

[О САЙТЕ](#)

Поиск по сайту...

[БЕЗ РУБРИКИ](#)[SECURITY](#)[WINDOWS](#)[НОВОЕ](#)[EXCHANGE/UC](#)[SYSTEM CENTER](#)[VIRTUALIZATION](#)[STORAGES](#)[КАРЬЕРА](#)[SQL](#)[NETWORKS](#)[SHAREPOINT](#)

[Главная](#) → [Windows](#), [Без рубрики](#), [Новое](#) → 7 способов выполнить команду на удалённом компьютере

7 способов выполнить команду на удалённом компьютере



Одна из самых популярных задач у системных администраторов это запуск, какой либо команды на удалённом компьютере, не вставая со своего места. Это может быть необходимо для установки программы или утилиты, изменения каких либо настроек, или для чего угодно ещё. И конечно, редко речь идёт лишь об одном компьютере, чаще команду нужно выполнить на множестве рабочих станций или серверов.

Так как задача эта популярная, то и способов её решения существует множество. Начиная от групповых политик (в которых можно применять для этой цели сценарии входа в систему или автозагрузки), и заканчивая мощными системами управления, вроде System Center Essentials или System Center Configuration Manager. Но я в этой статье хочу рассмотреть методы, которые доступны сразу из командной строки или файлов сценариев, а так же не требуют предварительной установки агентов и прочей суматохи. Впрочем, какие-то предварительные требования конечно есть. Например, у вас должны быть административные полномочия на том компьютере, на котором вы хотите выполнить команду (за исключением сценария с «проксированием», но об этом позже).

PsExec.exe

Один из моих любимых способов для решения этой задачи это утилита командной строки PsExec.exe написанная Марком Руссиновичем, которую вы можете свободно скачать с сайта Windows SysInternals. Ссылку на неё вы можете найти в конце статьи.

[Log in](#)

Последние комментарии

[Justinbat](#) → [Группы рассылки на сервере Exchange 2010](#)

[Loganel](#) → [Обзор видов кэша в SharePoint Server \(Часть 2...](#)

[Brianel](#) → [Группы рассылки на сервере Exchange 2010](#)

[OlgaCob](#) → [Обзор видов кэша в SharePoint Server \(Часть 2...](#)

Метки

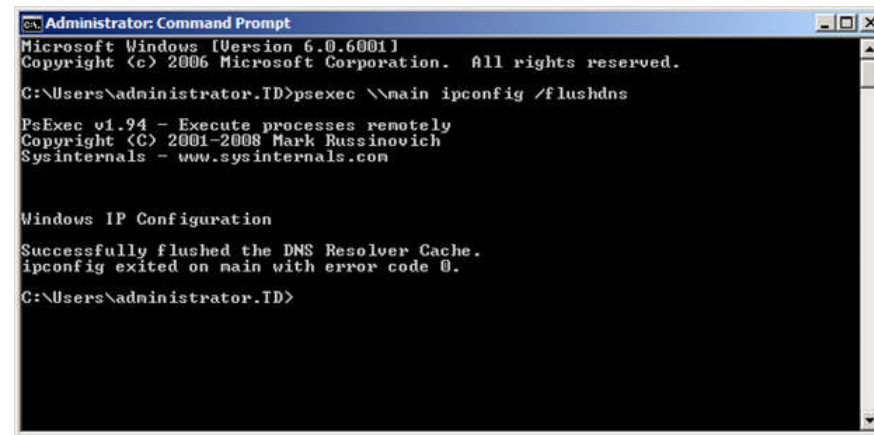
Active Directory AD RMS
Exchange 2007 Exchange
2010 Exchange 2013 Fibre Channel
Group Policy HDD Hyper-V iSCSI
Lync Microsoft.next NetApp
OpsMgr OSD PowerShell RAID Remote
Desktop SCCM SCMDM 2008

Она не требует установки в систему, вы можете просто скопировать её в одну из папок, содержащихся в переменной окружения %path% и вызывать из любой оболочки командной строки: Cmd или PowerShell.

Использовать PsExec очень просто. Например, чтобы выполнить ipconfig /flushdns на компьютере main, достаточно запустить следующую команду:

```
psexec \\main ipconfig /flushdns
```

Команда ipconfig будет запущена на компьютере main под вашими учетными данными. После завершения работы ipconfig весь текстовый вывод будет передан на ваш компьютер, а кроме того будет возвращён код выхода команды (error code). В случае если команда выполнена успешно, он будет равен 0.



```
Administrator: Command Prompt
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

C:\Users\administrator.ID>psexec \\main ipconfig /flushdns

PsExec v1.94 - Execute processes remotely
Copyright (C) 2001-2008 Mark Russinovich
Sysinternals - www.sysinternals.com

Windows IP Configuration
Successfully flushed the DNS Resolver Cache.
ipconfig exited on main with error code 0.

C:\Users\administrator.ID>
```

Разумеется, на этом возможности PsExec не заканчиваются. Вызвав утилиту без параметров, можно посмотреть другие доступные опции. Я обращаю внимание лишь на некоторые из них.

