized around the concept of *roles*; you install only the specific roles you need, thus further reducing the amount of code. Server Core is capable of supporting only a few roles: Domain controller, File/Print server, DHCP server, and DNS server—in other words, the primary infrastructure roles. It's also capable of being an IIS Web server, albeit a dumbed-down one that can only serve static Web pages. Server Core does *not* run the Microsoft .NET Framework because the Framework has dependencies on pieces of Windows that don't exist in Server Core.

Server Core doesn't support any kind of robust graphical user interface (GUI). Log on to the console, and you'll see a command-line window (not Windows PowerShell, but the old Cmd.exe; PowerShell needs the Framework, which doesn't exist on Server Core). Server Core has enough GUI capability to display the dialog boxes often raised by device driver installers, and enough to run most of Windows Notepad so that you have a simple text editor.

So how are you *supposed* to manage the services running on Server Core? The same way you've always been supposed to manage a Windows Server: remotely, using tools such as Active Directory Users and Computers, the DNS Management console, and so forth. Yes, you can Remote Desktop into Server Core but you should have very little need to do so apart from adding new roles (which is accomplished using a Microsoft-provided VBScript that you run from the command line because Server Manager isn't included on Server Core).

So why is Server Core the "killer app" for Win2008? It's pretty much what we've always wanted in an infrastructure server. Stick it on a 2U rack mount server with an x64 processor (or two), gobs of RAM, and a couple of decent-sized disks, and you've got the perfect domain controller/DHCP server/DNS server. Tack on a SAN, and you've got a great file server or print server. If you have a completely static Web site, then, er… you can use IIS. Honestly, the value of having IIS in there eludes me.

Server Core will also be able to be a virtual machine server thanks to Microsoft's Hyper-V technology. This will give you the thinnest possible version of Windows on which to base a virtual infrastructure, assuming you choose to use Hyper-V rather than something like VMWare ESX Server; Hyper-V hasn't proven itself in the performance arena yet, so that's an open question.

Two Words: Why Bother?

On the downside, Win2008 doesn't seem to offer a bevy of compelling new features to a company already based on Win2003. Yes, there definitely are new capabilities. Some of those, however, require windows Vista on the client side, and Vista hasn't been taking the corporate world by storm quite yet. Win2008 does promise more stability and security, and with its more modular design, should offer somewhat easier maintenance. However, none of those are issues absolutely killing Win2003 shops—or indeed, many Win2000 shops.

Hyper-V is a great new piece of functionality and the licensing for Datacenter lets you run unlimited virtual machines, but many companies who've already gotten serious about virtualization have done so with VMWare or Virtual Server, so they haven't been sitting around waiting for Hyper-V.

In short, Win2008 is a great evolution over Win2003, but it isn't a *revolution*. Upgrades will cost you, of course, both in licensing and in time to deploy; if your company hasn't been bouncing up and down waiting for some specific new feature in Win2008 that solves a specific business problem, well, then it's likely management will feel fine about waiting a bit. And it's tough to point at any single group of Win2008 features (aside, perhaps from Server Core) that make the "let's just wait" attitude seem wrong.

Now, I don't want to be down on Win2008. It is a great piece of software, and I like it a bunch more than its client-end cousin, Vista. Once you *do* move to Win2008, you'll find a bunch of features that make sense and help solve smaller problems that have doubtless been plaguing you. The new Group Policy capabilities, for example, are just fantastic. And Win2008 does offer solutions for specific problems that a lot of companies—especially those with strong security policies or who are dealing with regulatory compliance issues—will find useful, such as network access protection, quarantine, and the like. The new AD features targeted at branch offices are well thought out and if you have domain controllers scattered in branch offices, those features may be worth an upgrade of your domain controllers at least.

So you *should* look closely at what's in Win2008 to see if there is something that meets a specific need you have. Don't be alarmed, though, if there aren't any fireworks-worthy new features. Be relieved, in fact, that Win2008 is similar to the step we took from Win2000 to Win2003—fairly gentle.

