Einleitung

In dieser Einleitung:

Was lesen Sie in diesem Buch? 21
Wo gibt es die Beispiele? 21
Download, Installation und Systemvoraussetzungen 22
Die PowerShell lernen 22
Kontakt zum Autor 23
Danksagungen 23

Peter Monadjemi: Windows PowerShell 2.0 - Crashkurs, Microsoft Press 2010 (ISBN 978-3-86645-667-9)

Die *Windows PowerShell* ist eine moderne Shell, die Microsoft kostenlos für alle Windows-Versionen ab Windows XP zur Verfügung stellt, und die bei Windows Server 2008 R2 und Windows 7 in der aktuellen Version 2.0 von Anfang an integriert ist. Mit der PowerShell lassen sich sowohl kleine Ad-hoc-Abfragen wie »Liste alle Prozesse auf, die länger als 24 Stunden laufen, und die mehr als 10 MB Arbeitsspeicher belegen« als auch komplexe Skripts, die aus mehreren Hundert Befehlszeilen bestehen, ausführen. Die PowerShell richtet sich damit sowohl an den gelegentlichen Anwender als auch an den viel zitierten Power-User, der mit seinen Skripts ein aus 5.000 und mehr Arbeitsplätzen bestehendes Windows-Netzwerk administrieren möchte.

Die PowerShell wurde im Herbst 2006 in der Version 1.0 freigegeben. Seit Oktober 2009 liegt die Version 2.0 vor, die zahlreiche Verbesserungen in Gestalt erweiterter und zusätzlicher Befehle (Cmdlets) und Operatoren enthält. Die wichtigste Neuerung ist das in der Version 1.0 noch schmerzlich vermisste Remoting (auf der Grundlage von *Windows Remoting* und *WS-Management*), mit dem Befehle und ganze Skripts auf theoretisch beliebig vielen Computern im Netzwerk gleichzeitig ausgeführt werden (dazu muss auf allen diesen Computern die PowerShell 2.0 installiert sein). Mit der Version 2.0 gibt es für das Erstellen von Skripts mit der *PowerShell ISE* einen netten und vor allem flexibel erweiterbaren Editor, sodass PowerShell-Skripter nicht länger auf Notepad angewiesen sind.

Die PowerShell ist ein wenig anders als *Cmd.exe*, Stapeldateien, der *Windows Script Host* und andere Shells wie z.B. *Bash*. Die PowerShell setzt konsequent auf Objekte und eine Objektpipeline. Bei der PowerShell ist wirklich alles ein Objekt. Was sich zunächst nach einer eher akademischen Nuance oder dem üblichen Microsoft-Marketingslang anhören könnte (was es definitiv nicht ist), stellt sich schnell als ein unschätzbarer Vorteil heraus, der PowerShell-Befehle nicht nur einfacher, sondern deutlich logischer macht, als es bei Stapeldateien oder einem WSH-Skript der Fall ist. Hier ein kleines Beispiel: Möchte man alle Freigaben auflisten, in denen das Wort *Skript* vorkommt, erledigt das ein

```
Net Share | Findstr "Skript"
```

unter dem alten Befehlsinterpreter *Cmd.exe* schnell und zuverlässig. Doch was ist, wenn diese Freigaben nicht nur aufgelistet, sondern auch gelöscht werden sollen? Ein

```
Net Share | Findstr "Skript" | Net Share /delete
```

funktioniert (natürlich) nicht, weil der *Net Share*-Befehl dafür nicht ausgelegt ist. Echte Cmd-Profis könnten sich eine Stapeldatei basteln, welche die Aufgabe zuverlässig erledigt, doch würde der Aufwand vermutlich in keinem Verhältnis zum Nutzen stehen. Bei der PowerShell bietet die Objektpipeline eine genauso einfache wie elegante Lösung:

```
Get-WmiObject -Class Win32_Share -Filter "Name like '%skript%'" | Invoke-WmiMethod -Name Delete | Out-Null
```

Dies ist ein typischer PowerShell-Befehl, bei dem über den Pipe-Operator | gleich drei *Cmdlets* (so werden die ausführbaren PowerShell-Befehlsobjekte genannt) verknüpft werden. Zuerst holt das Cmdlet *Get-WmiObject* alle Freigaben, in deren Namen ein *Skript* vorkommt, und übergibt sie anschließend über die Objektpipeline an das folgende *Invoke-WmiMethod*-Cmdlet, das für jedes Objekt dessen *Delete*-Methode aufruft, die das Objekt und damit die Freigabe löscht. Das angehängte *Out-Null*-Cmdlet sorgt lediglich dafür, dass die »Löschbestätigung« verschluckt wird.