Ключ **-d** говорит PsExec что не нужно дожидаться выполнения команды, а достаточно лишь запустить её, и забыть. В этом случае мы не получим выходных данных от консольной утилиты, но зато сможем не дожидаясь завершения предыдущей команды запускать другие. Это очень полезно, если вам необходимо запустить, например установщик программы на нескольких компьютерах.

По умолчанию PsExec выполняет команды в скрытом режиме, то есть на системе где выполняется команда, не будут выводиться никакие окна или диалоги. Однако есть возможность изменить это поведение, с помощью ключа **-i**. После него можно указать номер сессии, в которой выводиться окна, а можно и не указывать, тогда интерфейс будет отображен в консольной сессии.

Таким образом, чтобы вывести окно с информацией о версии операционной системы

SCOM SCSM Security
SharePoint SQL SQL 2008
R2 Storages Update
Virtualization Visio VMware
Windows 7 Windows
2008 R2 Windows Server
2008 WSUS Конкурсы
Общее Видео Друзья
Пятничная тема СХД
виртуализация мониторинг
мысли файловая система

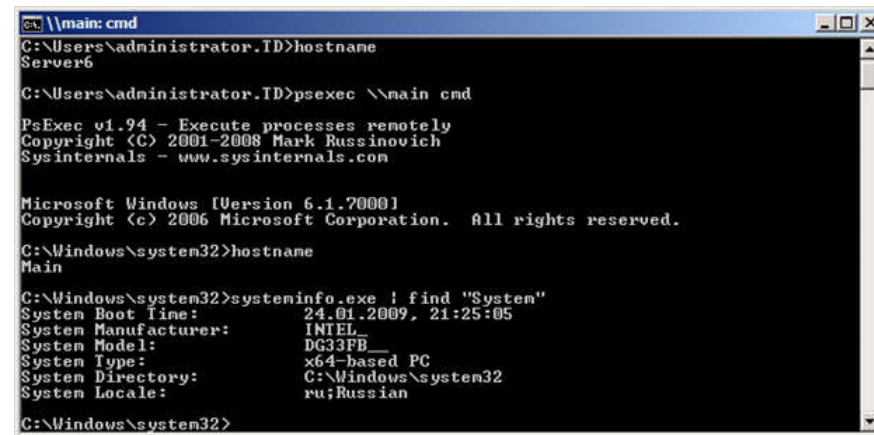
на компьютере main, следует запустить PsExec таким образом:

```
psexec -i \\main winver.exe
```

Если вы хотите выполнить команду сразу на нескольких компьютерах, вам пригодится возможность прочитать их имена из текстового файла списка.

```
psexec @c:\comps.txt systeminfo.exe
```

Ну и одной из самых полезных способностей PsExec является возможность интерактивного перенаправления ввода/вывода между компьютерами, что позволяет нам запустить, например cmd.exe на удалённом сервере, а давать ему команды и получать результаты на локальном компьютере.



```
\\main: cmd
C:\Users\administrator.ID>hostname
Server6

C:\Users\administrator.ID>psexec \\main cmd

PsExec v1.94 - Execute processes remotely
Copyright (C) 2001-2008 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 6.1.7000]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

C:\Windows\system32>hostname
Main

C:\Windows\system32>systeminfo.exe | find "System"
System Boot Time:      24.01.2009, 21:25:05
System Manufacturer:  INTEL
System Model:          DC33FB
System Type:            x64-based PC
System Directory:      C:\Windows\system32
System Locale:          ru;Russian

C:\Windows\system32>
```

Каким образом работает PsExec?

Всё гениальное просто. В ресурсах исполняемого файла PsExec.exe находится другой исполняемый файл – PSEXESVC, который является службой Windows. Перед выполнением команды, PsExec распаковывает этот ресурс на скрытую административную общую папку удалённого компьютера, в файл: \\ИмяКомпьютера\Admin\$\system32\psexesvc.exe. Если вы с помощью ключа -s указали что необходимо скопировать исполняемые файлы на эту систему, они тоже копируются в эту папку.

По завершению подготовительных действий, PsExec устанавливает и запускает службу, используя API функции Windows для управления службами. После того как PSEXESVC запустится, между ним и PsExec создаётся несколько каналов для передачи данных (вводимых команд, результатов, и т.д.). Завершив работу, PsExec останавливает службу, и удаляет её с целевого компьютера.

Windows Management Instrumentation (WMI)

Следующий способ реализации этой популярной задачи, о котором я хочу поведать –

использование Windows Management Instrumentation. WMI присутствует во всех операционных системах Microsoft, начиная с Windows 2000, и даже на Windows 9x его можно установить из отдельного пакета. WMI включён по умолчанию, и не требует дополнительной настройки. Для его использования достаточно административных прав, и разрешенного на брандмауэре протокола DCOM. WMI предоставляет огромные возможности для управления системами, но нас сейчас интересует лишь одна из них.