Don Jones is a Series Editor for Realtime Publishers, and the Director of Training and Publishing for SAPIEN Technologies. Visit him online at www.ScriptingAnswers.com.

Practical PowerShell

What's Up Doc?

by Jeffery Hicks

The Problem: You need to know how long a server has been running. That is, how long has it been since the computer last booted.

The PowerShell Solution: As you might expect, Windows Management Instrumentation (WMI) plays a big role in this:

```
PS C:\> get-wmiobject -query "Select LastBootUpTime from win32_operatingsystem"
```

If you run this expression on your computer, you'll see a value like 20080208173018.500000-300, which is a WMI datetime string. Fortunately, it doesn't take much in PowerShell to get this into a more user-friendly format:

```
PS C:\> $last=get-wmiobject -query "Select LastBootUpTime from win32_operatingsystem"
PS C:\> $last.ConvertToDateTime($last.LastBootUpTime)

Friday, February 08, 2008 5:30:18 PM
```

Listing 1 provides a complete function I wrote to simplify the entire process and provide a little extra value.

> You can download the code from the Realtime Web site at http://www.realtime-windowsserver.com/code/v1n3_Practical_PowerShell.zip

```
Function Get-Uptime {

Param([string]$computer="localhost",
    [System.Management.Automation.PSCredential]$credential)

    Function Get-Msg {
        Param([string]$text)

        # build a text string of information that can be passed
        # to different pipelines like Debug

        #here are some variables you might want to work with
        $computer=$env:computername
        $domain=$env:userdomain
        $user=$env:username
```

```
        $timestamp=(Get-Date -Format g).toString()

         [string]$msg="[$computer - $domain\$user] $timestamp $text"

         write $msg

    }

    $errorActionPreference="SilentlyContinue"
    $DebugPreference="SilentlyContinue"

    #store computers that fail in a global variable,
    #$Global:GetUptimeFailed. If the variable already
    #exists with members then do nothing, otherwise
    #create the variable.
    if ($Global:GetUptimeFailed.Count -lt 1) {
        $Global:GetUptimeFailed=@()
    }

    Write-Debug (Get-Msg "Connecting to $computer")
    $query="Select LastBootUpTime from Win32_OperatingSystem"

    #define repeated script blocks
    $WMI={Get-WmiObject -query $query -computername $computer -ea silentlycontinue}
    $WMI_Cred={Get-WmiObject -query $query -computername $computer -Credential $credential
-ea silentlycontinue}

    Write-Debug (Get-Msg "Executing $query")

    if ($credential) {
        Write-Debug (Get-Msg "Connecting with alternate credential")
        $time=&$WMI_Cred
        }
    else {
        $time=&$WMI
        }

    if ($time) {
       Write-Debug (Get-Msg $time)
       }
    else {
        Write-Warning "Failed to get WMI information from $computer"
        Write-Warning $Error[0]
        Write-Debug (Get-Msg "`$time is empty")
        Write-Debug (Get-Msg "Adding $computer to `$Global:GetUptimeFailed")
        $Global:GetUptimeFailed+=$computer.toUpper()
        Write-Debug (Get-Msg "failed computers to date:")
        Write-Debug (Get-Msg $Global:GetUptimeFailed)
        Write-Debug (Get-Msg "Exiting function")
```

```
        return
       }

    Write-Debug (Get-Msg "LastBootUpTime=$time.LastBootuptime")

    [Datetime]$t=$time.ConvertToDateTime($time.Lastbootuptime)

    Write-Debug (Get-Msg "`$t=$t")

    [TimeSpan]$uptime=New-TimeSpan $t $(get-date)
    Write-Debug (Get-Msg "`$uptime = $uptime")

    Write-Debug (Get-Msg "Creating new object `$obj")

    $obj = New-Object system.Object

    $obj | Add-Member -MemberType NoteProperty -Name "Computer" -Value $computer.
toUpper()
    $obj | Add-Member -MemberType NoteProperty -Name "LastBoot" -Value $t

    Write-Debug (Get-Msg "Adding $($uptime.days)d $($uptime.hours)h $($uptime.minutes)m
$($uptime.seconds)s")

    $obj | Add-Member -MemberType NoteProperty -Name "Uptime" -value "$($uptime.days)d
$($uptime.hours)h $($uptime.minutes)m $($uptime.seconds)s"
    $obj | Add-Member -MemberType NoteProperty -Name "Days" -Value $uptime.Days
    $obj | Add-Member -MemberType NoteProperty -Name "Hours" -Value $uptime.Hours
    $obj | Add-Member -MemberType NoteProperty -Name "Minutes" -Value $uptime.minutes
    $obj | Add-Member -MemberType NoteProperty -Name "Seconds" -Value $uptime.seconds

    Write-Debug (Get-Msg "Writing `$obj")

    write $obj

    Write-Debug (Get-Msg "Exiting function")
}
```

