

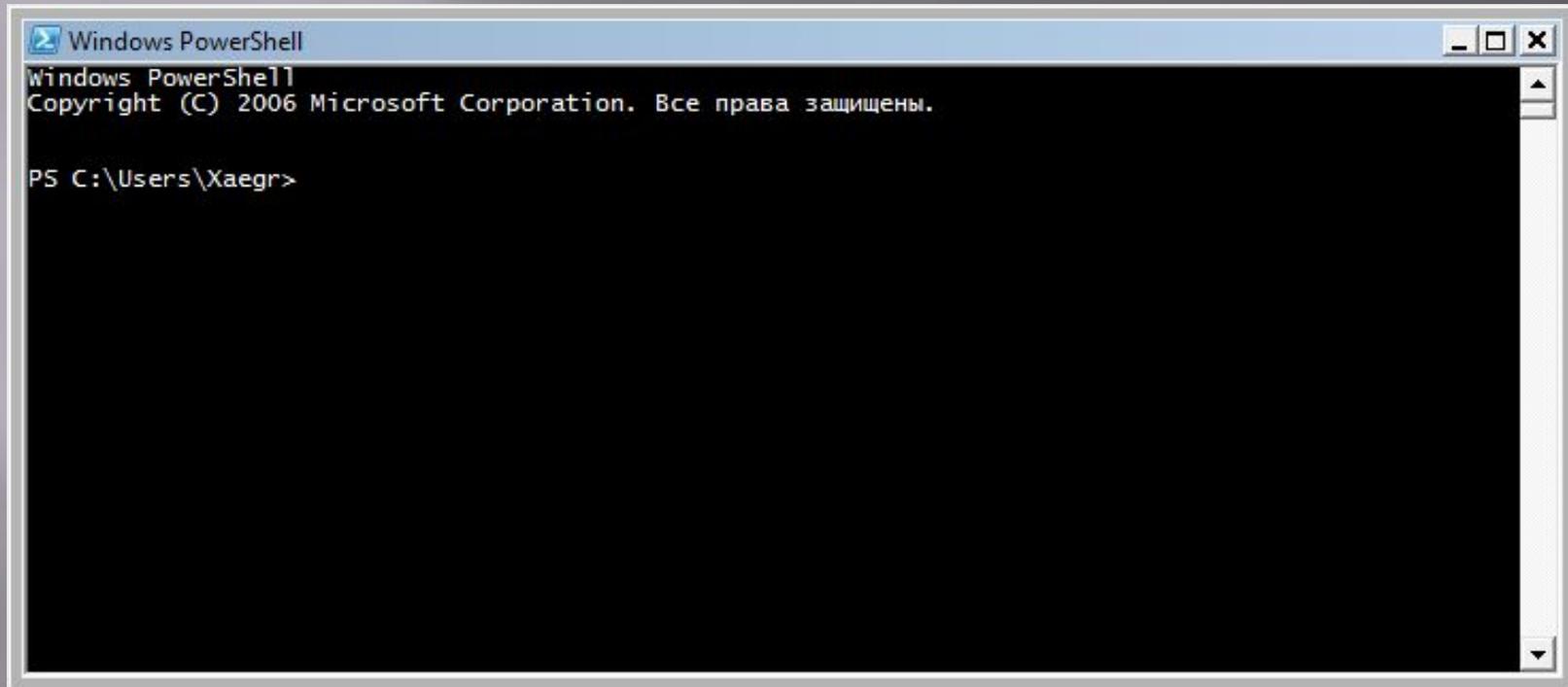
WINDOWS POWERSHELL

В практических примерах

Василий Гусев
хаegr@yandex.ru

Бешков Андрей
abeshkov@microsoft.com

PowerShell – что за зверь такой?



PowerShell – что за зверь такой?