Для запуска процессов нам потребуется метод Create класса Win32_Process. Использовать его достаточно несложно. В PowerShell это делается следующим образом:

```
$Computer = "main"
$Command = "cmd.exe /c systeminfo.exe > \\server\share%\%computername%.txt"
([wmiclass]"\\$Computer\root\cimv2:Win32_Process").create($Command)
```

Здесь в качестве запускаемого процесса я указал cmd.exe, а уже ему, в качестве аргументов передал нужную команду. Это необходимо в случае если вам нужно использовать переменные окружения удалённого компьютера или встроенные операторы cmd.exe, такие как «>» для перенаправления вывода в файл. Метод Create не дожидается завершения процесса, и не возвращает результатов, но зато сообщает нам его идентификатор – ProcessID.

Если вы используете компьютер, на котором пока не установлен PowerShell, вы можете вызвать этот метод WMI и из сценария на VBScript. Например вот так:

Листинг №1 – Запуск процесса используя WMI (VBScript)

```
Computer = "PC3"
Command = "cmd.exe /c systeminfo.exe > \\server\share%\%computername%.txt"
Set objWMIService = GetObject("winmgmts:\\\" & Computer & "\\root\cimv2:Win32_Process")
Result = objWMIService.Create("calc.exe", Null, Null, intProcessID)
```

Но гораздо проще воспользоваться утилитой командной строки wmic.exe которая предоставляет достаточно удобный интерфейс для работы с WMI и входит в состав операционных систем, начиная с Windows XP. В ней чтобы запустить, например калькулятор на компьютере main достаточно выполнить следующую команду:

```
wmic /node:main process call create calc.exe
```

Разумеется, возможности WMI не ограничиваются только запуском процессов. Если вам интересно дальнейшее изучение этой технологии, я рекомендую ознакомиться со статьями Константина Леонтьева, посвященными WMI, ссылки на которые вы можете найти в конце статьи.

WSH Remote Scripting

Да, как ни странно у Windows Script Host тоже есть возможность запуска сценариев на

других компьютерах. Правда эта функция не получила большой популярности, и скорее всего из-за того что требует слишком много подготовительных мероприятий, а взамен предоставляет совсем немного возможностей. Но я все равно расскажу об этом методе, так как и он может пригодиться.

Итак, для запуска сценария на другом компьютере с помощью WSH нам понадобится сделать следующее:

1. Права администратора на удалённом компьютере. Это само собой разумеется, и требуется почти для всех остальных методов запуска перечисленных в этой статье.
2. Разрешить WSH Remote Scripting создав в системном реестре строковый параметр Remote равный "1" в ключе реестра HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Script Host\Settings
3. Из за ошибки описанной в статье базы знаний Microsoft с номером 311269, на системах с Windows XP может понадобиться выполнить команду wscript -regserver
4. Если на компьютерах используется брандмауэр, то в нём необходимо разрешить обращения к DCOM. Причем сделать это надо не только на управляемом компьютере, но и на том с которого вы хотите запускать сценарий.
5. В системах Windows XP с пакетом обновлений 2 и выше, необходимо изменить параметры безопасности DCOM. Это можно сделать с помощью групповой политики. В узле Computer Configuration \ Windows Settings \ Security Settings \ Local Policies \ Security Options следует установить разрешения следующим образом:

1. DCOM: Machine Access Restrictions in Security Descriptor Definition Language (SDDL) syntax
Выдать группам Anonymous Logon и Everyone разрешения Allow Local и Allow Remote Access
2. DCOM: Machine Launch Restrictions in Security Descriptor Definition Language (SDDL) syntax
Выдать группе Administrators разрешения Allow Local Launch, Allow Remote Launch, Allow Local Activation, Allow Remote Activation
Группе Everyone – Allow Local Launch, Allow Local Activation

Ну и после всех этих процедур, можно попробовать запустить свой сценарий на другом компьютере.

Пример сценария, который использует эту технологию:

Листинг №2 – WSH remote scripting (VBScript)

```
Set objController = CreateObject("WshController")
Set objRemoteScript = objController.CreateScript("C:\test.vbs",
"PC5")WScript.ConnectObject objRemoteScript, "remote_"
objRemoteScript.Execute
Do While objRemoteScript.Status <> 1
```

```
WScript.Sleep 1000
Loop
MsgBox "Script complete"
Sub remote_Error
Dim objError
Set objError = objRemoteScript.Error
WScript.Echo "Error - Line: " & objError.Line & _
", Char: " & objError.Character & vbCrLf & _
"Description: " & objError.Description
WScript.Quit -1
End Sub
```

На второй его строчке, в качестве параметров для функции CreateScript указывается путь к файлу сценария, который будет выполнен на удаленном компьютере и собственно имя этого компьютера.

Более подробную статью об этой технологии можно прочитать в статье **Advanced VBScript for Microsoft Windows Administrators – Chapter 6: Remote Scripting** (см. Ссылки).

Планировщик заданий (Task Scheduler)

Планировщиком заданий можно управлять из командной строки используя две утилиты – at.exe и schtasks.exe. Обе эти утилиты позволяют указать имя удалённого компьютера для создания задания, и, следовательно, позволяют решить нашу задачу. Но подробно мы рассмотрим лишь schtasks.exe, так как она предоставляет гораздо больше возможностей.