Listing 1

Before I get into the meat of the function, let me go over the "extra value" features. As you look through the function, you'll see these lines towards the beginning:

```
$errorActionPreference="SilentlyContinue"
    $DebugPreference="SilentlyContinue"
```

The setting for $errorActionPreference is set to "SilentlyContinue." When the function executes, this will override the parent scope's default setting of "Continue." This setting will turn off any error messages in the error pipeline. When used in conjunction with the cmdlet parameter, -ErrorAction, "SilentlyContinue," I can have my code continue executing even

if there are errors and no PowerShell error messages will be displayed. This allows me to add my own error handling, which I've done:

```
if ($time) {
    Write-Debug (Get-Msg $time)
   }
else {
    Write-Warning "Failed to get WMI information from $computer"
    Write-Warning $Error[0]
```

Because you might want to keep track of computers that you fail to connect with, I use a global variable, $GetUptimeFailed, to hold computer names.

```
#store computers that fail in a global variable,
    #$Global:GetUptimeFailed. If the variable already
    #exists with members then do nothing, otherwise
    #create the variable.
    if ($Global:GetUptimeFailed.Count -lt 1) {
        $Global:GetUptimeFailed=@()
    }
```

I'm bending my practice to not reference out-of-scope variables, but because the function would be called repeatedly when processing a list of computers, I wanted a way to store names of all the computers that failed. This allows me to display a message to see my failures:

```
PS C:\> Write-Host "Failed to connect to: $Global:GetUptimeFailed" -foregroundcolor "RED"
-BackgroundColor "Yellow"
```

I can also dump the variable to a text file for later processing:

```
PS C:\> $getuptimefailed | out-file failed.txt
```

You'll appreciate this when you are processing a list of servers.

Now, what about the debug stuff? I prefer to add trace messages as I'm developing a script or function using Write-Debug. I control whether the debug message is displayed by setting $DebugPreference. If I change it to "Continue," all my messages will be displayed. I also added a little something extra to my debug messages: I created a function to return a string that includes information such as your computername, username, a date-time stamp, and a user-defined message string.

```
Function Get-Msg {
        Param([string]$text)

        # build a text string of information that can be passed
        # to different pipelines like Debug

        #here are some variables you might want to work with
        $computer=$env:computername
```

```
$domain=$env:userdomain
        $user=$env:username
        $timestamp=(Get-Date -Format g).toString()

        [string]$msg="[$computer - $domain\$user] $timestamp $text"

        write $msg

    }
```

I use a Write-Debug command that calls this function passing the message string:

```
   Write-Debug (Get-Msg "Connecting to $computer")
```

Listing 2 shows what you can expect when debugging output is enabled.

```
PS C:\> "OffLineServer","localhost" | foreach-object {Get-Uptime $_} | Select
Computer,LastBoot
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Connecting to OffLineServer
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Executing Select LastBootUpTime from Win32_
OperatingSystem
WARNING: Failed to get WMI information from OffLineServer
WARNING: The RPC server is unavailable. (Exception from HRESULT: 0x800706BA)
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM $time is empty
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Adding OffLineServer to
$Global:GetUptimeFailed
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM failed computers to date:
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM GODOT GODOT OFFLINESERVER OFFLINESERVER
OFFLINESERVER
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Exiting function
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Connecting to localhost
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Executing Select LastBootUpTime from Win32_
OperatingSystem
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Win32_OperatingSystem=@
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM LastBootUpTime=Win32_OperatingSystem=@.
LastBootuptime
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM $t=02/18/2008 08:09:38
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM $uptime = 02:46:39.6833640
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Creating new object $obj
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Adding 0d 2h 46m 39s
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Writing $obj
```

```
Computer                                        LastBoot
--------                                        --------
LOCALHOST                                       2/18/2008 8:09:38 AM
DEBUG: [MYPC - MYPC\Jeff] 2/18/2008 10:56 AM Exiting function


PS C:\>
```

