Визитка

ИВАН КОРОБКО, сертифицированный специалист MCP, автор более 50 статей и двух книг. Занимается созданием различных приложений для Active Directory

На языке PowerShell

Сценарий регистрации пользователей в сети. Часть 3

Подключение сетевых ресурсов — одна из приоритетных задач, решаемых сценарием регистрации пользователей в сети. В этот раз мы поговорим о динамическом управлении сетевыми дисками

Традиционно к сетевым ресурсам относятся сетевые диски и принтеры. В настоящее время этот список можно расширить и отдельно вычленить сетевые приложения. Если говорить точнее – ярлыки к сетевым приложениям. Их, как и сетевые диски, можно обнаружить в папке «Мой Компьютер». К сетевым приложениям можно отнести сетевые версии программ, например «1С» или «Консультант+», и portable-приложения, не требующие установки на персональный компьютер.

Идентификация групп безопасности

Динамическое управление подключением и отключением сетевых ресурсов осуществляется на основе принадлежности пользователя к соответствующей группе безопасности. Сценарий идентифицирует эти группы на основе префикса, указанного в конфигурационном файле в формате XML [1].

В таблице 1 приведен список рекомендуемых идентификаторов групп безопасности.

Для упрощения администрирования эти группы рекомендуется использовать не только для подключения сетевых ресурсов как таковых, но и для назначения прав на файловую систему NTFS. При таком использовании префиксов рекомендуется создавать префиксы, состоящие из двух частей: подключение сетевых ресурсов осуществлять по первой, одинаковой, части, а область действия или функционал фиксировать во второй части. Таким образом, уникальный идентификатор превращается в нечто похожее на командлет.

Например, для управления файловой системой в большинстве случаев достаточно двух наборов прав: Read и Modify. Поэтому префиксы групп безопасности для подключения сетевых дисков трансформируются в dsk$-read$_ и dsk$-modify$_ соответственно. В свойствах XML-файла

Таблица 1. Идентификаторы групп безопасности

Идентификатор	Описание
dsk$_	Подключение сетевого диска
prn$_	Подключение сетевых принтеров
mc$_	Подключение ярлыков сетевых приложений

Таблица 3. Описание переменных, используемых для подключения сетевых дисков

Переменная	Описание
$domain	Текущий домен. Определяется через переменные среды
$department	Подразделение, к которому относится пользователь. Определяется на основе данных Active Directory
$FIO	ФИО пользователя. Определяется на основе данных Active Directory

Таблица 2. Характеристики группы безопасности для подключения сетевых дисков

Характеристика	Параметр	Вкладка	Поле в Active Directory	Тип данных
Название группы	Name	General	cn	Строка
Имя диска (буква)	Description	General	Description	Строка
Описание диска	Notes, 1 строка	General	Info[0]	Многострочный
Точка монтирования	Notes, 2 строка	General	Info[1]	Многострочный
Список пользователей, к которым подключается данный сетевой диск	Members	Members	Member[i]	Массив

рекомендуется указывать не два префикса, а один – первую часть префикса, т.е. dsk$- (см. рис. 1).

Аналогична ситуация с сетевыми принтерами. Здесь также используются две группы: одна для подавляющего большинства пользователей и позволяет только распечатывать данные (prn$-print$_), вторая – управлять очередью печати (prn$-manage$_). Вместо прав доступа на файловую систему они устанавливаются для принтера, находящегося на сервере печати.

Механизм подключения, а соответственно и его реализация, обеспечивающая подключения ссылок на сетевые ресурсы, в том числе и ресурсы сети Интернет, несколько сложнее. Частично эта тема затрагивалась в [2].

Остановимся подробно на подключении сетевых дисков к рабочей станции. О подключении принтеров и создании ярлыков к сетевым дискам речь пойдет в следующих статьях.

Характеристики группы безопасности

При входе пользователя в сеть среди прочих сценариев выполняющийся сценарий анализирует группы безопасности, названия которых содержат идентификатор, считываемый из конфигурационного файла данного сценария. Для сетевых дисков идентификатором служит префикс dsk$- (см. таблицу 1). Для каждой группы, используемой для того или иного типа ресурсов, применяется индивидуальный набор характеристик: для принтеров один, а для дисков другой. Для подключения сетевых дисков используются характеристики группы безопасности, указанные в таблице 2.