Хотя выполнение команд на других компьютерах не является основным предназначением планировщика, тем не менее он позволяет реализовать немало интересных сценариев. Например, с его помощью можно включить установку программного обеспечения в период обеденного перерыва. Или если ваши пользователи обедают в разное время, запуск можно выполнять после определённого периода бездействия компьютера.

```
schtasks /create /s server6.td.local /tn install /tr \\main\data\install.cmd
/sc once /st 13:00 /ru system
```

Важно понимать от имени какой учетной записи будет выполняться задача. В этом примере я указал для параметра /ru значение system, следовательно, для выполнения установки учетной записи компьютера будет необходим доступ на чтение в сетевую папку с дистрибутивом программы.

Еще полезным решением, мне кажется запланировать какое либо действие, на ежедневное выполнение, и удалять задачу лишь при подтверждении его успеха. То

есть вы можете создать простой командный файл, который сначала запускает установщик программы, дожидается его завершения, и проверяет – успешно ли установилась программа. Если это так, то он удаляет задание из планировщика на этом компьютере. Пример такого файла:

Листинг №3 – Установка программы с последующим удалением задания (Windows Batch)

```
msiexec /qn /package \\server\share\subinac1.msi
if exist "c:\program files\Windows Resource Kits\Tools\subinac1.exe" (
subinac1 /tn Install_Subinac1 /f
)
```

WinRM (WS-Management)

WinRM – это реализация открытого стандарта DMTF (Distributed Management Task Force) от Microsoft, которая позволяет управлять системами с помощью веб-служб. Углубляться в устройство технологии я не буду, а лишь кратко опишу, что необходимо для её использования.

Версия WinRM 1 и выше входит в состав операционных систем, начиная с Windows Vista и Windows Server 2008. Для Windows XP и Windows Server 2003 можно установить WinRM в виде отдельного пакета (см. ссылки).

Для того чтобы быстро настроить компьютер для подключений к нему используя стандартные порты и разрешив подключения административным учетным записям, достаточно выполнить команду:

```
winrm quickconfig
```

Чтобы winrm не спрашивал подтверждения, можно добавить к вызову ключ -quiet. Узнать информацию о более тонкой настройке можно посмотреть встроенную справку winrm:

```
winrm help config
```

Если на управляемом компьютере работает веб-сервер, WinRM никак ему не мешает, хоть и использует по умолчанию стандартные порты HTTP. Он будет перехватывать лишь подключения предназначенные специально для него.

Разумеется необязательно выполнять эту команду вручную, на каждом компьютере которым вы хотите управлять. Все необходимые настройки легко сделать с помощью групповых политик. Для этого нужно:

1. Настроить службу WinRM (Windows Remote Management)на автоматический запуск
2. Настроить элемент групповой политики Computer Configuration \ Administrative Templates \ Windows Components \ Windows Remote Management (WinRM) \ WinRM Service \ Allow automatic configuration of listeners. Тут нужно указать диапазоны IP-

адресов с которых разрешаются подключения.

3. Разумеется, еще вам будет необходимо разрешить подключения на соответствующие порты (по умолчанию 80) в брандмауэре Windows.

Независимо от того используется ли порт HTTP (80) или HTTPS (443) трафик передаваемый WinRM шифруется (если конечно вы не отключите эту опцию). Для аутентификации по умолчанию используется протокол Kerberos.

Но хватит о настройках, лучше перейдем непосредственно к использованию. Хотя утилита winrm позволяет настраивать службу WinRM, а так же выполнять например WMI запросы, нам более интересна другая – winrs. Буквы RS тут означают Remote Shell. WinRS работает очень похоже на PsExec хотя и использует технологию WinRM. Имя компьютера задаётся ключом -r, а после него следует команда которую нужно выполнить. Вот несколько примеров:

```
winrs -r:Core ver.exe
```

Так как winrs и так использует cmd.exe в качестве удалённой оболочки, в командах можно легко обращаться к удалённым переменным окружения, или использовать другие встроенные команды cmd.exe:

```
winrs -r:Core "dir c:\temp > c:\temp\list.txt"
```

Как и PsExec, утилита winrs позволяет открыть интерактивный сеанс на удалённом компьютере:

```
winrs -r:main cmd.exe
```

Эта функция аналогична telnet сессии, но использование winrs однозначно лучше telnet и даже PsExec, с точки зрения безопасности. Независимо от того используется ли порт HTTP (80) или HTTPS (443), трафик передаваемый WinRM шифруется (если конечно вы не отключите эту опцию). Для аутентификации по умолчанию используется протокол Kerberos.

Windows PowerShell 2.0 Remoting

Хотя вторая версия Windows PowerShell на момент написания статьи находится еще в состоянии бета тестирования, о её возможностях в области удалённого выполнения команд определённо стоит рассказать уже сейчас. Попробовать его своими руками вы можете либо загрузив предварительную версию (см. ссылки) либо в составе бета-версии Windows 7 или Windows Server 2008 R2.