- ▣ Новый интерпретатор командной строки и системный скриптовый язык
- ▣ Долгожданный полноценный «shell» с возможностями не хуже Unix аналогов
- ▣ Объектно ориентирован
- ▣ Работает с унаследованными скриптами VBS, WSH и утилитами командной строки
- ▣ 130 командлетов в стандартной поставке

PowerShell – работает под управлением

- ▣ Windows XP
- ▣ Windows Vista
- ▣ Windows Server 2003
- ▣ Windows Server 2008

PowerShell – перспективы

- Входит в поставку Windows 2008 Server по умолчанию
- Большинство административных интерфейсов превратятся в обертку над Powershell
- Будет встроен во все новые серверные продукты от Microsoft
- Будет использоваться в проектах VMWare и Citrix

Плюсы PowerShell с точки зрения системного администратора

- Ускоряет автоматизирование типичных задач системного администратора
- Прост и интуитивно понятен
- Единый интерфейс к множеству рычагов управления, позволяющий легко связывать их воедино
 - Привычные утилиты командной строки
 - WMI, ADSI, COM
 - Новый рычаг - .Net
 - И многое другое...

Плюсы PowerShell с точки зрения системного администратора

- ▣ Один язык для интерактивной работы, разработки скриптов и их отладки
- ▣ Прост в изучении
 - Руководство пользователя и встроенная справка на русском языке
 - Доступно множество полезных книг
 - Большинство элементов языка вам уже знакомы
 - Новые вещи изучаются интерактивно
 - Знания полученные во время изучения одного компонента легко применимы к другим
 - Время потраченное на обучение не пропадет зря
- ▣ **Создан специально для системных администраторов.**
- ▣ **Доступно множество дополнительных компонентов от сторонних разработчиков**

Недостатки PowerShell

- ❑ Пока малопригоден для logon/startup скриптов
- ❑ Скорость выполнения не высока
- ❑ Требуется инсталляция*
- ❑ Не работает на Windows 2000
- ❑ Пока недоступен в Windows 2008 Server Core

PowerGUI

- GUI хост для PowerShell
- Позволяет работать с PowerShell не зная его
- Легко расширяемый с помощью несложных скриптов
- Превосходный редактор с подсветкой и автоматическим завершением кода и встроенным отладчиком
- Возможность выполнить действие с помощью графического интерфейса, а затем посмотреть соответствующий код PowerShell
- Доступна русификация
- <http://powergui.org/>

PowerGUI

The screenshot shows a remote desktop session titled "Demo1 - 192.168.1.80 - Remote Desktop". The main window is "IsaToolbox.ps1 - PowerGUI Script Editor". The script editor has a menu bar (File, Edit, View, Debug, Tools, Help) and a toolbar with icons for file operations and execution. A "Script Parameters:" field contains the placeholder text "<Input script parameters here>".

```
1 Function Get-IsaArray ([string] $Name="*")
2 {
3     $Root = New-Object -ComObject "FPC.Root"
4     if( $root.Arrays.Count -gt 0)
5     {
6         $root.Arrays|?($_ -like $name)
7     }
8     else
9     {
10        if ($name -eq "*")
11        {
12            $name = Read-Host -
13        }
14        $root.Arrays.Connect
15    }
16 }
17
18 function Get-IsaDomainSet (
19 {
20     if (!$ISA) {
21         Write-Verbose "Подключаемся к ISA серверу"
22         $ISA = Get-IsaArray
23     }
24
25     Write-Verbose "Возвращаем содержимое набора доменных имен"
26     $ISA.RuleElements.DomainNameSets.Item($DomainSet)
27 }
```

A tooltip for the `-AsSecureString` parameter is displayed over the script. The tooltip text is:

AsSecureString <System.Management.Automation.SwitchParameter>
If set to true, the input will be echoed as star characters (*). The output will then be a Securestring object.
Required: False. Position: named. Accept pipeline input: False. Accept wildcard characters: false

The Windows taskbar at the bottom shows the Start button, the active window title "* IsaToolbox.ps1 - Po...", and the system tray with icons for network, volume, and power, along with the time "23:38". The status bar at the bottom of the application window shows "Ready" on the left and "Ln 12 | Col 32 | Ch 23 www.powergui.org" on the right.