Определение точки монтирования сетевых дисков

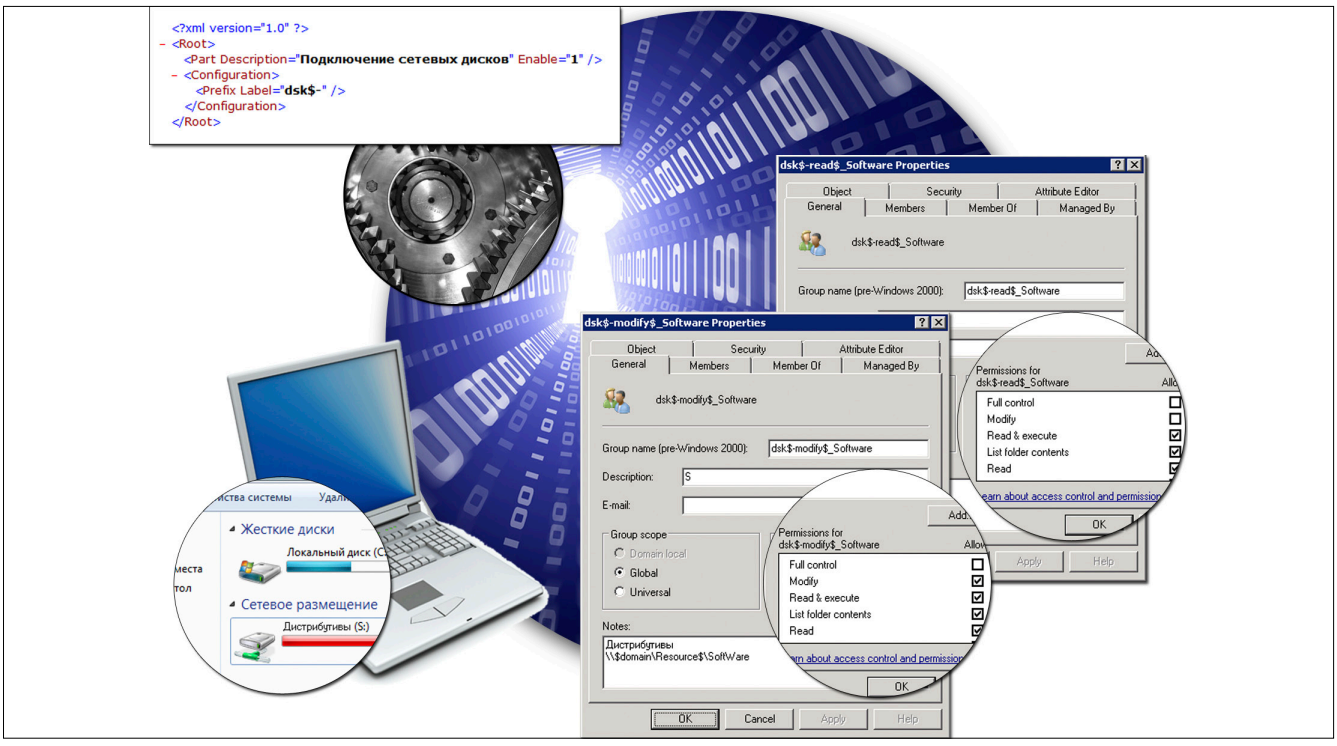
Точка монтирования диска – самая важная характеристика, хранящаяся в свойствах группы безопасности. В большинстве случаев она индивидуальна, как и данные, хранимые пользователем. Создавать для каждого сотрудника персональную группу нежелательно, поскольку это усложнит администрирование ресурсов. Для наглядности приведем небольшой расчет. Если в эксплуатируемой вами сети есть хотя бы три сетевых диска, которые подключаются всем сотрудникам, то при большом количестве работающих необходимо создать 1500 групп безопасности против трех. Как видно, перевес в разы. Порядок определяется количеством пользователей.

Скрипт подключения сетевых дисков, сформировав UNC-путь к ресурсам, выполняет их подключение. Чтобы обеспечить минимальное количество групп безопасности, в пути, участвуют переменные, значение которых приведено в таблице 3.