Listing 2

Again, to control debug output, all I need to do is change $DebugPreference. If there are any issues in connecting to a remote system, you should receive an error message such as this:

```
PS C:\> get-uptime file01
WARNING: Failed to get WMI information from file01
WARNING: Access is denied. (Exception from HRESULT: 0x80070005 (E_ACCESSDENIED))
PS C:\>
```

The benefit is that if you are processing a list of computers, the function will continue but you'll still see the problems.

The function expects computer name and a PSCredential parameters. If you don't specify a computer name, the function defaults to 'localhost.' The function defines two script blocks that essentially execute the same WMI query except one uses the –credential parameter:

```
#define repeated script blocks
    $WMI={Get-WmiObject -query $query -computername $computer -ea silentlycontinue}
    $WMI_Cred={Get-WmiObject -query $query -computername $computer -Credential
$credential -ea silentlycontinue}
```

The appropriate script block is executed depending on whether a PSCredential was passed:

```
if ($credential) {
    Write-Debug (Get-Msg "Connecting with alternate credential")
    $time=&$WMI_Cred
    }
else {
    $time=&$WMI
    }
```

The result of the Get-Wmiobject command is stored in $time. If $time has a value, nothing really happens, but if it doesn't exist, I can assume there was some error such as access denied or RPC server unavailable. The function writes a warning message to the console and exits.

```
if ($time) {
    Write-Debug (Get-Msg $time)
    }
```

```
else {
    Write-Warning “Failed to get WMI information from $computer”
    Write-Warning $Error[0]
    Write-Debug (Get-Msg “`$time is empty”)
    Write-Debug (Get-Msg “Adding $computer to `$Global:GetUptimeFailed”)
    $Global:GetUptimeFailed+=$computer.toUpper()
    Write-Debug (Get-Msg “failed computers to date:”)
    Write-Debug (Get-Msg $Global:GetUptimeFailed)
    Write-Debug (Get-Msg “Exiting function”)

     return
    }
```

If $time has a value, the function proceeds and converts the LastUpTime property to a more meaningful value using the ConvertToDateTime(0) method:

```
   [Datetime]$t=$time.ConvertToDateTime($time.Lastbootuptime)
```

I also create a TimeSpan object that starts with the $t and ends with the current date and time:

```
[TimeSpan]$uptime=New-TimeSpan $t $(get-date)
```

I’ll show you how this is used in just a moment.
The function’s output is a custom object with properties such as Computer and LastBoot.

```
    $obj = New-Object system.Object

    $obj | Add-Member -MemberType NoteProperty -Name “Computer” -Value $computer.
toUpper()
    $obj | Add-Member -MemberType NoteProperty -Name “LastBoot” -Value $t
    $obj | Add-Member -MemberType NoteProperty -Name “Uptime” -value “$($uptime.days)d
$($uptime.hours)h $($uptime.minutes)m $($uptime.seconds)s”
    $obj | Add-Member -MemberType NoteProperty -Name “Days” -Value $uptime.Days
    $obj | Add-Member -MemberType NoteProperty -Name “Hours” -Value $uptime.Hours
    $obj | Add-Member -MemberType NoteProperty -Name “Minutes” -Value $uptime.minutes
    $obj | Add-Member -MemberType NoteProperty -Name “Seconds” -Value $uptime.seconds
```

As you can see, I defined a LastBoot property that takes the value of $t as its value. I’m also using $uptime to define a few other custom properties.