PowerGUI

Demo1 - 192.168.1.80 - Remote Desktop

PowerGUI

File Tools Help

PowerGUI [BETA]

- Active Directory
 - Users
 - Groups
 - Computers
 - OUs
 - Browse Active Directo
 - Configuration
 - test
 - test
 - Schema
- Network
- Local System
 - Processes
 - Services
 - Event Logs
 - Network Configuratio
 - Drives
 - WMI Browser

Processes

Filters

Property	Operator	Value
Path	Like	c:\windows*
	Equal	
	GreaterOrEqual	
	LessOrEqual	
	NotEqual	
	Greater	
	Less	
	Like	
	NotLike	

Apply Clear Save As...

ProcessName	Handles	NPM(K)	PM(K)	WS
ctfmon		83	3	544
dfssvc		125	6	1896
dns		186	17	7752
dsamain		319	43	14008
explorer		371	11	5740
ismserv		122	6	1836
logon.scr		22	1	344
lsass		955	103	20656

Links

Add new item...

Actions

- Stop
- Set Priority Class
- Set Processor Affinity
- Set Max Working Size
- Set Min Working Size

Add new item...

Actions: Common

- Report as XML
- Report as CSV
- Report as HTML
- Copy to Clipboard

Add new item...

43 objects

Start PowerGUI

EN 23:41

Quest Software AD Cmdlets

- ▣ Инструменты для облегчения работы с Active Directory из командной строки.

```
$u = Get-QADUser dsotnikov
    $u.TsProfilePath = 'c:\profile'
    $u.CommitChanges()

Import-Csv users.csv | New-QADUser
    -ParentContainer mydomain.local/test
```

- ▣ Бесплатен, прекрасная поддержка.
- ▣ <http://www.quest.com/activeroles-server/arms.aspx>

Mo\W's PowerTab

- Дополняет:
 - Командлеты и их параметры
 - Классы WMI
 - Классы, методы, конструкторы, перечисления .Net
 - Свойства объектов
 - Каталоги, файлы, ключи реестра(и другие объекты текущего PSDrive)
 - Имена исполняемых файлов из \$env:path
 - Имена скриптов .PS1 и их параметры
 - Имена компьютеров и общих папок
 - Преобразовывает псевдонимы в команды
 - Имена и параметры пользовательских функций
 - Имена переменных
 - Настраиваемые пользователем элементы
 - Многое другое...

Mo\W's PowerTab

- ▣ Список вариантов с помощью псевдографики:
 - позволяет уточнять запрос после вызова меню донабирая текст вручную или курсорными клавишами.
- ▣ База настроек в XML файле, для сохранения пользовательских параметров и оптимизации.
- ▣ Цветовые темы
- ▣ Бесплатен
- ▣ Написан на PowerShell
- ▣ Скачивать тут -
<http://thepowershellguy.com/blogs/posh/pages/powertab.aspx>
- ▣ Скринкаст с демонстрацией установки и использования –
<http://хаegr.wordpress.com/2008/02/04/powertab-screencast/>