Инфраструктура PowerShell Remoting основана на WinRM версии 2.0, и поэтому наследует все преимущества этой технологии, такие как шифрование передаваемых данных, и возможность работать по стандартным портам HTTP/HTTPS. Но благодаря богатым возможностям языка Windows PowerShell, и его способностям работы с объектами, мы получаем еще большие возможности. На данный момент пакет

WinRM2.0 тоже находится в состоянии бета-тестирования, и доступен для загрузки только для систем Windows Vista и Windows 2008. В системы Windows 7 и Windows Server 2008R2 он будет встроен изначально, как и PowerShell 2.0.

Обновление: К моменту публикации статьи на ItBand.ru, финальные версии PowerShell 2.0 и WinRM 2.0 доступны уже для всех поддерживаемых платформ. В состав Windows Server 2008R2 и Windows 7 они уже включены как неотъемлемые компоненты системы, а для Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008 все необходимые компоненты можно получить в виде пакета называемого [Windows Management Framework](#).

Перед тем как воспользоваться всеми этими преимуществами, PowerShell Remoting необходимо активизировать, на управляющем, и управляемых компьютерах. Сделать это просто, запустив командлет (команду Windows PowerShell) Enable-PSRemoting. Причем если добавить ключ -Force то никаких подтверждений запрошено не будет. Этот командлет при необходимости вызовет winrs quickconfig, и создаст исключения в брандмауэре Windows, так что никаких дополнительных действий выполнять не нужно.


После этого вы сможете легко выполнять команды на других компьютерах используя командлет Invoke-Command (или его псевдоним icm):

```
Invoke-Command -ComputerName Main -ScriptBlock {netsh interface dump > c:\ipconfig.txt}
```

Разумеется команду можно заранее поместить в переменную, а для параметра -ComputerName указать имена не одного, а сразу нескольких компьютеров. Следующая последовательность позволяет вывести версию файла Explorer.exe сразу с трех компьютеров.

```
$Command = {(get-item c:\Windows\explorer.exe).VersionInfo.FileVersion}
```

```
Invoke-Command -ComputerName Main, Server7, Replica -ScriptBlock $Command
```



```
Administrator: powershell (running as td\administrator)
Windows PowerShell U2
Copyright (C) 2008 Microsoft Corporation. All rights reserved.

PS> $Cmds = <
>> $Procs = Get-Process | where {$_.path -like "c:\windows\*"}
>> $Procs | sort cpu | select -last 3
>> }
>>
PS> $Result = Invoke-Command -ComputerName Main, Server7, Replica -Script $Cmds
PS> $Result | sort cpu -Descending | Format-Table name, cpu, PSComputerName

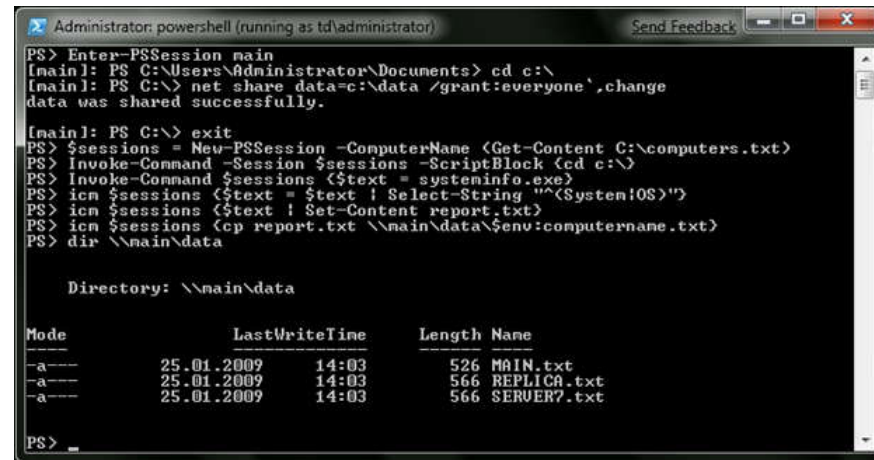
Name                                CPU PSComputerName
-----
explorer                            575,203125 main
suchost                             253,484375 main
vmwp                                 214,09375 main
suchost                              61,0625 replica
lsass                                42,3125 replica
spnsvc                               27,421875 replica
suchost                               6,8125 server7
suchost                               3,640625 server7
spoolsv                              2,453125 server7

PS>
```

Как видно на, можно передавать сразу несколько команд в одном блоке, помещать их результаты выполнения на нескольких компьютерах в переменную, а затем обрабатывать на рабочей станции используя возможности Windows PowerShell по работе с объектами.

Впрочем возможности PowerShell Remoting на этом только начинаются. С помощью командлета Enter-PSSession вы можете войти в интерактивную сессию Windows PowerShell на удалённом компьютере. Выйти из такого сеанса можно использовав командлет Exit-PSSession, или просто exit.