Diese Variante mag auf den ersten Blick nicht besonders attraktiv und innovativ wirken, doch steckt dahinter ein großes Potenzial. Es spielt nämlich keine Rolle, ob Freigaben, Prozesse (lokal wie remote), Benutzerkonten in Active Directory oder Postfächer auf einem Exchange Server gelöscht werden sollen, das Prinzip ist stets exakt dasselbe (auch wenn natürlich jedes Mal andere Cmdlets im Spiel sind). Das macht die PowerShell nicht nur flexibler, sondern auch leichter erlernbar.

Was lesen Sie in diesem Buch?

Der *PowerShell-Crashkurs* von *Microsoft Press* soll Sie anhand zahlreicher Beispiele in die faszinierende Welt der PowerShell einführen und deutlich machen, wo die PowerShell für die moderne Systemadministration vorteilhaft eingesetzt werden kann. Im Vergleich zur ersten Auflage wurde das Buch komplett neu geschrieben und sein Inhalt nicht nur auf die Version 2.0 abgestimmt, sondern auch die Themen stärker an das angepasst, was Anwender heutzutage im Praxisalltag an Know-how benötigen.

Manches mag in diesem Buch zunächst ein wenig kompliziert erscheinen, insbesondere die verschiedenen Skripts, die ab Kapitel 7 vorgestellt werden. Das könnte den Eindruck erwecken, dass die PowerShell nichts für Anwender ist, die nur ein paar einfache Aufgaben erledigen, dabei aber nicht zum Hardcoreexperten werden wollen. Hätten sich die Entwickler bei Microsoft nicht etwas mehr Mühe geben und die Dinge etwas einfacher gestalten können? Im Prinzip sicherlich, doch geht Einfachheit bekanntlich immer zu Lasten der Flexibilität und die tendiert bei der PowerShell in einigen Bereichen gegen unendlich. Und was das vermeintlich Komplizierte angeht: Stapeldateien sind teilweise deutlich kryptischer als ihre PowerShell-Pendants. Wer mit einem Batchbefehl wie

```
for /F "tokens=1,* delims= " %%i in (logon.txt) do w32tm /ntte %%i
```

auf Anhieb etwas anfangen kann, gilt als famoser Befehlszeilenguru (auch in den Augen des Autors). Doch um eine solche Befehlszeile verstehen und vor allem selbst erstellen zu können, bedarf es Ausdauer und eines harten Trainings. Bei der *PowerShell* sieht derselbe Befehl wie folgt aus:

```
Get-Content Logon.txt | ForEach-Object { W32tm /ntte $_ }
```

Vielleicht nicht ganz so elegant, aber in jedem Fall deutlich besser lesbar und vor allem nachvollziehbar (auf die Umsetzung der Optionen des *for*-Befehls wurde verzichtet, da diese beim *Get-Content*-Cmdlet keine Rolle spielen). Und das Beste daran: Diesen Befehl werden Sie problemlos selbst hinbekommen, wenn Sie das Buch (gewissenhaft) durcharbeiten. Bei der PowerShell gilt das Motto: Jeder kann zum Experten werden.

Wo gibt es die Beispiele?

In diesem Buch werden über 100 PowerShell-Skripts vorgestellt, die Sie natürlich nicht abtippen müssen. Sie finden sie als Download z. B unter *http://powershellcrashkurs.wordpress.com* oder auf der Verlagswebseite unter *http://www.microsoft-press.de/support.asp*.

Download, Installation und Systemvoraussetzungen

Download und Installation der Beispieldateien werden in Kapitel 2 beschrieben, an dieser Stelle nur so viel: Wer noch nicht mit Windows Server 2008 R2 oder Windows 7 arbeitet, muss die PowerShell 2.0 in Gestalt des *Windows Management Framework* von der Microsoft-Downloadseite herunterladen. Die PowerShell ist aufgrund ihrer eher bescheidenen Größe eine recht anspruchslose Anwendung, die in wenigen Minuten installiert werden kann. Voraussetzung ist das .NET Framework 2.0, das ab Windows Server 2008 und Windows Vista bereits von Anfang an dabei ist. Soll die *PowerShell ISE* benutzt werden (was im Allgemeinen empfehlenswert ist), muss mindestens das .NET Framework 3.0 installiert werden. (.NET 3.5 SP1 wird dann voraussetzt, wenn das Out-GridView-Cmdlet zum Einsatz kommen soll).

KB968929 und KB968930

Die PowerShell 2.0 wird nicht als Anwendung, sondern als Update installiert (die Version1.0 wurde als Hotfix installiert). Die Nummer des Updates, das neben der PowerShell 2.0 auch BITS 4.0 und WinRM 2.0 umfasst, ist KB968929, die für Windows Management Framework Core (ohne BITS 4.0) KB968930 (in der Registry werden die installierten Updates unter dem Schlüssel *HKey_Local_Machine\Software\Microsoft\CurrentVesion\Uninstall* aufgeführt). Der Download besteht aus einer Datei, deren Name vom Betriebssystem, deren Plattform und der Sprache abhängt (für ein deutschsprachiges 32-Bit-XP z.B. WindowsXP-KP968930-x86-DEU.exe).