MoV's PowerTab

Demo1 - 192.168.1.80 - Remote Desktop

Windows PowerShell

```
IPS Desktop]: Set-ExecutionPolicy remotesigned
IPS Desktop]: . .\isatoolbox.ps1
Функции для работы с ISA загружены
IPS Desktop]: $ISA = Get-IsaArray
IPS Desktop]: $ISA.RuleElements -
```

```
$ISA.RuleElements.
$ISA.RuleElements.PSBase
$ISA.RuleElements.CancelWaitForChanges<
$ISA.RuleElements.CanImport<
$ISA.RuleElements.Export<
$ISA.RuleElements.ExportToFile<
$ISA.RuleElements.GetDirtyFeatureMask<
$ISA.RuleElements.GetServiceRestartMask<
$ISA.RuleElements.Import<
$ISA.RuleElements.ImportAsNew<
$ISA.RuleElements.ImportFromFile<
$ISA.RuleElements.LoadDocProperties<
$ISA.RuleElements.Refresh<
$ISA.RuleElements.Save<
$ISA.RuleElements.WaitForChanges<
$ISA.RuleElements.AddressRanges
$ISA.RuleElements.AuthenticationSchemes
$ISA.RuleElements.Computers
$ISA.RuleElements.ComputerSets
$ISA.RuleElements.ContentTypeSets
$ISA.RuleElements.DomainNameSets
$ISA.RuleElements.EventDefinitions
$ISA.RuleElements.LdapMatchingPatterns
$ISA.RuleElements.LdapServersSets
$ISA.RuleElements.PersistentName
$ISA.RuleElements.ProtocolDefinitions
$ISA.RuleElements.RadiusServers
$ISA.RuleElements.Schedules
$ISA.RuleElements.ServerFarms
$ISA.RuleElements.Subnets
$ISA.RuleElements.URLSets
$ISA.RuleElements.UserAgentMappings
$ISA.RuleElements.UserSets
$ISA.RuleElements.VendorParametersSets
$ISA.RuleElements.WebListeners
[18] 1-34 [34]
```

Windows Server 2003, Enterprise Edition
Evaluation copy, Build 3790 (Service Pack 2)

Start Windows PowerShell EN 23:43

MoW's PowerTab

```
Demo1 - 192.168.1.80 - Remote Desktop
Windows PowerShell
[PS Desktop]: Set-ExecutionPolicy remoteSigned
[PS Desktop]: . .\isatoolbox.ps1
Функции для работы с ISA загружены
[PS Desktop]: $isa = Get-IsaArray
[PS Desktop]: Import-IsaDomainSet baddomains.txt
Введите имя Domain Set'a: Banned domains
VERBOSE: Получаем объект набора доменных имен
VERBOSE: Добавляем строки из файла
VERBOSE: Сохраняем изменения
[PS Desktop]: Get-IsaDomainSet "Banned domains"
VERBOSE: Возвращаем содержимое набора доменных имен
yahoo.com
rambler.ru
altavista.com
[PS Desktop]: Clear-i_
Clear-i
Clear-IsaDomainSet
Clear-Item
Clear-ItemProperty
[1] 1-3 (3/6)
```

Windows Server 2003, Enterprise Edition
Evaluation copy. Build 3790 (Service Pack 2)

Start Windows PowerShell EN 23:45

PowerShell Community Extensions

- ▣ Новые командлеты и функции:
 - Get/Set/Out-Clipboard
 - *-Bitmap
 - New-Hardlink
 - New-Junction
 - New-Shortcut
 - Get-Hash
 - Ping-Host; Resolve-Host
 - Get/Stop/Disconnect-TerminalSession
 - Write-Zip; Write-BZip; Write-GZip
 - Elevate
 - И еще множество других...
- ▣ PSDrive провайдеры
 - Feed storage
 - Active Directory
- ▣ Скачивать тут -
<http://www.codeplex.com/PowerShellCX>

Get-IsaArray.ps1

Скрипт для подключения к COM объекту ISA

```
param ([string]$Name="*")
$Root = New-Object -comObject "FPC.Root"
if( $root.Arrays.Count -gt 0)
{
    $root.Arrays|?{$_ -like $name}
}
else
{
    if ($name -eq "*")
    {
        $name = read-host "Enter name of the ISA array"
    }
    $root.Arrays.Connect($name)
}
```

Просмотр и изменение портов SSL соединений

Подключаемся к com-объекту

```
$isa = .\Get-IsaArray.ps1
```

Смотрим какие диапазоны портов уже разрешены

```
$isa.ArrayPolicy.WebProxy.TunnelPortRanges
```