Here’s a basic example of the Get-Uptime function in action:

```
PS C:\> get-uptime


Computer : LOCALHOST
LastBoot : 2/13/2008 7:28:58 AM
Uptime   : 2d 6h 11m 12s
Days     : 2
Hours    : 6
Minutes  : 11
Seconds  : 12
```

See the custom object output? Here's another example where I'm using a previously defined PSCredential stored as $cred:

```
PS C:\> (get-uptime dc01 $cred).lastboot

Monday, January 14, 2008 11:27:44 AM


PS C:\>
```

The other reason I wrote the function to output a custom object is so that it can participate in the pipeline. For example, here's how I processed a list of servers, sorting the output on the LastBoot property and displaying a table with only the computer, lastboot, and uptime properties:

```
PS C:\> $cred=get-credential mycompany\administrator
PS C:\> get-content c:\servers.txt | foreach-object `
>> {get-uptime $_ -cred $cred} | `
>> Sort LastBoot | format-table Computer,LastBoot,Uptime -auto
>>

Computer       LastBoot             Uptime
--------       --------             ------
EXCH01         2/4/2008 3:21:45 AM  11d 10h 16m 16s
EXCH07         2/9/2008 5:21:40 PM  5d 20h 16m 23s
MYCOMPANY-DC01 2/11/2008 8:41:27 PM 3d 16h 56m 32s
```

You might also use the function to create an HTML server uptime report. Or what about creating a CSV report of server uptimes, or writing the information to a database? Custom objects give you plenty of options and opportunities.

I hope you find ways to put this function to work and will let me know how it works out for you.

Jeffery Hicks, MCSE, MCSA, MCT, and Microsoft PowerShell MVP, is a Scripting Guru for SAPIEN Technologies. Jeff is a 16-year IT veteran. He has co-authored and authored several books, courseware, and training videos on administrative scripting and automation. His latest book is WSH and VBScript Core: TFM (SAPIEN Press 2007). You can contact him at jhicks@sapien.com.

Anti-Spam Features of Exchange 2007 Edge and Hub Transport Servers

by J. Peter Bruzzese

Spam…well…it *sucks*. I know, we are pulling our tech talk down into the gutter, and yes we should probably come up with a more appropriate term here utilizing terminology like "spam hinders legitimate communication and productivity while overflowing the storage limits of our mail servers." But, result definitions aside, the feelings we Exchange administrators get when fighting the good fight against spam can be summed up in that one simple word. At times we might want to give up, but like the poet Dylan Thomas encouraged "do not go gentle into that good night…rage, rage against the dying of the light." His subject was death; ours is spam—the encouragement is the same.

Where Are Your Anti-Spam Tools?

So, how can we make good use of Exchange 2007's anti-spam tools? Well, for starters, you need to locate them. On an Edge Transport Server, this proves to be simple. The anti-spam agents are installed by default, so you merely have to select the Edge Transport settings, select the server on which you want to make changes, and you will automatically see an Anti-spam tab (as shown in Figure 1).

With Hub Transport servers, this feature is disabled by default and so you need to first run the antispamagents.ps1 script from the command prompt (first navigating to the folder with Exchange scripts in order to run this script). Or you can open the Exchange Management Shell and type

```
Install-AntispamAgents.ps1
```

In either case, you might need to restart the "Microsoft Exchange Transport" service.

The reason the agents are not installed by default on the Hub Transport server is because the idea is that you should use the Edge Transport server on your perimeter network to protect your environment from spam—and this is the ideal. However, if you do not plan to use Edge Transport servers in your environment (in the event it is a small environment or one that is low on funds for additional hardware/software choices like these), you can enable them on the Hub Transport