Непосредственно в сценарии подстановка переменных осуществляется в функции Freplace, которая имеет один параметр – точку монтирования, например, \\$domain\$department\$FIO\Персональные данные. В начале функции описаны два массива. В первом из них ($ArFrom) содержатся переменные, которые необходимо заменить, во втором ($ArTo) – имена переменных, на которые необходимо заменить переменные. При последовательном перебирании элементов массива и анализе UNC-пути если переменная найдена, то осуществляется ее замена на заранее определенное значение. В конце работы функции строка не будет содержать каких-либо переменных.

Вызов функции может быть реализован двумя способами: традиционным и с помощью конвейера. Принципиальной разницы в данном случае нет, однако для удобства рекомендуется использовать передачу параметра функции по конвейеру.

Рисунок 1. Схема взаимодействия скрипта для сетевых дисков и файловой системы



По правилам синтаксиса PowerShell функция должна быть определена перед ее вызовом, поэтому вызов функции находится в конце примера, приведенного в листинге 1. В результате работы сценария значение $PathToFolder будет преобразовано в \\island\Бухгалтерия\Иванов Петр Иванович\Персональные данные.

Листинг 1. Функция замены переменных на значения

```
Function Freplace{
# Элементы массива $ArFrom - заменяемые переменные
$ArFrom = ('$fio'), ('$domain'), ('$department')
# Элементы массива $ArTo - подставляемые значения
$ArTo = $fio, $sdomain, $department
    ForEach ($element in $input)
    {
    $temp= New-Object System.Text.StringBuilder
    For ($i=0; $i -lt ($ArFrom.length); $i++)
        {
        if ($element -like ("*"+$ArFrom[$i]+"*"))
            {
            $temp=$element
            $element=$temp.Replace($ArFrom[$i] ,$ArTo[$i])
            }
        }
    }
    $element     # Возвращаемое значение
}

$fio = 'Иванов Петр Иванович'
$sdomain = 'island'
$department = 'Бухгалтерия'
$PathToFolder = '\\$domain\$department\$FIO\ ↵
       Персональные данные'
$PathToFolder | FReplace
```

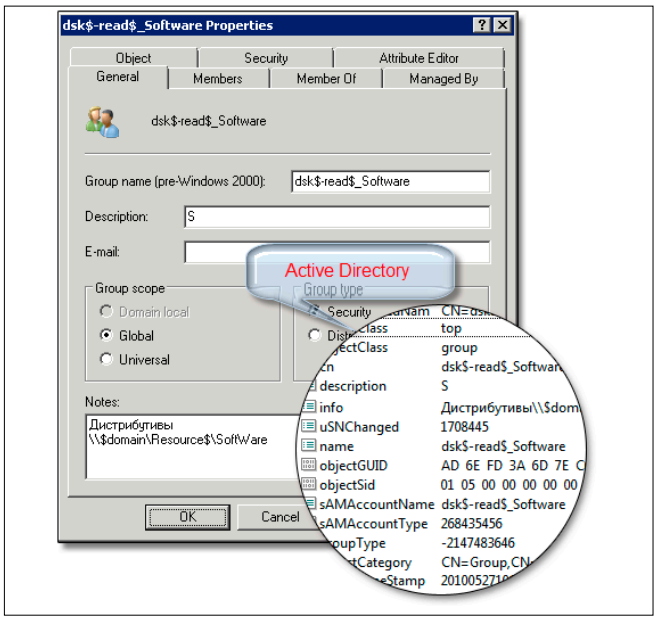
Формирование точек монтирования

Использование сценария с вычисляемой точкой монтирования подразумевает наличие файловой структуры, устроенной по каким-либо закономерностям. На выбор закономерности влияет множество факторов: безопасность, отказоустойчивость, квотирование рабочего пространства, наличие иерархии и многое другое. Проектирование системы – непростая задача, однако ее решение становится очевидным, если в качестве отправной точки рассматривать иерархическую структуру штатного расписания.

Как бы ни выглядела построенная файловая система, в ней будут присутствовать точки монтирования для дисков, подключаемых всем пользователям (личные файлы, диск для обмена данными) и точки монтирования для необязательных дисков (дистрибутивы программного обеспечения). Список обязательных дисков и некоторых необязательных приведен в таблице 4.