Добавляем диапазон портов

```
$isa.ArrayPolicy.WebProxy.TunnelPortRanges.AddRange(  
    "SSL 1234", 1234, 1234)
```

Удаляем диапазон

```
$isa.ArrayPolicy.WebProxy.TunnelPortRanges.Remove("SSL  
1234")
```

Применяем изменения

```
$isa.ApplyChanges()
```

Экспорт настроек ISA

Подключаемся к com-объекту

```
$isa = .\Get-IsaArray.ps1
```

Записываем текущую дату в нужном формате в переменную

```
$date = Get-Date -Format "yyyy-MM-dd"
```

Экспортируем настройки политик

```
$ISA.ArrayPolicy.ExportToFile(  
    "c:\logs\ISA-Policy-$date.xml",0,"", "Exported at $date")
```

Отчет по правилам ISA

```
# Подключаемся к com-объекту
```

```
$isa = .\Get-IsaArray.ps1
```

```
# Записываем текущую дату в нужном формате в переменную
```

```
$date = Get-Date -Format "yyyy-MM-dd"
```

```
# Получаем политики, выбираем пользовательские, форматируем и сохраняем
```

```
$ISA.ArrayPolicy.PolicyRules | where {-not $_.System} |
```

```
select Order, Name, Enabled,
```

```
@{Name="Type"; Expression={
```

```
    switch($_.type){
```

```
        0 {"Access"};
```

```
        1 {"Publishing"};
```

```
        2 {"Web Publishing"}
```

```
    }
```

```
}},
```

```
@{Name="Action"; Expression={
```

```
    if($_.action -eq 0){"Allow"}else{"Deny"}
```

```
},
```

```
EnableLogging, Description |
```

```
ConvertTo-HTML | Set-Content "c:\reports\Report-$date.html"
```

```
Send-SmtpMail -To "CIO@domain.ru" -Subject "SPAM" -AttachmentPath
```

```
"c:\reports\Report-$date.html"
```

Get-PortState.ps1

Смотрим какие порты открыты снаружи на межсетевом экране.

```
param ([int[]]$ports=@(25,80,443))

$wc = new-object System.Net.WebClient

foreach ($port in $ports)
{
    $url = "http://www.utorrent.com/testport.php?port=$port"
    $ret = $wc.DownloadString($url)
    new-object psobject | select @{N="Port"; E={$port}},
    @{N="State"; E={$ret -match "port $port is open"}}
}
```

.Net

Пусть программисты завидуют

Смотрим процессы на другом компьютере

```
[System.Diagnostics.Process]::GetProcesses("PC01")
```

Отправляем почту

```
$smtp = New-Object System.Net.Mail.SmtpClient
```

```
$smtp.Host = "localhost"
```

```
$smtpclient.Send("from@domain.ru","to@domain.ru","Тема",  
"Текст сообщения")
```

Декодируем URI строку

```
$string = "%D0%9F%D0%BE%D0%B2%D0%B5%D1%80%D0%A8%D0%B5%D0%BB%D0%BB«  
[System.Uri]::UnescapeDataString($string)
```

Получаем произвольное число

```
$rnd = New-Object random
```

```
$rnd.Next(1,100)
```

Функция Out-Notepad

```
function Out-Notepad {
```

```
$file = [System.IO.Path]::GetTempFileName()
```

```
$input | Out-String | Set-Content $file
```

```
notepad.exe $file }
```

WMI

Не хуже других объектов

Список общих папок

```
Get-WmiObject Win32_Share -ComputerName "PC02"
```

Смена метки диска

```
$disk = Get-WmiObject Win32_LogicalDisk |  
  where {$_.deviceId -eq "C:"}  
$disk.VolumeName = "Main"  
$disk.Put()
```

Запускаем процесс на другом компьютере

```
$proc = [wmiClass]"\\PC02\ROOT\CIMV2:win32_process"  
$proc.create("Calc")
```