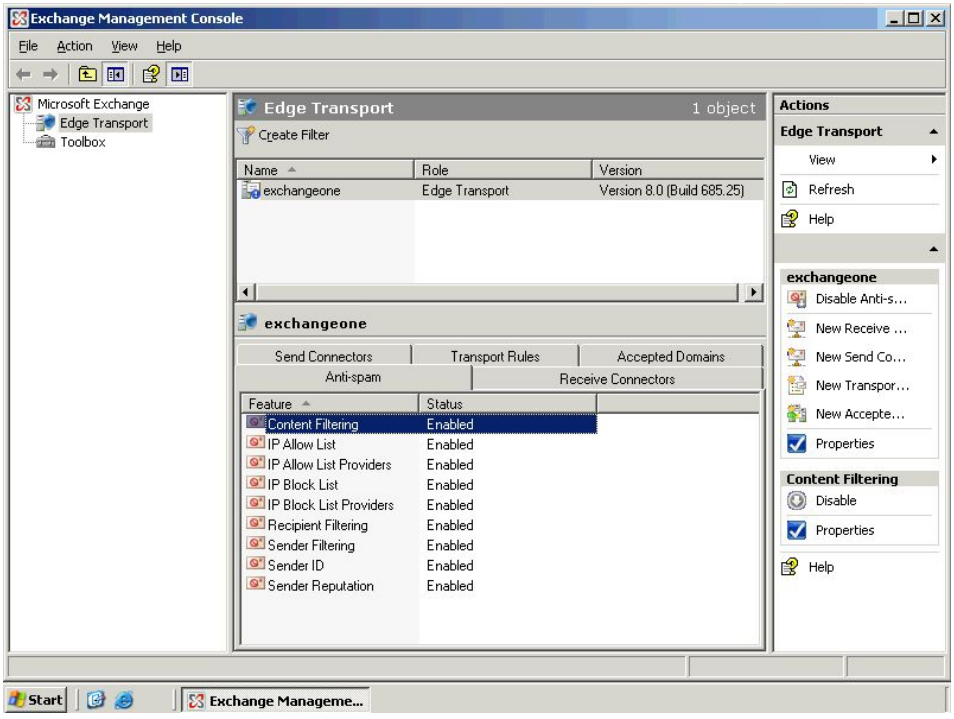

Figure 1: The Anti-Spam features that are automatically built-in to Exchange Transport servers.

for safe measure. Once installed, however, you can open your Exchange Management Console, expand your Organization Configuration—Hub Transport section, and the new tab Anti-spam tab should greet you.

The Nine Anti-Spam Features

You can note from Figure 1 that there are nine interesting settings to choose from that can help protect you if you know what you are doing. These include the following with a simple definition taken from the Microsoft dialogs as to what each of these can do:

- Content Filtering—Filters junk email by using a probability-based algorithm that can learn what is and what isn't spam. Use the Content Filtering feature to filter junk email based on the content of the message. You can set the filtering threshold actions, how content is analyzed, recipient exceptions, and specific words and phrases for the Content Filtering feature to act upon. Note the options available in Figure 2.
- IP Allow List—Specify IP addresses that you are always allowed to connect to and transmit email messages to this server. Accept connections from individual IP addresses or from ranges of IP addresses.
- IP Allow List Providers—Maintain lists of sender domains that can be relied on not to send junk email. Use this feature to determine which IP Allow List provider to use.
- IP Block List—Similar to its 'Allow List' counterpart, you specify IP addresses that you are blocking from connecting to and transmitting email messages to and from.