Файловая структура представляет собой иерархическую структуру. В качестве инструмента, позволяющего увеличить гибкость и безопасность файловой системы, рекомендуется использовать службу DFS (Distributed File System) и ABE (Access Based Enumeration). Описание построения распределенной файловой структуры для организации выходит за рамки данной статьи [3].

Рисунок 2 . Группа безопасности для подключения сетевых дисков



Определение параметров для подстановки

В таблице 3 приведен примерный список параметров, которые могут использоваться для подстановки. Вполне возможно, что для использования в вашей системе необходимо ввести новые переменные.

Для получения информации о пользователе, домене, рабочей станции используется всего два источника: переменные среды и данные из каталога Active Directory. Некоторые данные пересекаются, и их можно получить из любого источника.

Чтение переменных среды

В традиционной оболочке командной строки для определения переменной среды используется команда Set. В оболочке PowerShell для работы с этим списком используется виртуальный диск ENV:. Список доступных дисков можно получить с помощью командлета Get-PSDrive.

Чтение значения переменной среды осуществляется с помощью конструкции:

```
$ЗНАЧЕНИЕ=(Get-ChildItem ENV:ПЕРЕМЕННАЯ).Value
```

где:

ПЕРЕМЕННАЯ – переменная среды, значение которой необходимо получить;

ЗНАЧЕНИЕ – значение переменной среды в виде строки.

Если вспомнить о том, что Get-ChildItem имеет псевдоним DIR, то шаблон можно упростить:

```
$ЗНАЧЕНИЕ=(DIR ENV:ПЕРЕМЕННАЯ).Value.
```

Переменные среды позволяют определить такие важные характеристики, как имя пользователя в сети (Login), сокращенное имя домена (userdomain), сервер загрузки (logonserver) и т.д. Примерный список переменных среды, используемых в Windows 7, приведен в таблице 5.

Он может изменяться при установке 64-битной версии ОС. Поскольку список переменных меняется в зависимости от версии Windows, то при создании сценария необходимо учитывать этот факт, используя только те переменные, которые задействованы во всех версия операционных систем, составляющих парк рабочих станций вашей сети. В против-

ном случае на некоторых рабочих станциях сценарий будет работать некорректно.

Чтение информации из каталога Active Directory

Механизм получения данных из каталога Active Directory был описан в [4], поэтому ограничимся лишь шаблоном, обеспечивающим доступ к каталогу Active Directory, который приведен в листинге 2. Для идентификации объекта при составлении фильтра (метод filter()) используется два параметра. Первый из них определяет тип объекта – objectClass. Возможные значения этого параметра приведены в таблице 6. Второй параметр – сокращенное имя объекта в сети, используемое для идентификации пользователя в сети. На жаргоне системных администраторов – это «логин», в каталоге Active Directory он хранится как значение атрибута sAMAccountName.

Таблица 4. Список дисков, подключаемых пользователям

Буква диска	Назначение	Статус
H (Home)	Персональная папка пользователя. Содержит два каталога: «Входящие файлы» и «Персональные файлы»	Постоянный диск
X (eXchange)	Диск обмена данными. Внутри – список папок, которые по своей сути являются «Входящими файлами»	Постоянный диск
W (Work)	Общая (рабочая) папка подразделения (отдела)	Постоянный диск
P (Project)	Проект – логическое объединение двух и более дисков W	Дополнительный диск
S (Soft)	Диск с дистрибутивами программного обеспечения. Доступен сервисной службе ИТ-подразделения	Дополнительный диск
M (Manager)	Диск для руководителя. Позволяет просматривать содержимое дисков сотрудников вверенного ему подразделения	Дополнительный диск