Список установленного ПО

```
Get-WmiObject Win32_Product | sort vendor | format-table name, vendor, version
```

Планки памяти

```
Get-WmiObject Win32_MemoryDevice |  
  Format-Table DeviceId,  
  @{label="Size"; expression={$_.EndingAddress - $_.startingAddress}}
```

Топ 10 засорителей почтовых ящиков Exchange 2003

```
Get-WmiObject -Class Exchange_Mailbox -Namespace ROOT\MicrosoftExchangev2 |  
  sort size -Descending | select -first 10 |  
  Format-Table *DisplayName, Size, TotalItems
```

Разные полезные мелочи

Быстрая выборка параметров

```
{function:...}={process {$Object=$_;  
  $args[0]|%{$Object.($_)}}}  
Get-Process powershell | ... Id  
dir p* | ... Fullname
```

Быстрый For

```
1..10 | foreach {"Число $_"}  
Get-Content .\computers.txt | foreach { ping.exe $_ -n 1 |  
  Select-String "Ответ" }
```

Регулярные выражения

```
Get-Content ftp.log |  
where {$_ -match "^(\\S+) .+USER (\\S+)"} |  
foreach {"Юзер $($matches[2]) зашел на FTP в  
  $($matches[1])"}
```

СРАВНЕНИЕ POWERSHELL И VBS

VBS: Удаление файлов, созданных до заданной даты

```
set objNamedArgs=Wscript.Arguments.Named

path=objNamedArgs.item("path")
killdate=date() - objNamedArgs.item("killdate")
recur=objNamedArgs.item("recur")
wscript.echo path, killdate, recur
arFiles = Array()
set fso = createobject("scripting.filesystemobject")

'Ничего не удаляем, пока пробегаем по возвращенному набору файлов.
'Набор может быть перемешан.
'Создаём массив файловых объектов, чтобы этого избежать

SelectFiles path, killdate, arFiles, recur

nDeleted = 0
for n = 0 to ubound(arFiles)

on error resume next 'in case of 'in use' files...
arFiles(n).delete true
if err.number = 0 then
nDeleted = nDeleted + 1

end if
on error goto 0
next
```

VBS: Удаление файлов, созданных до заданной даты

```
sub SelectFiles(sPath,vKillDate,arFilesToKill,bIncludeSubFolders)
on error resume next
' добавляем файлы на удаление в массив

set folder = fso.getfolder(sPath)
set files = folder.files
for each file in files
' на всякий случай отслеживаем ошибки доступа к
' свойству Date
,
dtlastmodified = null
on error resume Next
dtlastmodified = file.datelastmodified
on error goto 0
if not isnull(dtlastmodified) Then
if dtlastmodified < vKillDate then
count = ubound(arFilesToKill) + 1
redim preserve arFilesToKill(count)
set arFilesToKill(count) = file
end if
end if
next

if bIncludeSubFolders then
for each fldr in folder.subfolders
SelectFiles fldr.path,vKillDate,arFilesToKill,true
next
end if
end sub
```

Powershell: Удаление файлов, созданных до заданной даты

Получаем текущую дату

```
$DateX = Get-Date
```

“Прибавляем” к ней минус 7 дней

```
$DateX = $DateX.AddDays(-7)
```

Получаем список файлов в каталоге

```
Dir -Recurse |
```

Выбираем те где дата создания меньше \$dateX

```
where {$_.LastWriteTime -lt $DateX} |
```