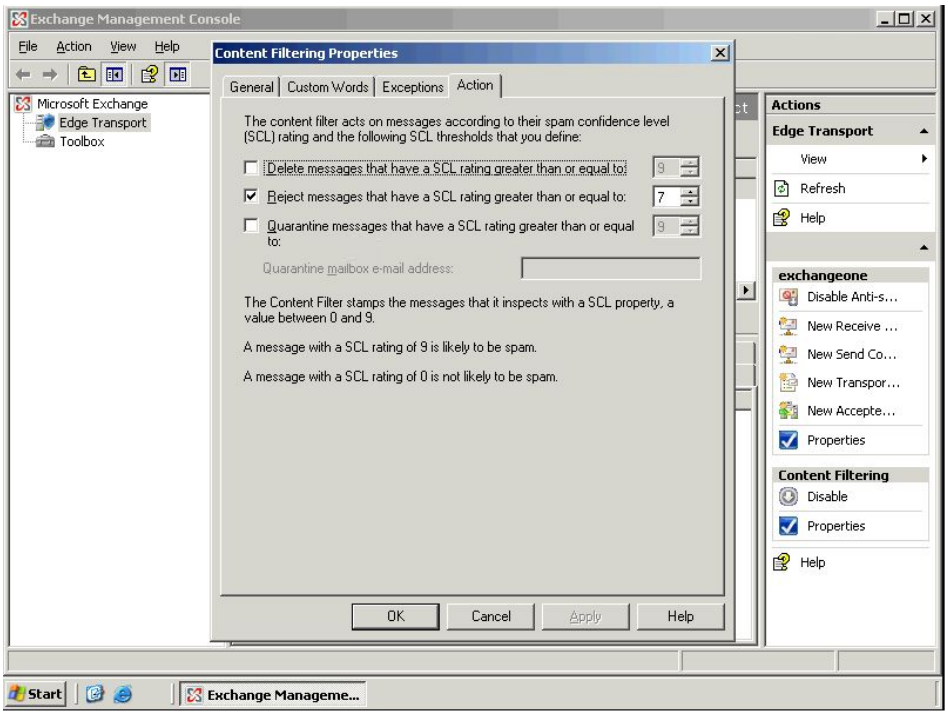

Figure 2: Content Filtering Properties that show you methods of controlling spam based upon spam confidence levels (SCL)

- IP Block List Providers—Maintain lists of sender domains that cannot be relied on that must be blocked from connecting to and transmitting email messages to and from. These block list providers can be invaluable in that they maintain up-to-date lists that you can connect to and utilize.
- Recipient Filtering—A simple feature where you can specify a list of email recipients from which the server will not accept messages. You can block individuals or domains. There is also a check box option on the Blocked Recipient tab that allows you to, with one click, Block messages sent to recipients not listed in the Global Address List.
- Sender Filtering—Specify a list of email senders that you want to block completely. You can block individuals, domains, or whole domain hierarchies. You can also specify how Exchange Transport servers respond when a blocked sender or domain transmits a message. The Sender Filter feature also lets you block messages that do not specify who sent a message (so no more messages with blank senders).
- Sender ID—Intended to combat email spoofing and to provide enhanced protection against phishing schemes. Use Sender ID to examine a sender's purported responsible address (PRA). If the check fails, you can determine whether you want to reject or delete the message or send it along with a stamped message of Sender ID results.
- Sender Reputation—Collects information about recent email messages received; if a sender appears to be the source of junk email, the address is added to a list. There is some flexibility as to the length of time a sender can be blocked and you can also enable/disable open proxy testing.

For a really great step-by-step through the anti-spam features, check out the Exchange Genie blog spot at http://exchange-genie.blogspot.com/2007/12/exchange-2007-anti-spam.html.

As for keeping the definitions up to date for your Edge or Hub Transport servers (if you use the built-in anti-spam features; obviously, a third-party solution will have its own method for pushing updates), you can keep the definitions up to date through Windows Update. Microsoft assists in the process, ensuring you have the latest.

Conclusion

Microsoft has provided these anti-spam features directly within their messaging server to assist in the fight against spam. By utilizing the tools you can slowly begin to weed it out before it ever reaches beyond the transport servers into the mailbox servers. Continue to fight the good fight and rage against the dying of the light!

J. Peter Bruzzese is an MCSE (NT,2K,2K3)/MCT, and MCITP: Enterprise Messaging Administrator. His expertise is in messaging through Exchange and Outlook. J.P.B. is the Series Instructor for Exchange 2007 for CBT Nuggets. His latest book is Tricks of the Vista Masters. *He is co-founder of ClipTraining.com, a provider of short, educational screencasts on Exchange, Windows Server, Vista and Office 2007. You can reach Peter at jpb@cliptraining.com.*

Copyright Statement

© 2008 Realtime Publishers, all rights reserved. This eJournal contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the "Materials") and this work and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers or its sponsors. In no event shall Realtime Publishers or its sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com. ◆