Die PowerShell lernen

Zum routinierten PowerShell-Anwender, der so schnell Skripts schreibt, wie manche ihre Kurznachrichten ins Handy eintippen können, wird man nicht über Nacht. Das dauert im Allgemeinen eine Weile (ca. 1–2 Jahre). Das Buch soll sowohl Ihr Begleiter beim Kennenlernen der PowerShell-Philosophie sein als auch ein Nachschlagewerk für das Lösen von Alltagsproblemen. Daneben gibt es natürlich noch viele andere Quellen mit PowerShell-Know-how. Angefangen bei der wirklich umfassenden und verständlich geschriebenen PowerShell-Hilfe über die unzähligen Blogs und Webseiten der verschiedenen PowerShell-Gurus bis zum empfehlenswerten *TechNet Script Center,* das unter *http://gallery.technet.microsoft.com/scriptcenter/en-us/* PowerShell-Skripts zu praktisch allen Lebenslagen anbietet.

Eine hervorragende Gelegenheit, das eigene PowerShell-Wissen zu vermehren, ist inzwischen *Twitter* geworden. *Twitter* ist eine faszinierende Technik, die jeder einmal selbst ausprobieren sollte. Das folgende (kleine) PowerShell-Skript durchsucht den Twitter-Space nach Nachrichten, in denen das Wort *PowerShell* als Suchbegriff enthalten ist:

```
function Get-Twitter
($SuchBegriff)
{
  $Wc = New-Object -Type System.Net.WebClient
  $Results =
[Xml]($Wc.DownloadString("http://search.twitter.com/search.atom?rpp=100&page=1&q=$SuchBegriff"))
  $SearchItems = $Results.feed.entry
  $SearchItems
}

Get-Twitter -SuchBegriff PowerShell | Select-Object Title
```

Wenn Sie dieses Buch gewissenhaft durcharbeiten, werden Sie am Ende jedes Detail in diesem Skript verstehen (wenn *das* keine Motivation ist?). Wem das noch zu speziell ist, sollte als Erstes die Webseite der deutschsprachigen PowerShell-Anwendergruppe unter *http://www.powershell-ag.de* aufrufen. Hier lassen sich nicht nur Tutorials herunterladen und in den Foren Fragen stellen, die in der Regel kompetent und freundlich beantwortet werden, auf der Startseite können Sie auch die aktuellen Twitter-Meldungen lesen und sich auf diese Weise einen Eindruck vom Twitter-Phänomen verschaffen.

Kontakt zum Autor

Als »passionierter Schreiberling« freue mich stets über Zuschriften, Lob und natürlich auch Kritik (aber bitte in verträglicher Dosis) zum Buch und beantworte auch gerne Fragen rund um die PowerShell, wenngleich es dafür heutzutage jede Menge Anlaufstellen im Internet gibt. Unter der Adresse *http://powershellcrashkurs.wordpress.com* betreibe ich, bereits seit dem Erscheinen der ersten Fassung des Buchs, einen Blog, in dem es auch aktuelle Informationen und eventuelle Fehlerkorrekturen zu diesem Buch gibt. Hier sind auch die Beispielskripts abrufbar, die in diesem Buch vorgestellt werden.

Als freiberuflicher PowerShell-Trainer führe ich auch Inhousetrainings zu allen Themen rund um die PowerShell durch. Sie erreichen mich unter anderem über *pm@activetraining.de*.

Danksagungen

Es ist wieder einmal Zeit für Danksagungen, die für mich stets eine Herzensangelegenheit und alles andere als eine Formalität sind. Anders als in der ersten Auflage fasse ich mich dieses Mal etwas kürzer (und verzichte auf persönliche Referenzen), zumal es nicht zufällig dieselben Menschen sind, denen ich an dieser Stelle danken möchte. Da wäre zunächst mein Lektor *Florian Helmchen*, dem ich wieder einmal für die angenehme Zusammenarbeit (und wie beim letzten Mal für die Geduld, was den Abgabetermin angeht) danken möchte. *Thomas Irlbeck* hat das Buch wie beim letzten Mal souverän fachlektoriert und dabei noch so manche Unstimmigkeit entdeckt, die ich auch bei mehrmaligem Durchschauen übersehen hatte. Sollten wider Erwarten noch inhaltliche Fehler vorhanden sein, habe ich sie nachträglich wieder eingebaut. Die persönliche Widmung ist für meine Drea. Ohne sie wäre das Buch nicht zustande gekommen.

Esslingen, im Januar 2010

Peter Monadjemi: Windows PowerShell 2.0 - Crashkurs, Microsoft Press 2010 (ISBN 978-3-86645-667-9)