Таблица 5. Список переменных среды

Переменная	Описание
%ALLUSERSPROFILE%	Размещение профиля All Users
%APPDATA%	Используемое по умолчанию размещение используемых приложениями данных
%COMMONPROGRAMFILES%	Путь к папке Common Files
%COMPUTERNAME%	Имя компьютера
%COMSPEC%	Путь к исполняемой командной оболочке
%HOMEDRIVE%	Имя диска локальной рабочей станции, связанного с основным каталогом пользователя. Задается на основании расположения основного каталога. Основной каталог пользователя указывается в оснастке «Локальные пользователи и группы»
%HOMEPATH%	Полный путь к основному каталогу пользователя. Задается на основании расположения основного каталога. Основной каталог пользователя указывается в оснастке «Локальные пользователи и группы»
%LOCALAPPPATH%	Путь к папке AppData\Local в профиле пользователя
%LOGONSEVER%	Имя контроллера домена, который проверял подлинность текущей сессии
%NUMBER_OF_PROCESSORS%	Количество процессоров, установленных на компьютере
%OS%	Имя операционной системы. При использовании Windows 2000 и XP имя операционной системы отображается как Windows_NT
%PATH%	Путь поиска для исполняемых файлов
%PROCESSOR_ARCHITECTURE%	Архитектура процессора. Возможные значения: x86, IA64
%PROCESSOR_IDENTFIER%	Описание процессора
%PROCESSOR_LEVEL%	Номер модели процессора, установленного на компьютере
%PROGRAMFILES %	Путь к папке с установленным программным обеспечением
%SESSIONNAME%	Имя сессии. Определяется приложением, которым запущен интерпретатор
%SYSTEMDRIVE%	Имя диска, содержащего корневой каталог Windows XP (то есть системный каталог)
%SYSTEMROOT%	Размещение системного каталога Windows XP
%TEMP% %TMP%	Временные папки, по умолчанию используемые приложениями, которые доступны пользователям, выполнившим вход в систему. Некоторые приложения используют переменную TEMP, некотрые – переменную TMP
%USERDNSDOMAIN%	DNS-имя домена
%USERDOMAIN%	NetBIOS-имя домена
%USERNAME%	Имя пользователя, выполнившего вход в систему
%USERPROFILE%	Размещение профиля для текущего пользователя
%WINDIR%	Размещение каталога операционной системы

Листинг 2. Шаблон поиска объектов в каталоге Active Directory

```
# Определение имени домена в формате RDN
$root=[ADSI]'LDAP://RootDSE'
$domain = $root.defaultNamingContext
# Поиск пользователя по login (поле sAMAccountName)
$obj = New-Object DirectoryServices.Directorysearcher( ↵
       "LDAP://"+$domain)
$obj.Filter="(&(АТРИБУТ1=ЗНАЧЕНИЕ1)( АТРИБУТ1=ЗНАЧЕНИЕ1)(…))"
$Searcher=$Obj.FindOne()
# Чтение значение нужного поля
$Result=$Searcher.GetDirectoryEntry()
$СЧИТАННОЕ_ЗНАЧЕНИЕ=$Result.Properties.Item( ↵
       "СЧИТЫВАЕМОЕ_СВОЙСТВО").Value
```

где:

АТРИБУТ1, АТРИБУТ2… – название атрибута Active Directory. Пример: objectClass, sAMAccountName, cn и др;

ЗНАЧЕНИЕ1, ЗНАЧЕНИЕ2… – соответствующее известное значение указанного атрибута;

СЧИТАННОЕ_ЗНАЧЕНИЕ – определяемое значение;

СЧИТЫВАЕМОЕ_СВОЙСТВО – имя атрибута, значение которого необходимо получить.

Комбинированное использование методов

Некоторые данные, например ФИО пользователя, можно определить, используя только оба этих метода. В листинге 3 приведен пример определения ФИО пользователя, хранящегося в поле Description учетной записи пользователя. Поскольку имя пользователя в сети уникально, то вместо метода FindAll(), возвращающего массив объектов, рекомендуется использовать метод FindOne().

Листинг 3. Определение ФИО пользователя

```
# Определение имени (login) текущего пользователя
# переменные окружения set
$UserLogin=(dir env:username).value
# Определение имени домена в формате RDN
$root=[ADSI]'LDAP://RootDSE'
# Поиск пользователя по login (поле sAMAccountName)
$domain = $root.defaultNamingContext
$objUser = New-Object DirectoryServices.Directorysearcher( ↵
       "LDAP://"+$domain)
$objUser.filter="(&(objectclass=person) ↵
       (samaccountname="+$UserLogin+"))"
$searchUser=$objUser.findone()
# Чтение значение нужного поля
$ResultUser=$searchUser.getdirectoryentry()
$FIO=$ResultUser.properties.item("description").value
```

Подключение диска

Как это ни странно звучит, но в PowerShell нет встроенного командлета, используемого для подключения сетевых дисков Windows. Решить поставленную задачу можно тремя способами:

- используя команду net use оболочки командной строки, которую поддерживает PowerShell;
- с помощью COM-объекта Windows Script Host (WSH);
- помощью API-функций.