Командлет New-PSSession создает сессии на удалённых компьютерах, указатели на которые можно поместить в переменную, а затем передавая её как аргумент для Invoke-Command выполнять команды сразу на нескольких компьютерах, в постоянном окружении. Пример вы можете увидеть на скриншоте, где я выполняю последовательность команд сразу на нескольких компьютерах из списка c:\computers.txt.



```
Administrator: powershell (running as td\administrator)
PS> Enter-PSSession main
[main]: PS C:\Users\Administrator\Documents> cd c:\
[main]: PS C:\> net share data=c:\data /grant:everyone',change
data was shared successfully.
[main]: PS C:\> exit
PS> $sessions = New-PSSession -ComputerName (Get-Content C:\computers.txt)
PS> Invoke-Command -Session $sessions -ScriptBlock {cd c:\}
PS> Invoke-Command $sessions {$text = systeminfo.exe}
PS> icm $sessions {$text = $text | Select-String "(SystemIOS)" }
PS> icm $sessions {$text | Set-Content report.txt}
PS> icm $sessions {cp report.txt \\main\data\$env:computername.txt}
PS> dir \\main\data

Directory: \\main\data

Mode                LastWriteTime         Length Name
----                -
-a----           25.01.2009   14:03             526 MAIN.txt
-a----           25.01.2009   14:03             566 REPLICA.txt
-a----           25.01.2009   14:03             566 SERVER7.txt

PS> _
```

Проксирование

Этот метод отличается от всех вышеперечисленных, и служит совсем для других задач, но не менее актуален. Когда делегирование полномочий невозможно, или предоставляет слишком большие возможности, он позволяет разрешить обычному пользователю выполнять некую команду, требующую административных привилегий, никаким образом не выдавая дополнительных полномочий и не подставляя под угрозу пароль администратора.

Чаще всего такие проблемы люди решают с помощью утилит вроде srau.exe (см. ссылки) которые создают файл с зашифрованным паролем административной учетной записи, позволяющий запускать определённую программу. Проблема, однако, в том,

что хоть пароль и зашифрован, перед запуском программы утилите придётся его расшифровать. А соответственно пользователь может использовать утилиту повторяющую алгоритм расшифровки пароля, и узнать его, чтобы затем использовать для запуска других программ или получения дополнительных привилегий. Практически это конечно достаточно сложно для обычных пользователей, не обладающих специальными знаниями, но, тем не менее, вполне возможно. Еще раз уточню, это не беда конкретной утилиты, а проблема такого подхода вообще.

Еще может показаться, что для решения задачи подойдет параметр /savecred утилиты runas. Но тут есть даже две проблемы. Во-первых, как и вышеописанном случае, пароль сохраняется на компьютере пользователя, а, следовательно, может быть расшифрован, хотя в случае с runas для этого и понадобятся права локального администратора. Во-вторых, runas сохраняет учетные данные, не связывая их с конкретной командой, а, следовательно, пользователь сможет запустить с завышенными правами не только ту команду, доступ к которой вы хотели ему предоставить, но и любую другую.

Чтобы избежать этих проблем, но, тем не менее, разрешить выполнение конкретной команды, можно использовать методику, которая называется "проксированием".

Работает она следующим образом. На компьютере постоянно работает сценарий с высокими привилегиями. Например, в нашем случае он будет запущен из-под учетной записи, обладающей правами администратора на файловом сервере. По сигналу пользователя он будет выполнять одну, заранее определённую команду. В этом примере – закрывать все файлы, открытые по сети.

Для организации этой системы мы поместим на сервере, например в папке c:\scripts\ командные файлы Server.cmd и Action.cmd .

Листинг №4 – Server.cmd (Windows Batch)

```
set trigger=c:\commandShare\trigger.txt
set action=c:\scripts\action.cmd
set log=c:\scripts\log.txt
:start
if exist %trigger% start %action% & echo %time% %date%>>%log% & del %trigger%
sleep.exe 5
goto start
```

Листинг №5 – Action.cmd (Windows Batch)

```
for /f "skip=4 tokens=1" %a in ('net files') do net files %a /close
exit
```

Server.cmd будет ждать знака от пользователя (создание файла в определенном месте), и получив его, запускать файл с командами – Action.cmd. Разумеется, в эту папку пользователи не должны иметь никакого доступа. Автоматический запуск Server.cmd

при запуске компьютера можно организовать, просто создав соответствующую задачу в планировщике:

```
schtasks /create /ru domain\administrator /rp /sc onstart /tn ProxyScript /tr  
c:\scripts\server.cmd
```

После параметра /ru указывается учетная запись, под которой будет выполняться сценарий (в нашем случае она обладает правами администратора на сервере), так как после параметра /rp пароль не указан – он будет запрошен при создании задачи. Параметр /sc позволяет указать момент запуска сценария, в нашем случае – при включении компьютера. Ну а /tn и /tr позволяют указать имя задачи, и исполняемый файл.

Теперь, для того чтобы пользователь мог подать сценарию сигнал, мы создадим папку c:\commandShare и сделаем её доступной по сети. Доступ на запись в эту папку должен быть только у тех пользователей, которые будут запускать команду.

После этого достаточно будет поместить пользователю на рабочий стол файл Run.cmd.

Листинг №6 – Run.cmd (Windows Batch)

```
echo test > \\server\commandShare\trigger.txt
```

При его выполнении, от имени пользователя, будет создаваться файл \\server\commandShare\trigger.txt. Сценарий Server.cmd, заметив его, запустит на выполнение со своими привилегиями файл Action.cmd, добавит запись в файл c:\scripts\log.txt о текущем времени, а затем удалит trigger.txt чтобы не выполнять команду снова до следующего сигнала пользователя.