Будто бы удаляем отобранные файлы

```
Del -Whatif
```

VBS: Информация об учетных записях пользователей на удаленном компьютере

```
On error Resume Next
Const ForReading = 1, ForWriting = 2, ForAppending = 8
'*****
strComputer = "193.125.10.5"
strUser = "Andy_user"
strPassword = "PASSWORD"
strDomain = ""
'*****
Err.Clear
'--- Подключаемся ---
Set objSWbemLocator = CreateObject ("WbemScripting.SWbemLocator")
If (Err.Number <> 0) Then
    WScript.Echo "Error (objSWbemLocator) : " & Err.Number & " " & Err.Description
    WScript.Quit
End If
Err.Clear
Set objSWbemServices = objSWbemLocator.ConnectServer ( _
    strComputer, _
    "root\cimv2", _
    strUser, _
    strPassword, _
    "MS_409", _
    "ntlmomain:" & strDomain)
```

VBS: Информация об учетных записях пользователей на удаленном компьютере

```
If (Err.Number <> 0) Then
    WScript.Echo "Error (objSWbemServices) : " & Err.Number & " " & Err.Description
    WScript.Quit
End If
Err.Clear
Set file_object = CreateObject("Scripting.FileSystemObject")
Set list = file_object.OpenTextFile("./Rezult.log",ForWriting,True)
Set colAcc = objSWbemServices.ExecQuery ("Select * from Win32_UserAccount")
WScript.Echo "Обнаружено " & colAcc.Count & " учетных записей."
For Each ttt in colAcc
    list.Write ttt.Caption & " " & vbCrLf
    list.Write " " & "Caption : " & ttt.Caption & vbCrLf
    list.Write " " & "Name : " & ttt.Name & vbCrLf
    list.Write " " & "Description : " & ttt.Description & vbCrLf
    list.Write " " & "Domain : " & ttt.Domain & vbCrLf
    list.Write " " & "SID : " & ttt.SID & vbCrLf
    list.Write " " & "SIDType : " & ttt.SIDType & vbCrLf
    list.Write " " & "Disabled : " & ttt.Disabled & vbCrLf
    list.Write " " & "Lockout : " & ttt.Lockout & vbCrLf
    list.Write " " & "PasswordChangeable : " & ttt.PasswordChangeable & vbCrLf
    list.Write " " & "PasswordExpires : " & ttt.PasswordExpires & vbCrLf
    list.Write " " & "PasswordRequired : " & ttt.PasswordRequired & vbCrLf
    list.Write " ----- " & vbCrLf
Next
list.Close
```

Powershell: Информация об учетных записях пользователей на удаленном компьютере

```
# Получаем учетные данные с другого компьютера, указав другие учетные данные
$Accounts = Get-WmiObject -ComputerName scenic12 Win32_UserAccount
             -Credential (Get-Credential)
```

```
# Выводим количество записей
```

```
"Обнаружено $($Accounts.Count) учетных записей"
```

```
# Выводим выбранные свойства в виде автоматически выровненной таблицы
```

```
$Accounts | Format-Table Caption, sid*, disabled,
             lockout, password*, description -AutoSize
```

```
# Выводим то же самое в HTML файл
```

```
$Accounts | Select Caption, sid*, disabled,
             lockout, password*, description |
             ConvertTo-Html | Set-Content Accounts.html
```

VBS: Список пользователей не входивших в домен X и более дней

```
var sDomain = ""; // enter your domain here.
var iCutOffDays = 0; // last login cut-off in days.

var TRUE = 1;
var ForReading = 1;
var ForWriting = 2;
var DAYMSECS = 86400000; // number of milliseconds in a day
var DomObj, CollObj, sSubDir;
var iNumUsers = 0;
var CutOff;

var WSHShell = new ActiveXObject("WScript.Shell");
var fs = new ActiveXObject("Scripting.FileSystemObject");

WSHShell.Popup("Starting Last Log Report " );

CutOff = new Date();
CutOff.setTime( CutOff.valueOf() - iCutOffDays*DAYMSECS );

DomObj = GetObject("WinNT://" + sDomain );
CollObj = new Enumerator(DomObj);
```