Все перечисленные способы в конечном счете сводятся к управлению соответствующей API-функцией. Не рекомендуется использовать первый способ, если в сети эксплуатируется PowerShell 1.0. Он корректно работает только в Windows Management Framework Core (PowerShell 2.0 + Win RM).

Внимание: командлет New-PSDrive подключает внутренние диски PowerShell, которые недоступны проводнику Windows.

Использование команды net use

Команду net use знают абсолютно все системные администраторы. Она очень органично вписалась в Windows PowerShell. В первой версии PowerShell разработчиками была допущена досадная ошибка, следствием чего стало выборочное подключение сетевых дисках на некоторых рабочих станциях. В обновленной версии Windows PowerShell 2.0, объединенной с WinRM в пакет Windows Management Framework Core, эта ошибка исправлена.

В листинге 4 приведен фрагмент сценария, в котором осуществляется подключение сетевого диска.

Листинг 4. Подключение сетевого диска с помощью команды net use

```
…
$ConnectPath = $PathToFolder | FReplace
$ConnectLetter = ↵
       $ObjGroup.Properties.Item("description").value
Net Use $ConnectLetter":" $ConnectPath
```

Использование COM-объекта

Вспомнив, что ранее сценарии использовали только с COM-объектами, применим эту технологию в Windows PowerShell. Для работы с сетевыми дисками используется встроенный для совместимости в операционную систему объект WShNetwork. Подключение сетевого диска осуществляется вызовом метода MapNetworkDrive, который имеет два параметра: имя диска и путь к диску. В листинге 5 приведен пример подключения сетевого диска с помощью WSH.

Листинг 5. Подключение сетевого диска с помощью COM-объекта

```
…
$ConnectPath = $PathToFolder | FReplace
$ConnectLetter =
       $ObjGroup.Properties.Item("description").value

$obj = New-Object -ComObject Wscript.Network
$obj.MapNetworkDrive($t+":", $path)
```

Использование API-функции

Использование команды net use или COM-объекта для подключения сетевого диска в конечном счете сводится к вызову API-функции. Поэтому этот способ самый надежный и самый быстрый с точки зрения скорости подключения ресурса, хотя и не из простых.

В PowerShell не существует командлета, позволяющего вызывать API-функции. Поддержка библиотек .NET Framework также не позволяет реализовать их вызов. Единственные решение – интегрировать фрагмент листинга на VB.NET или C#, скомпилировать в памяти и выполнить. То, как реализуется этот алгоритм на практике, было подробно описано в одной из предыдущих статей [5].

Подключение сетевого диска осуществляется с помощью API-функции WNetAddConnection2, хранящейся в файле mpr.dll, имеющей несколько параметров:

netResource – структура, содержащая описание подключаемого ресурса: UNC-путь, букву диска и т.д.; подроб-

ное описание характеристик можно найти на официальном сайте Microsoft [6];

password – пароль учетной записи, с привилегиями которой осуществляется подключение ресурса;

Username – имя учетной записи, с привилегиями которой осуществляется подключение ресурса;

Flag – всегда равен 0.

Nothing – имя учетной записи и пароль, если подключение осуществляется от имени текущего пользователя.

В листинге 6 приведен пример подключения сетевого диска с помощью API-функции:

Листинг 6. Подключение сетевого диска с помощью API-функции

```
…
$ConnectPath = …..
$ConnectLetter = ↵
        $ObjGroup.Properties.Item("description").value
# Подключение к компилятору VB.NET
$provider = New-Object Microsoft.VisualBasic.VBCodeProvider
$params = New-Object System.CodeDom.Compiler. ↵
        CompilerParameters
$params.GenerateInMemory = $True
$refs = "System.dll","Microsoft.VisualBasic.dll"
$params.ReferencedAssemblies.AddRange($refs)

# Листинг API-функции на VB.NET
$txtCode = @"
Class ConnectNetDisk
<System.Runtime.InteropServices.StructLayout(System. ↵
        Runtime.InteropServices.LayoutKind.Sequential)> ↵
        Public Structure NETRESOURCE
Public dwScope As Integer
Public dwType As Integer
Public dwDisplayType As Integer
Public dwUsage As Integer
Public LocalName As String
Public RemoteName As String
Public Comment As String
Public Provider As String
End Structure
Public Declare Function WNetAddConnection2 Lib "mpr.dll" ↵
        Alias "WNetAddConnection2A" (ByRef netResource ↵
        As NETRESOURCE, ByVal password As String, ↵
        ByVal Username As String, ByVal Flag As Integer) ↵
        As Integer
Function ConnectDisk(a,b)
Dim myNetResource As New NETRESOURCE
myNetResource.dwScope = 2 'RESOURCE_GLOBALNET
myNetResource.dwType = 1 'RESOURCETYPE_DISK
myNetResource.dwDisplayType = 3 'RESOURCEDISPLAYTYPE_SHARE
myNetResource.dwUsage = 1 'RESOURCEUSAGE_CONNECTABLE
myNetResource.LocalName = a
myNetResource.RemoteName = b
myNetResource.Provider = Nothing
Dim ret As Integer = WNetAddConnection2(myNetResource, ↵
        Nothing, Nothing, 0)
ConnectDisk = cstr(ret) + "   "+a + "  " + b
End function
End Class
"@

# Вызов компилятора
$results = $provider.CompileAssemblyFromSource($params, ↵
        $txtCode)
   $mAssembly = $results.CompiledAssembly
# Вызов VB.NET-функции, использующей API-функцию
# для подключения диска
   $i = $mAssembly.CreateInstance("ConnectNetDisk")
   $result = $i.ConnectDisk([string]($ $ConnectLetter+":"), ↵
        [string]($ConnectPath))
$result
```

***

Сложность создаваемого сценария напрямую зависит от задачи, которую он решает. Однако, вне зависимости от сложности сценария необходимо следить, чтобы соблюдались очень простые правила: сделайте свой сценарий очень надежным в работе, все внешние настройки сосредоточьте во внешнем файле настроек. Благодаря этому администратор сможет быстро отреагировать на изменение внешних условий работы скрипта и обеспечить высокое качество работы обслуживаемых сервисов.

В следующей статье поговорим о динамическом управлении подключением сетевых принтеров. EOF

1. Коробко И. На языке PowerShell. Сценарий регистрации пользователей в сети. Часть 1. //«Системный администратор», № 9, 2010 г. – С. 43-45.
2. Коробко И. Создаём персональный набор ярлыков для каждого пользователя в папке «Мой Компьютер». //«Системный администратор», №3, 2008 г. – С. 24-27 (http://samag.ru/archive/article/802).
3. Коробко И. Проектируем систему обмена данными. //«Системный администратор», №9, 2007 г. – С. 24-27 (http://samag.ru/archive/article/785).
4. Коробко И. Управление объектами в Active Directory. Часть 3. //«Системный администратор», №7, 2008 г. – С. 30-35.
5. Коробко И. Расширяем возможности. PowerShell: другие языки программирования. //«Системный администратор», №1-2, 2010 г. – С. 86-89.
6. NETRESOURCE Structure – http://msdn.microsoft.com/en-us/library/aa385353(VS.85).aspx.

Таблица 6. Список – идентификатор объекта для поиска в каталоге Active Directory

Комментарий	Тип объекта	Фрагмент поискового запроса
Учетная запись компьютера	Computer	objectClass='Computer'
Контакт, используется в почтовых приложениях	Contact	objectClass='Contact'
Группа безопасности	Group	objectClass='Group'
Учетная запись пользователя, не совместимая с доменами Windows 2k	InetOrgPerson	objectClass='InetOrgPerson'
Папка дерева каталогов Active Directory	OU	objectClass='OrganizationalUnit'
Опубликованный в Active Directory сетевой принтер	Printer	objectClass='PrintQueue'
Опубликованная в Active Directory сетевая папка	Shared Folder	objectClass='Volume'
Учетная запись пользователя, совместимая с доменами Windows NT	User	objectClass='User' and not objectClass='Computer'
Контейнер (папка), создаваемая в каталоге Active Directory по умолчанию	Container	objectClass='Container'