В сценарии Server.cmd используется утилита Sleep.exe, позволяющая сделать паузу в выполнении сценария на заданный в секундах промежуток времени. Она не входит в состав операционной системы, но её можно взять из набора Resource Kit Tools (см. ссылки) и просто скопировать на любой компьютер.

Ссылки

[PsExec.exe](#)

[Windows SysInternals](#)

http://www.script-coding.info/WMI_Processes.html

[BUG: You receive an "ActiveX component can't create object" error message when you Use Windows Script Host to execute remote script](#)

[Advanced VBScript for Microsoft Windows Administrators – Chapter 6: Remote Scripting](#)

[Вы всё ещё не используете WMI? Часть 1](#)

[Вы всё ещё не используете WMI? Часть 2](#)

[Узнай секреты WMI: события и провайдеры](#)

[Узнай секреты WMI: события и провайдеры](#)

[WS-Management v1.1 для Windows XP и Windows Server 2003](#)

[CPAU.exe](#)

[Windows PowerShell 1.0](#)

[Windows PowerShell 2.0 Community Technology Preview 3 –](#)

[WinRM 2.0 Community Technology Preview 3](#)

[Windows Server 2003 Resource Kit Tools](#)

Ранее статья была опубликована в журнале Windows IT Pro RE в №4 за 2009 год.

Василий Гусев

<http://xaegr.wordpress.com/>

Рубрика: [Windows](#), [Без рубрики](#), [Новое](#) Автор: Василий Гусев Дата: Monday 16 Nov 2009

Комментарии



naPmu3aH

17 November,
2009

Спасибо, интересно. Некоторые способы не знал (а про powershell 2 remoting слышал в Вашей с Сотниковым презентации на Платформе)
Кстати, в тексте про psexec попадаете powershell 😊



Василий
Гусев

17 November,
2009

Угу, вот что значит быть маньяком 😊 Завтра попробую поправить.
Плюс еще надо исправить кавычки на нормальные. Если еще что-то заметите – говорите 😊



MaxxxR

17 November,
2009

Для запуска приложения с повышенными привелегиями предлагаю использовать Autolt предварительно сохранив пароль в программе (скрипте). На сколько известно Autolt v3 не подвержен декомпиляции. Хотя вытащить сохраненный пароль – тока дело времени.



Василий
Гусев

17 November,
2009

МахххR: Предложите еще поместить пароль в файл password.txt и переименовать в password.exe, авось никто не догадается. Методика аналогичная. В статье я привел преимущества проксирования относительно программ шифрующих пароль, но видимо надо было особо указать что нешифрующие – тем более небезопасны. Ах да, и “декомпиляторы” для Autolt существуют. Ах да, и вытащить пароль из “программы” при вашем подходе наверняка можно и с помощью strings.exe 😊
И еще раз напомним, неважно даже как хранится пароль, для использования – его придётся расшифровать и ввести, и в этот момент его можно и перехватить.



MaximillianGr

17 November,
2009

Василий, статья хорошая.
Непонятен момент: “На компьютере постоянно работает сценарий с высокими привилегиями. Например, в нашем случае он будет запущен из-под учетной записи, обладающей правами администратора на файловом сервере.”
Как предполагается его запускать?



Василий
Гусев

17 November,
2009

MaximillianGreat: С помощью планировщика например. Я в статье написал же.



MaximillianGr

17 November,
2009

Чем сохранённый в планировщике пароль принципиально лучше сохранённых учётных данных в других приложениях?



Василий
Гусев

17 November,
2009

MaximillianGreat: Тем что он сохранён в планировщике на сервере, к которому пользователь не имеет никакого доступа, кроме как запись в определенную шару 😊



Ну тогда это сильно не аналог остальным способам. Область



MaximillianGr

17 November,
2009
Василий
Гусев17 November,
2009

применения совсем другая.
MaximillianGreat: Именно, "Этот метод отличается от всех
вышеперечисленных, и служит совсем для других задач" 😊
Возможно действительно стоило вынести его в отдельную
статью/пост, а вместо него рассказать например о телнете 😊



MaximillianGr

17 November,
2009

Так же очень сильно сбивает с толку сравнение с sru.exe и runas.

Василий
Гусев17 November,
2009

А вот sru и runas /savecred в последнем примере очень даже к
месту. Именно для них я хотел показать безопасную альтернативу.



MaximillianGr

17 November,
2009

Не понял. Как их можно использовать на "сервере, к которому
пользователь не имеет никакого доступа, кроме как запись в
определенную шару"

Василий
Гусев17 November,
2009

Сервер это уже детали реализации. А реализуемая задача –
"делегирование одной административной функции обычным
пользователям"



MaximillianGr

17 November,
2009

Не. Детали реализации это файл флаг или сокет на сервере 😊 Тут
всё-таки разные области применения.
Или я всё-таки чего-то не понимаю.