VBS: Список пользователей не входивших в домен X и более дней

```
for ( ; !CollObj.atEnd(); CollObj.moveNext())
{
var Obj = CollObj.item();
if ( Obj.Class == "User" )
{
iNumUsers++;
try
{
if ( Obj.LastLogin < CutOff.valueOf() )
WriteLOG( Obj.name + " " + Obj.LastLogin );
}
catch( ErrorObj )
{
WriteLOG( Obj.name + " has never logged in" );
}
}
}

WriteLOG("Total number of users is " + iNumUsers );

WSHShell.Popup("Finished");
WScript.Quit();
```

VBS: Список пользователей не входивших в домен X и более дней

```
////////////////////////////////////  
// WriteLOG  
//  
var LogFile = new Object();  
  
function WriteLOG( sLogLine )  
{  
var d = new Date();  
  
if ( LogFile == null )  
{  
LogFile = fs.OpenTextFile( ".\\Logfile.txt" , ForWriting, TRUE );  
LogFile.WriteLine( sLogLine );  
}  
else  
{  
LogFile.WriteLine( sLogLine );  
}  
}
```

Powershell: Список пользователей не входивших в домен X и более дней

```
# Получаем текущую дату
```

```
$DateX = Get-Date
```

```
# “Прибавляем” к ней минус 3 месяца
```

```
$DateX = $DateX.AddMonths(-3)
```

```
# Получаем объекты из AD с указанными свойствами
```

```
Get-QADUser -IncludedProperties Name, LastLogon |
```

```
  # Выбираем с lastlogon меньше $DateX
```

```
  where {$_ .lastlogon -lt $DateX} |
```

```
  # Будто бы удаляем отобранные учетки
```

```
  Disable-QADUser -WhatIf
```

Дополнительные ресурсы:

- ❑ ~~Если ничего не помогает?~~ Сначала прочитайте инструкцию
 - Знакомство с Windows PowerShell (Getting started)
 - Введение в Windows PowerShell (User guide)
 - Вводная статья Андрея Бирюкова
 - <http://www.samag.ru/cgi-bin/go.pl?q=articles;n=11.2007;a=01>
- ❑ Книги
 - PowerShell in Action (Bruce Payette)
 - PowerShell Cookbook (Lee Holmes)
 - PowerShell Course book (Бесплатная, на английском и немецком языках, а возможно скоро и на русском)
 - <https://blogs.technet.com/chitpro-de/archive/2007/05/10/english-version-of-windows-powershell-course-book-available-for-download.aspx>

Дополнительные ресурсы:

- ▣ Официальные сайты
 - <http://www.microsoft.com/powershell>
 - <http://www.microsoft.com/technet/scriptcenter/hubs/msh.mspx>
- ▣ Блоги на русском языке
 - <http://www.itcommunity.ru/blogs/dmitrysotnikov/>
 - <http://хаegr.wordpress.com/>
 - <http://blogs.technet.com/abeshkov/>
- ▣ Блоги на английском языке
 - <http://blogs.msdn.com/powershell/>
 - <http://thepowershellguy.com/>
 - <http://www.leeholmes.com/blog/>
- ▣ Список ресурсов по Powershell
<http://windowspowershell.ru>

Дополнительные ресурсы:

- ▣ Веб-трансляции на русском языке
 - <http://www.microsoft.com/rus/events/detail.aspx?eventid=1032358044>
- ▣ Веб-трансляции на английском языке
 - <http://search.microsoft.com/results.aspx?mkt=en-US&setlang=en-US&q=powershell+webcast>
- ▣ Скринкасты на русском языке
 - <http://хаegr.wordpress.com/category/screencast/>

Вопросы?

Бешков Андрей

Почта: abeshkov@microsoft.com

Live Messenger: [abeshkov@microsoft.com](https://www.live.com/abeshkov)

Блог: <http://blogs.technet.com/abeshkov/>

Василий Гусев

Почта: xaegr@yandex.ru

Live Messenger: [xaegr@yandex.ru](https://www.live.com/xagr)

Блог: <http://xaegr.wordpress.com>