MaximillianGr
Гусев
17 November,
2009

Область доступа не одна, а 6. Кроме образцов, которые будут работать с удалённым сервером?
Возьмем пример из статьи – дать возможность конкретному пользователю закрыть сессии на сервере, не давая административных прав на этом же сервере. Можно решить с помощью runas /savecred или sraи на клиенте, или с помощью проксирования. Но только первые два метода будут небезопасны.

6 способов мало? 😞



Василий
Гусев

17 November,
2009



MaximillianGr

17 November,
2009

Василий, вы что-то меня совсем запутали 😊
Если “на сервере, к которому пользователь не имеет никакого доступа, кроме как запись в определенную шару” то => 6 способов он может использовать только после повышения привилегий у себя на компьютере, так?
Вообще пример какой-то совсем не жизненный – пользователь повышает привилегии на _своём_ компьютере, для того что сделать что-то на _другом_. Вам часто такое встречалось?



Василий
Гусев

17 November,
2009

Пример как раз вполне жизненный. Причем реализовывал я его двумя разными методами. Где повышать привилегии – обычно не важно. Если в домене обладаешь учеткой с правом доступа на сервер – можешь подключиться к нему и удалённо (что поясняют первые способы). А вот безопасно дать возможность обычному пользователю возможность выполнить одну команду с административными правами – сложно.



MaximillianGr

17 November,
2009

>Где повышать привилегии — обычно не важно
обычно – не важно, но в данном примере это имеет принципиальное значение.



MaximillianGr

18 November,

хаегр, да, вот ещё что хотел спросить, вчера забыл 😊
С WinRM 2.0 vs WinXP как дела будут обстоять?
Windows Management Framework это оно? Или нет?
Плюс я ещё слышал что в XP ремотинг с иисом на одном порту не



Хаегр

18 November,
2009

работает, это так?

MaximillianGreat: Для XP всё отлично, это как раз Windows Management Framework. Я вставил врезку в статью об этом. Ремонтиг по умолчанию сейчас вообще использует не 80/443, а другие порты. Будут ли проблемы если повесить на один – не знаю, не пробовал, но мне кажется нет.



Azs

26 November,
2009

За инфу о PowerShell спасибо большое.



Хаегр

26 November,
2009

Azs: Заходите ко мне в блог, там еще много 😊



Courier

27 November,
2009

А почему в выводе, например, "psexec \main ipconfig /all" вместо русских – кракозябры?



Хаегр

27 November,
2009

2 Courier: Это бага нескольких локализованных консольных утилит... Фикс уже врядли будет, так что:

<http://xaegr.wordpress.com/2007/01/24/decoder/>

или

<http://xaegr.wordpress.com/2008/02/12/decode-2cp866/>



NeoNaft

27 November,
2009

Интересно а можно автоматику прибить? чтобы автоматически конвертилось?





NeoNaft
27 November,
2009

27 November,
2009

Можно использовать netsh, он не страдает такими проблемами. Лучший PoSh маньяк России? 😊



Хаегр

27 November,
2009

Нео, пока netsh не заменили ps-модулем, я буду рекомендовать его 😊



Nyuk

4 December,
2009

Если на удаленном хосте нет привилегий админа, какой из способов сработает ?



Хаегр

4 December,
2009

Nyuk: Только последний. Но разумеется для его подготовки и организации нужны привилегии администратора. Кроме того, PowerShell remoting позволяет настроить разрешения таким образом что к компьютеру смогут подключаться и не администраторы (Set-PSSessionConfiguration microsoft.powershell -ShowSecurityDescriptorUI). Разумеется они будут работать со своими привилегиями, но удалённо. Кроме того можно всячески урезать доступные им командлеты, параметры, и вообще функционал языка (если вы не хотите чтобы пользователь с помощью for сожрал память на сервере 😊). Я планирую серию постов у себя в блоге по этой теме. Для некоторых других методов тоже возможно делегирование возможностей удалённого доступа не администраторам, но это крайне сложно и зачастую нецелесообразно. Не говоря уже о безопасности.

Загрузка
Nive на
удаленном
компьютере.
« Kazun

7 December,
2010

[...] Ps. Так же прочтите статью Василия Гусева – 7 способов выполнить команду на удалённом компьютере [...]

7 способов

выполнить
команду на
удалённом
компьютере



5 February,
2014

5 February,
2014

[...] [commentary was posted in Windows](#). Bookmark the permalink.
← Как изменить UID продукта F-Secure без переустановки антивируса
LikeBe the first to like this post. [...]

Чет не работает PsExec с сетевыми ресурсами.
Запускаю на удаленном юзерском компе инсталляшку с файлового сервера. Моментально вываливается ответ, что прога exited on КОМП with error code 0, и ессно, ничего не установлено.
Вариант с подключением сетевого диска тоже не канаает. Только копировать прогу на целевой комп, и запускать оттуда.
Это у меня баг, или общая фича?



Колян

22 April, 2016

Эт всё одни способ! Программы ток разные!

Имя

Ваш комментарий

Опубликовать

© 2009-2017 ITBand.ru

При использовании материалов, ссылка обязательна.
66 / 2.624 / 30.67mb